Model vrednovanja računalnog razmišljanja za koncepte apstrakcije, dekompozicije i algoritamskog razmišljanja temeljen na dokazu

Bubica, Nikolina

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

NIKOLINA BUBICA

MODEL VREDNOVANJA RAČUNALNOG RAZMIŠLJANJA ZA KONCEPTE APSTRAKCIJE, DEKOMPOZICIJE I ALGORITAMSKOG RAZMIŠLJANJA TEMELJEN NA DOKAZU

DOKTORSKI RAD

Split, 2022.

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

NIKOLINA BUBICA

MODEL VREDNOVANJA RAČUNALNOG RAZMIŠLJANJA ZA KONCEPTE APSTRAKCIJE, DEKOMPOZICIJE I ALGORITAMSKOG RAZMIŠLJANJA TEMELJEN NA DOKAZU

DOKTORSKI RAD

MENTOR: IZV. PROF. DR. SC. IVICA BOLJAT

Split, 2022.

FACULTY OF SCIENCE

NIKOLINA BUBICA

EVIDENCE CENTERED EVALUATION MODEL OF COMPUTATIONAL THINKING FOR CONCEPTS OF ABSTRACTION, DECOMPOSITION AND ALGORITHMIC THINKING

DOCTORAL THESIS

SUPERVISOR: ASSOC. PROF. IVICA BOLJAT PH. D.

Split, 2022.

Zahvale

Velika hvala mom mentoru, izv. prof. dr. sc. Ivici Boljatu koji me svojim konstruktivnim komentarima, velikom stručnošću, ali i zanimljivim raspravama vodio i usmjeravao od mojih prvih istraživačkih radova pa sve do izrade ove doktorske disertacije.

Posebno se zahvaljujem profesorici doc. dr. sc. Goranki Nogo na ukazanom povjerenju, pomoći tijekom istraživačkog rada te kvalitetnim raspavama koje su me još više motivirale za daljnji rad.

Hvala svim članovima povjerenstva: izv. prof. dr. sc. Saši Mladenoviću, izv. prof. dr. sc. Ani Grubišić, prof. dr. sc. Andrini Granić, prof. dr. sc. Marku Rosiću na trudu uloženom u vrednovanje ovog rada.

Hvala dragim kolegama koji su zainteresirano i aktivno sudjelovali u provedbi samog istraživanja.

Posebno se zahvaljujem svojoj obitelji koja me ohrabrila da krenem u ovom smjeru, vjerovala u mene i žrtvovala se zbog mene. Vi ste moja snaga, moja motivacija i moja ljubav.

Dragi moji Niko, Maja i Petre, dragi moji roditelji, ovaj rad posvećujem vama!

TEMELJNA DOKUMENTACIJSKA KARTICA

Sveučilište u Splitu Doktorska disertacija
Prirodoslovno-matematički fakultet
Poslijediplomski sveučilišni studij
„Istraživanje u edukaciji u području prirodnih i tehničkih znanosti“

MODEL VREDNOVANJA RAČUNALNOG RAZMIŠLJANJA ZA KONCEPTE APSTRAKCIJE, DEKOMPOZICIJE I ALGORITAMSKOG RAZMIŠLJANJA TEMELJEN NA DOKAZU

Nikolina Bubica

Prirodoslovno-matematički fakultet, Ruđera Boškovića 33, 21000 Split, Hrvatska

Sažetak
Posljednjih nekoliko godina u nastavnu praksu osnovnih i srednjih škola Republike Hrvatske uveden je novi kurikul predmeta Informatika koji u ishode učenja uvodi koncepte računalnog razmišljanja. Primjenom pristupa vrednovanju temeljenom na dokazu kreiran je model vrednovanja računalnog razmišljanja koji je usklađen s ishodima učenja novog informatičkog kurikula. U području istraživanja informatičkog obrazovanja sve je veći interes prema mogućnosti vrednovanja koncepata računalnog razmišljanja: koncepata apstrakcije, dekompozicije i algoritamskog razmišljanja u stvarnoj razrednoj situaciji te primjerima takvog vrednovanja. U tu svrhu provedeno je istraživanje. Podatci istraživanja prikupljeni su testom i upitnikom nad 407 učenika (10 osnovnih škola; uzrast učenika 12 godina) koji su analizirani istraživačkom faktorskom analizom i neparametrijskim testovima. Stvoreni model vrednovanja omogućuje učiteljima izgradnju vlastitog mjernog instrumenta u stvarnom vremenu koji naglašava koncepte računalnog razmišljanja, a ne sintaksu programskog jezika. Rezultati istraživanja pokazali su da je predstavljeni model prikladan za vrednovanje razumijevanja pojmova apstrakcije i algoritamskog razmišljanja, neovisno o prethodnom iskustvu rada s pojedinim programskim jezicima i spolu učenika dok su za vrednovanje dekompozicije predstavljene neke preporuke. Razvijeni model nudi i formativni oblik vrednovanja. Stvaranjem okvira vrednovanja koji se temelji na obrascima dizajna i dokazima znanja koje učenici demonstriraju u svojim rješenjima, omogućuje se analiza učeničkih mentalnih modela ključnih pojmova. Također, model vrednovanja dobio je pozitivne povratne informacije od učenika i učitelja što implicira da bi takav model vrednovanja mogao pomoći učiteljima u izgradnji mjernog instrumenta u stvarnoj razrednoj situaciji.
(198 stranica, 39 slika, 48 tablica, 182 literaturna navoda, jezik izvornika: hrvatski).
Rad je pohranjen u Sveučilišnoj knjižnici u Splitu, Ruđera Boškovića 31, Split te Nacionalnoj i sveučilišnoj knjižnici, Ul. Hrvatske bratske zajednice 4, Zagreb.

Ključne riječi: obrazovno vrednovanje, računalno razmišljanje, vrednovanje temeljeno na kurikulu, analiza jedinki testa, programiranje, faktorska analiza

Mentor: izv. prof. dr. sc. Ivica Boljat
Ocjenjivači:
1. Izv. prof. dr. sc. Saša Mladenović
2. Doc. dr. sc. Goranka Nogo
3. Izv. prof. dr. sc. Ani Grubišić
4. Prof. dr. sc. Andrina Granić
5. Prof. dr. sc. Marko Rosić

Rad prihvaćen: 22. prosinca 2021. godine

BASIC DOCUMENTATION CARD

University of Split Doctoral Thesis
Faculty of Science
Doctoral programme "Education Research in Natural and Technical Sciences"

EVIDENCE CENTERED EVALUATION MODEL OF COMPUTATIONAL THINKING FOR CONCEPTS OF ABSTRACTION, DECOMPOSITION AND ALGORITHMIC THINKING

Nikolina Bubica

Faculty of Science, Ruđera Boškovića 33, 21 000 Split, Croatia

Abstract
The new Croatian Informatics curriculum, which introduces computational thinking concepts into learning outcomes has been put into practice. A computational thinking assessment model reflecting the learning outcomes of the Croatian curriculum was created using an evidence-centered design approach. The possibility of assessing the computational thinking concepts, abstraction, decomposition, and algorithmic thinking, in an actual classroom situation and examples of such assessment is increasingly coming to the forefront of computer science educational research. Precisely for that purpose, the research was conducted. Research data are collected through the test and questionnaire of 407 pupils (10 middle schools, age 12), analysed by exploratory factor analysis and non-parametric tests. The created assessment model allows teachers to build their own real-time measurement instrument that emphasizes computational thinking concepts rather than programming language syntax. Results showed that the presented model was suitable to assess the understanding of the concepts of abstraction and algorithmic thinking, independently of the previous experience with programming languages and pupil's gender, while for assessment of decomposition some recommendations are provided. The developed assessment model also enables formative assessment. By creating an assessment framework based on design patterns and knowledge evidence that students demonstrate in their solutions, it enables the analysis of students' mental models of key concepts. Also, it received positive feedback from students and teachers what implicated that such an assessment model could help teachers in building a real-time measurement instrument.
(198 pages, 39 images, 48 tables, 182 references, original in Croatian).
Thesis is deposited in the University Library of Split, Ruđera Boškovića 31, Split and National and University Library, Ul. Hrvatske bratske zajednice 4, Zagreb.

Keywords: educational testing, computational thinking, curriculum-based assessment, item analysis, programming, factor analysis

Supervisor: associate professor Ivica Boljat Ph. D.
Reviewers:
1. Associate Professor Saša Mladenović, Ph. D.
2. Assistant Professor Goranka Nogo, Ph. D.
3. Associate Professor Ani Grubišić, Ph. D.
4. Professor Andrina Granić, Ph. D.
5. Professor Marko Rosić, Ph. D.

Thesis accepted: 22 December, 2021.

SADRŽAJ

Popis slika

Popis tablica

1. UVOD

Tehnologija je sveprisutna u našem životu. Upravljanje podatcima i računalna znanost polako postaju temelj informatičkog društva, ali i društva u cjelini. Bez obzira na buduća zanimanja mladih ljudi, njihovu dob ili vrstu tehnologije koju koriste, od njih se sve češće očekuje da posjeduju neke generičke kompetencije kao što su sposobnost svakodnevnog rješavanja problema, razdvajanja složenih problema na jednostavnije, generaliziranje rješenja i slično. Takve kompetencije koje su temelj računalnog razmišljanja sve više nadilaze područje predmetnog kurikula nastojeći ponuditi učenicima novi alat u njihovom razmišljanju i razumijevanju digitalnog, fizičkog te društvenog svijeta koji ih okružuje.

Primjena računalnog razmišljanja kroz modeliranje i simulaciju već sada snažno utječe na istraživanja u biološkim, društvenim i prirodnim znanostima. Na primjer, modeli neurona pomažu nam u razumijevanju načina na koji mozak radi; evolucijski modeli pomažu objasniti biološke pojave; modeli zasnovani na agentima opisuju društvene sustave; računalni modeli omogućuju znanstvenicima način razumijevanja klime. Istraživanjem takvih pojava stvara se bogato interdisciplinarno područje djelovanja u kojem se potiče razvoj računalnog razmišljanja te osnažuje interes prema računalstvu općenito (Interdisciplinary Computational Thinking, 2017).

Prepoznajući važnost računalnog razmišljanja i njegovog uvođenja u osnovno i srednjoškolsko obrazovanje, učitelji informatike nalaze se pred izazovnim zadatkom uključivanja tih koncepata u svoje poučavanje (Csizmadia i sur., 2015). Sve se više naglašava potreba da programi podučavanja i usavršavanja učitelja informatike moraju obuhvatiti rasprave o računalnom razmišljanju te pružiti preporuke i primjere najboljih praksi za uključivanje koncepata računalnog razmišljanja u obrazovnu praksu (Yadav, Stephenson i sur., 2017). Ističe se potreba za boljom suradnjom između učitelja informatike te znanstvenih ustanova koje se bave poučavanjem učitelja informatike kako bi se općenito unaprijedilo obrazovanje budućih te napredovanje sadašnjih učitelja (Yadav, Gretter i sur., 2017).

U ožujku 2018. hrvatsko MZO objavilo je novi kurikul predmeta Informatika za osnovne i srednje škole. Većina se učitelja i nastavnika osjećala sputano starim i zastarjelim Nastavnim planom i programom (NPiP) koji je već dulje vrijeme bio u primjeni. Novi kurikul predmeta Informatika napokon je uključio računalno razmišljanje kao važan dio informatičkog obrazovanja s osnovnim ciljem poticanja učenika na logičko razmišljanje, modeliranje, apstrahiranje i rješavanje problema. Takvim informatičkim obrazovanjem koje se temelji na

računalnom razmišljanju i kreativnosti, trebalo bi pripremiti učenike za razumijevanje promjena u svijetu oko nas (Ministarstvo znanosti i obrazovanja, 2018).

Tijekom protekle tri godine svjedoci smo primjene novog kurikula Informatike u sklopu obrazovne reforme u Republici Hrvatskoj. Reforma predstavlja promjenu u procesu poučavanja i učenja. Novo područje predmetnog kurikula, područje računalnog razmišljanja, od izuzetnog je interesa u znanstvenoj zajednici informatičkog obrazovanja. Iako se danas može pronaći dosta radova i rasprava koji se bave pitanjima njegove definicije i poučavanja, činjenica da još uvijek nema formalne definicije računalnog razmišljanja značajno otežava napredak tog područja. Vrednovanje računalnog razmišljanja stoga postaje poseban izazov informatičkom obrazovanju. Pri vrednovanju takvih koncepata traži se pronalazak dokaza o dubinskom razumijevanju problema kojeg je učenik riješio. Osnovni je zadatak prepoznati i razumjeti način na koji je učenik nadogradio ili stvorio neko rješenje. Upravo vrednovanje takvih, teško mjerljivih konstrukata traži sustavni pristup.

U ovom radu predstavljen je model vrednovanja računalnog razmišljanja temeljen na dokazima (engl. Evidence-Centered Design) i usklađen s hrvatskim predmetnim kurikulom. U provedenom vrednovanju računalnog razmišljanja stavljen je naglasak na koncepte apstrakcije, algoritamskog razmišljanja i dekompozicije. Predstavljeni su rezultati njegove provedbe, rasprava o samoj primjeni i rezultatima te kroz zaključak istaknuti ključni doprinosi ovog rada. Razvijeni model vrednovanja ima za cilj omogućiti stvaranje kvalitetnih zadataka vrednovanja računalnog razmišljanja primjenjivih u svakodnevnoj nastavnoj praksi formativnog i sumativnog vrednovanja znanja.

2. TEORIJSKI OKVIR

U ovom poglavlju definirat će se pojam računalnog razmišljanja te koncepti koje ono uključuje. Predstavit će se postupci poučavanja te vrednovanja računalnog razmišljanja. Opisat će se djelovanje nekih osobnih obilježja učenika koja mogu utjecati na uspjeh pri učenju, a odnose se na inteligenciju i učenje, sposobnost rješavanja problema te razmatranje usvojenih mentalnih modela kod učenika.

Zbog snažne povezanosti računalnog razmišljanja i programiranja, u promišljanju o uspjehu poseban je naglasak stavljen na faktore koji mogu značajno utjecati na postizanje uspjeha pri rješavanju problema programiranjem. Računalno razmišljanje i programiranje nisu isti pojmovi, ali su jako povezani. Znanje programiranja uključuje sposobnost čitanja i pisanja u nekom programskom jeziku (Roman-Gonzales, 2014) dok je računalno razmišljanje ključna vještina rješavanja problema koja to omogućuje. Može se reći da je programiranje važno za omogućavanje računalnog razmišljanja (Lye i Koh, 2014), ali i da se računalno razmišljanje može primijeniti na različite vrste problema koji direktno ne uključuju zadatke programiranja (Wing, 2008).

2.1. Računalno razmišljanje

2.1.1. Definicija računalnog razmišljanja

Još uvijek postoje veliki prijepori oko same definicije te čitavog niza pitanja i izazova koje treba adresirati kad se govori o računalnom razmišljanju (engl. Computational Thinking). Ideje računalnog razmišljanja nisu nove, već ih pronalazimo još kod Paperta. U svojim radovima o Logo programiranju i djeci koja upravljaju računalima, Papert ističe da se upravo programiranjem razvija proceduralno razmišljanje (Papert, 1980). Vjerovao je da će učenici, uporabom tehnologije i programiranja za kreiranje mikrosvjetova, razviti vještine koje će se dalje prenijeti na ne-programerski kontekst, pa čak i izvan nastavnog okružja. Mali broj znanstvenika prihvaća takve ideje te smatra da " ….ideja da će programiranje omogućiti vježbanje za najvišu mentalnu sposobnost te da će se kognitivni razvoj koji će nam osigurati programiranje, generalizirati ili prenijeti na druga područja u životu djeteta – samo nada." (Pea, 1987, str. 13). Neka su istraživanja potvrdila da su učenici, koji su nekoliko mjeseci razvijali vještine popravljanja programa (engl. debugging) u programskom jeziku Logo, pokazali jasno unapređenje vještina proceduralnog razmišljanja u ne-programerskom okružju (Swan, 1989; Carver, 1986; Klahr i Carver, 1988).

U novije vrijeme pokazalo se da poučavanje pristupom temeljenim na mikrosvjetovima i računalnom razmišljanju može pozitivno utjecati na razvoj jezične pismenosti (Jenkins, 2013).

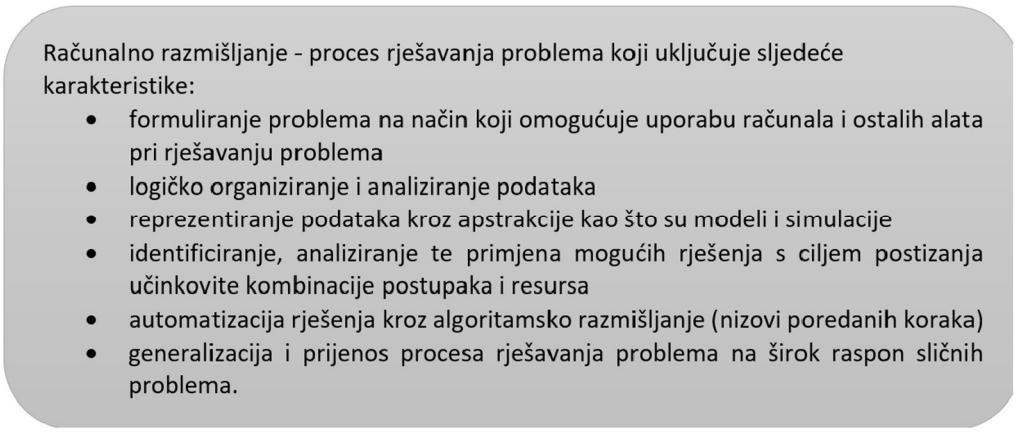
Dok Papert gotovo izjednačava računalno razmišljanje s programiranjem u konkretnom programskom jeziku Logo, današnja promišljanja o računalnom razmišljanju sve se više fokusiraju na uporabu njegovih općenitih koncepata izvan računalne znanosti (Lu i Fletcher, 2009).

Računalno razmišljanje smatra se nekom univerzalnom kompetencijom svakog djeteta koja bi uz analitičke sposobnosti bila temelj školskog učenja svakog djeteta (Wing, 2006). Nadalje, ističe se kako računalno razmišljanje nadopunjuje matematičko i inženjersko razmišljanje s naglaskom na vještini rješavanja složenih problema s kojima se ljudi susreću u svakodnevnom životu (Wing, 2006).

Denning (2009) raspravlja o tome pripada li računalno razmišljanje isključivo u domenu računalne znanosti te ističe: „Pojam računalnog razmišljanja već je toliko proširen da uključuje razmišljanje s mnogim razinama apstrakcije, korištenje matematike za razvoj algoritama te ispituje koliko se dobro rješenje može prilagoditi problemima različitih veličina….. To nije princip već praksa. Računalno razmišljanje jedna je od ključnih kompetencija koje bi računalni znanstvenici trebali posjedovati.“ (Denning, 2009, str. 2).

Guzdial (2008) naglašava da računalno razmišljanje treba promatrati veoma široko te ga čak naziva pismenošću 21. stoljeća koja je neophodna na čitavom nizu fakulteta (Guzdial, 2008). Wing (2010) dalje nadopunjuje svoju definiciju računalnog razmišljanja i kaže: „Računalno razmišljanje je misaoni proces koji je uključen u proces formuliranja problema i njihovih rješenja, ali na način da rješenja budu prezentirana u obliku koji omogućuje njihovo učinkovito izvođenje nekim agentom za obradu informacija“ (Wing, 2010, str. 1).

U raspravama o računalnom razmišljanju često se postavlja pitanje na koji se način ono razlikuje od algoritamskog razmišljanja, no upravo Denning (2010, str. 371) jasno povezuje ta dva načina razmišljanja te ističe da „…razmišljati računalno znači interpretirati problem kao informacijski proces kojem tada nastojimo pronaći algoritamsko rješenje“ (Denning, 2010). U cilju stvaranja operativne definicije računalnog razmišljanja ISTE i CSTA organizacije analiziraju povratne informacije od oko 700 anketiranih učitelja te istraživača računalne znanosti (Slika 1).



Slika 1: Definicija računalnog razmišljanja prema ISTE i CSTE, 2011

Takva operativna definicija (ISTE i CSTA, 2011) imala je za cilj ponuditi odgovarajući okvir i rječnik pri razmatranju računalnog razmišljanja na razini K12 obrazovanja. Ova operativna definicija računalnog razmišljanja još je uvijek jedna od najšire prihvaćenih definicija.

Pri razmatranjima računalnog razmišljanja, uzimajući u obzir zahtjevnost vještina i procesa koji predstavljaju takvo razmišljanje, ističu se i stavovi kako bi možda bilo primjerenije raditi na većoj dostupnosti i prilagođenosti računala onima koji ne vole razmišljati računalno, nego mijenjati korisnike s ciljem poboljšanja njihove sposobnosti da razmišljaju računalno. Blackwell i sur. naglašavaju da je zaista izuzetno korisno primjenjivati apstraktne formalne opise kakve zagovara računalno razmišljanje, prvenstveno kroz razvoj apstrakcije (Blackwell i sur., 2008).

Također, ističu da razvoj računalnog razmišljanja ne bi trebao isključivo krenuti u smjeru promatranja apstrakcije kao nečeg što je čovjeku prirodno i „prijateljsko“, već istraživati nove alternativne pristupe koji će omogućiti krajnjem korisniku stvaranje programa koji se ne oslanjaju isključivo na apstrakciju. Slično, Aho (2012) definira računalno razmišljanje kao sklop misaonih procesa uključenih u pronalaženje odgovarajućih računalnih modela kojima se formuliraju problemi i rješenja, prikazani kao niz računalnih postupaka i algoritama (Aho, 2012).

U novije vrijeme u poučavanju programiranja sve su više prisutna različita grafička okružja. Primjenom takvih okružja koja omogućuju mladim ljudima samostalno kreiranje interaktivnih priča, igara i simulacija, pojavila su se nastojanja da se aktivnosti učenja temelje na dizajnu, npr. na dizajnu interaktivnih medija u nekom bliskom kontekstu kao što je to npr. grafičko okružje programiranja Scratch.

Takvim pristupom računalno razmišljanje može se definirati kroz tri osnovne dimenzije (Brennan i Resnick, 2012):

- dimenziju računalnih koncepata (koncepti koje koriste dizajneri dok programiraju: slijed, petlje, paralelizam, događaji, uvjeti, operatori i podatci)
- dimenziju računalne prakse (praksa koju razvijaju dizajneri dok programiraju: testiranje i ispravljanje, ponovo korištenje i miješanje, apstrahiranje i moduliranje te iterativnost i nadogradivost)
- dimenziju računalne perspektive (perspektiva dizajnera o svijetu koji ih okružuje te o njima samima: izražavanje, povezivanje i propitivanje).

U daljnjim nastojanjima pronalaska što jednostavnije definicije računalnog razmišljanja Royal Society ističe osnovnu ideju računalnog razmišljanja kao „.. procesa prepoznavanja aspekata računalstva u svijetu koji nas okružuje te procesa primjene alata i tehnika računalne znanosti za razumijevanje te razmišljanje o prirodnim i umjetnim sustavima i procesima…“ (Royal Society , 2012). Grover i Pea (2013) sintetiziraju promišljanja o računalnom razmišljanju u relevantnoj literaturi te ga definiraju kao (Grover i Pea, 2012):

- apstrakciju i generalizaciju uzoraka (uključujući modeliranje i simulaciju)
- sustavnu obradu podataka
- sustav simbola te njihove prezentacije
- algoritamsku notaciju tijeka kontrole
- strukturiranu dekompoziciju problema
- iterativno, rekurzivno te paralelno razmišljanje
- uvjetovanu logiku
- ograničenja učinkovitosti te izvedbe
- ispravljanje te sustavno otkrivanje pogrešaka.

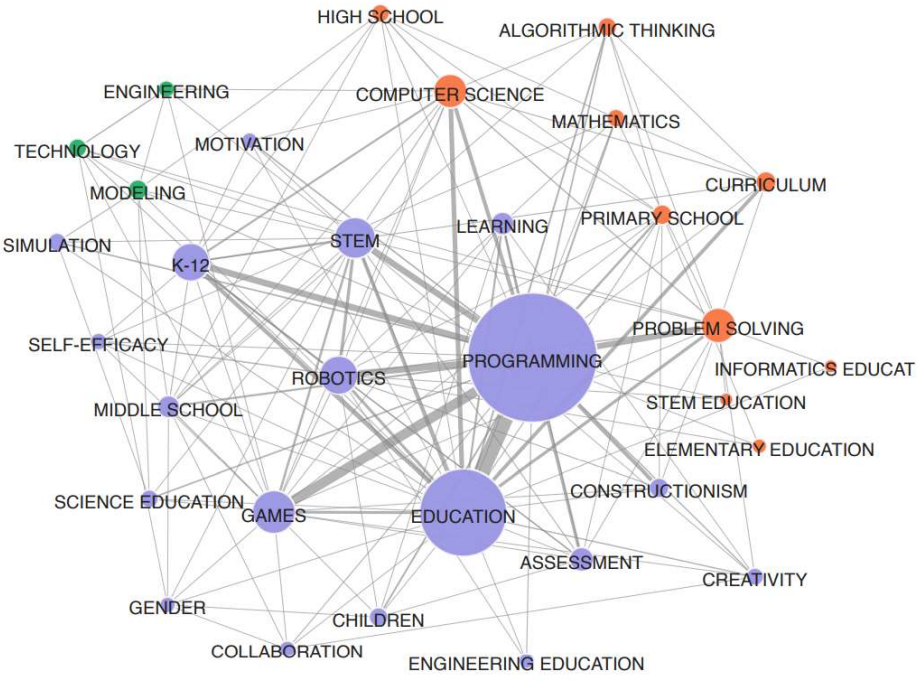
U ovom poglavlju predstavljene su definicije računalnog razmišljanja koje su trenutno prisutne u stručnoj literaturi. Iako ne postoji dogovor oko formalne definicije računalnog razmišljanja među navedenim definicijama može se uočiti njihova snažna povezanost te brojni zajednički koncepti na koje se definicije odnose. Sve se više ističe interdisciplinarna primjena računalnog razmišljanja kroz predmete kao što su biologija, kemija, fizika, filozofija, matematika, glazba i ples te strane jezici (Interdisciplinary Computatonal Thinking, 2017).

Ipak, temeljni koncepti računalnog razmišljanja, dakle apstrakcija, dekompozicija, algoritamsko razmišljanje, generalizacija te modeliranje usko se povezuju s procesom programiranja, planiranjem ili izvođenjem nekog programa odnosno niza naredbi prilagođenih za izvođenje na računalu. U ovom radu pod terminom programiranja razmatraju se vještine

rješavanja problema programiranjem, odnosno analiziranje problemskog zadatka formuliranjem algoritamskog rješenja zadanog problema, vrednovanje te implementiranje algoritma u nekom programskom jeziku.

U svim navedenim definicijama računalnog razmišljanja možda najviše zbunjuje uloga programiranja te može li se ono odvojiti od poučavanja osnovne računalne znanosti.

Djelomičan odgovor na to pitanje može ponuditi bibliometrijska meta-analiza autora i tema provedena nad 1874 istraživačka rada koji sadrže ključni pojam „računalno razmišljanje“, a preuzeti su iz baze istraživačkih radova Scopus (Saqr i sur., 2021). Jedan od rezultata analize prikazan je sljedećom slikom (Slika 2). Slika ističe najčešće ključne riječi analiziranih istraživačkih radova koje se vežu uz pojam računalnog razmišljanja, nadalje broj ponavljanja pojedine ključne riječi (veličina kruga), učestalost veze dvaju povezanih pojmova (debljina poveznice) te pripadnost klasteru ključnih riječi (boja kruga). Očigledno je da se uz termin računalnog razmišljanja najviše spominje pojam programiranja i K12 obrazovanja te se ističe snažna povezanost upravo tih triju pojmova.



Slika 2: Povezanost ključnih riječi s pojmom “računalno razmišljanje” prema (Saqr i sur., 2021)

U klasteru koji povezuje školske koncepte (ljubičasti klaster) najčešće se spominju pojmovi kao što su obrazovanje, robotika, STEM, vrednovanje, učenje, kreativnost, konstruktivizam, igre, suradnja, inženjerstvo i sl. Klaster koji povezuje tradicionalne računane teme/discipline (crveni klaster) najčešće spominje teme kao što su rješavanje problema, računalna znanost, matematika, algoritamsko razmišljanje, osnovnoškolsko i informatičko obrazovanje, kurikul i sl. dok su od koncepata povezanih s obrazovnom stranom istraživanja informatičkog obrazovanja (zeleni klaster) najčešće spominju modeliranje, inženjerstvo i tehnologija. Iz navedenog se može potvrditi da se u stručnoj literaturi uz razmatranja računalnog razmišljanja vrlo često raspravlja i o programiranju.

Mogu li se računalno razmišljanje i programiranje jasno razdvojiti? Iz ranije navedenog može se reći da programiranje i računalno razmišljanje predstavljaju dva skupa koja dijele veći broj zajedničkih pojmova i vještina. U procesu rješavanja problema programiranjem naglasak je na stvaranju računalnog rješenja u nekom programskom jeziku, dakle na formuliranju jasnih i sistematičnih uputa koje naređuju računalu da izvrši neki zadatak. Ako se pri rješavanju problema stavlja naglasak na razvoj vještina računalnog razmišljanja krajnji rezultat može, ali i ne mora biti računalno rješenje. To mogu biti tekstualne skice, dijagram tijeka, nadalje različiti grafički prikazi u kojima su naznačeni koraci rješavanja, plakati, modeli, simulacije i slično.

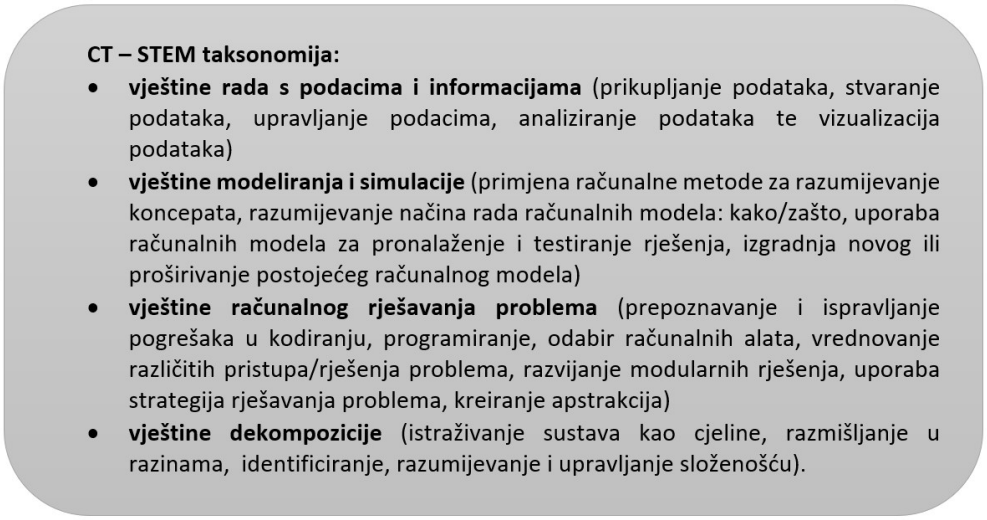
Što znači riješiti problem vještinama računalnog razmišljanja? Problem je potrebno analizirati kako bi se mogao formulirati na način koji je rješiv primjenom računala. Prepoznaju se manji, jednostavniji dijelovi problema koji se dalje razvijaju. Promišlja se o podatcima koji će se koristiti pri rješavanju problema. Predviđa se način prikupljanja podataka te prepoznaju uzorci među podatcima. Identificiraju se i uklanjaju nepotrebni detalji (proces apstrahiranja) te stvaraju odgovarajući modeli kako bi se olakšao proces stvaranja rješenja. Definiraju se koraci za rješavanje problema (stvaranje algoritma). Procesom vrednovanja te dodatnog uređivanja rješenja ispravljaju se eventualne pogreške te nadograđuje i unapređuje rješenje. Prepoznaju se problemi koji se mogu riješiti na sličan način (proces generalizacije). Iz svega navedenog slijedi da je računalno razmišljanje puno veća i šira priča od programiranja.

Često se u opisima i definicijama računalnog razmišljanja povezuju duboko i apstraktno razumijevanje temeljnih koncepata računalne znanosti. Pod dubokim razumijevanjem podrazumijeva se sposobnost prepoznavanja temeljnih koncepata približenih kroz programerske koncepte, a apstraktno razumijevanje kao razdvajanje stvarnog smisla od samih sintaktičkih detalja (Touretzky i sur., 2013).

Neki autori definiraju računalno razmišljanje kao temeljni sveprisutni alat za rješavanje problema te predlažu nekoliko aktivnosti i projekata kojima žele definirati te adresirati računalno razmišljanje (Astrachan i sur., 2009). U pristupima koje zagovaraju, predlažu različite načine uključivanja programiranja u poučavanje računalnog razmišljanja. Pristupi variraju od onih u kojima programiranje predstavlja temeljnu vještinu računalnog razmišljanja pa sve do pristupa u kojima se računalno razmišljanje integrira kroz različite kolegije općeg obrazovanja.

Uvođenjem vještina računalnog razmišljanja među nužne vještine obrazovanja novih generacija učenika, pojavila se potreba da se što bolje definira pojam računalnog razmišljanja u širem kontekstu. Kao što se iz ranije navedenog može primijetiti, rasprave o računalnom razmišljanju uglavnom se vrte oko temeljnog pitanja: „Kako definirati računalno razmišljanje?“. Weintrop i sur. (2016) idu dalje i razmatraju: „Kako se računalno razmišljanje razlikuje od ostalih oblika razmišljanja kao što je matematičko razmišljanje, algoritamsko razmišljanje te rješavanje problema?“ te „Zahtijeva li takvo razmišljanje programiranje te koliko? Zahtijeva li računalo?“ (Weintrop i sur., 2016).

U razmatranju ovih pitanja fokusirali su se na područje STEM-a, te ponudili cjelovitu taksonomiju vještina računalnog razmišljanja u STEM području, odnosno definiciju za vještine koje čine računalno razmišljanje u STEM-u (CT-STEM). U svom radu posebnu su pažnju posvetili potrebama učitelja od kojih se očekuje da praksu računalnog razmišljanja uspješno integriraju u svoje kolegije. Štoviše, CT-STEM taksonomija vještina, između ostalog, nastala je na temelju prakse uporabe računalnog razmišljanja u profesionalnom i obrazovnom STEM radu i razdvojena je u četiri kategorije (Slika 3).



Slika 3: CT-STEM taksonomija

Iako zanimljiva, predložena taksonomija još uvijek je u razvoju i daljnja bi istraživanja trebala otkriti njenu snagu.

O definiciji računalnog razmišljanja još uvijek traju rasprave. Iako mnogi istraživači te učitelji računalne znanosti teže naglašavanju snage i važnosti računalnog razmišljanja kako za informatičko, tako i za opće obrazovanje, treba naglasiti sljedeće: usprkos velikim očekivanjima još uvijek nema potvrde da će se sposobnost rješenja problema, razvijena kroz programiranje, prenijeti i na ostala područja; pretjerane i presnažne tvrdnje o samom računalnom razmišljanju te njegovoj ulozi, kako u računalnoj znanosti tako i u ostalim znanostima, mogu samo štetiti pokretu koji se njime bavi, odnosno istražuje (Tedre i Denning, 2016).

U ovom radu pod pojmom računalnog razmišljanja razmatrat će se način rješavanja problema kojeg obilježava:

- promišljanje o podatcima koji ga opisuju
- prepoznavanje različitih razina problema
- uočavanje obrazaca (uzoraka) i odnosa među podatcima koji se koriste te zadatcima nad podatcima koji je potrebno izvršiti
- stvaranje rješenja problema koji jasno ističe korake i redoslijed izvršavanja
- promišljanje o široj uporabi ili nadogradnji rješenja prepoznavanjem sličnih problema koji se tako mogu riješiti.

2.1.2. Osnovni koncepti i prakse računalnog razmišljanja

Kao što je već u prethodnim poglavljima navedeno, postoje mnogi izazovi i još uvijek otvorena pitanja uz samo definiranje pojma računalnog razmišljanja. Slično se može primijetiti i s temeljnim konceptima računalnog razmišljanja. Prema Wing (2008) temeljni koncepti računalnog razmišljanja uključivali bi (Wing, 2008):

- apstrakciju – jer je to mentalni alat neophodan za rješavanje problema, osobito u računalstvu, zatim
- razine - jer se problemi moraju promatrati i rješavati na različitim razinama te
- povezanost između apstrakcija i razina.

Navedena sistematizacija stavlja naglasak na apstrakciju kao jednom od važnih koncepata računalnog razmišljanja, ali nedovoljno detaljizira razine te način uspostavljanja veza između razina i apstrakcije.

Selby i Woolard (2013) pregledom relevantne literature predlažu sljedeće koncepte računalnog razmišljanja:

- algoritamsko razmišljanje, kao način dolaženja do rješenja jasno definiranim koracima koje je potrebno napraviti određenim redoslijedom
- vrednovanje, proces kojim se osigurava da predloženo algoritamsko rješenje zaista odgovara zahtjevima (točnost, brzina izvođenja, ekonomičnost uporabe rekurzije, jednostavnost uporabe za korisnika, promoviranje primjerene prakse)
- dekompoziciju, način razmišljanja o problemu, algoritmima, artefaktima, procesima te sustavima sa stanovišta njegovih dijelova
- apstrakciju, drugačiji i jednostavniji način prikazivanja nekog problema ili sustava
- generalizaciju, način brzog rješavanja novih problema na temelju ranije riješenih problema (Selby i Woollard, 2013).

U kakvom su odnosu računalno razmišljanje i računalna znanost? Računalna se znanost prvenstveno bavi informacijama te njihovom obradom, pohranom i prezentacijom, načinima koji uključuju računala. Kao što je već navedeno, računalno razmišljanje obuhvaća vještine i znanja kao što su dijeljenje problema na manje dijelove, prepoznavanje te pronalaženje uzoraka među podatcima ili algoritamsko razmišljanje, odnosno stvaranje niza uputa za rješavanje nekog problema.

Jasno je da se ove vještine lako mogu primijeniti u računalnoj, ali i nekim drugim znanostima. Tako se dijeljenje problema na manje dijelove u računalnoj znanosti može realizirati

prepoznavanjem manjih dijelova problema koji se rješavaju, npr. potprogramima, pri čemu svi potprogrami, povezani zajedno, čine cjelovito rješenje problema.

Također u drugim znanostima, npr. književnosti, dijeljenje problema na manje dijelove može se realizirati kroz planiranje školskih novina pri čemu je potrebno uočiti/prepoznati manje aktivnosti kao što su planiranje i realizacija dizajna školskih novina, prikupljanje i uređivanje sadržaja, lektoriranje te objavljivanje materijala. Sve navedene aktivnosti, bilo da se radi o problemima izvan ili unutar računalne znanosti, mogu se realizirati tako da se prepoznaju manji dijelovi problema koji se zasebno rješavaju uz uvjet da svi dijelovi zajedno predstavljaju cjelovito rješenje problema.

Nadalje, vještina pronalaženja uzorka u računalnog znanosti očituje se pri stvaranje kvalitetnih računalnih rješenja u kojima se prepoznaju nizovi naredbi za koje vrijedi zajedničko pravilo. Primjer tome može biti crtanje pravilnog mnogokuta (konačan broj ponavljanja crtanja stranice mnogokuta te odgovarajućeg unutarnjeg kuta mnogokuta umjesto slijeda naredbi kojima se programira zasebno crtanje svake stranice mnogokuta). Prepoznavanjem odabranog uzorka (niza naredbi) stvaraju se programi koji su kraći, fleksibilniji, pregledniji i razumljiviji od početnog rješenja.

Vještina prepoznavanja uzoraka izvan računalne znanosti može se primjenjivati npr. u razvoju jezičnih kompetencija. Pri učenju nekog stranog jezika uče se nove riječi te gradi jezični leksikon, ali upravo izučavanjem gramatičkih uzoraka usvajaju se fraze ili rečenice koje se koriste sa zadanim pridjevom, imenicom ili glagolom i omogućuje funkcionalno korištenje novog znanja.

Vještine računalnog razmišljanja nužne su za računalnu znanost, ali jasno je da su od velike važnosti i za ostale znanosti. Vještinom algoritamskog razmišljanja u računalnoj znanosti razvijaju se vještine stvaranja računalnog programa koji će izvršiti određeni zadatak. U nekoj drugoj znanstvenoj disciplini ili svakodnevnom životu takav način razmišljanja unapređuje sposobnost definiranja niza specifičnih uputa kojima se rješava neki problem/zadatak, kao što je npr. zadavanje preciznih uputa korisnicima za izradu kulinarskog obroka, zadavanje preciznih uputa arhiviranja, slaganja i pretraživanja pojedinog sadržaja (knjiga, medija,…) i slično. Prema Wing (2010) računalno razmišljanje u svakodnevnom životu znači „…

- razumjeti koji su aspekti problema računalno zanimljivi
- vrednovati usklađenost pojedinih računanih alata i tehnika s problemom
- razumjeti ograničenja i snagu računalnih alata i tehnika
- primijeniti i prilagoditi računalne alate i tehnike nekoj novoj primjeni

- prepoznati mogućnosti uporabe računalstva na neki novi način
- primijeniti računalne strategije kao što su Podijeli i vladaj (engl. Divide and conquer) u nekom drugom području“ (Wing, 2010).

U ovoj raspravi o konceptima računalnog razmišljanja vidi se da autori imaju različite pristupe, ali i da su koncepti oko kojih postoji suglasnost većine relevantnih autora sljedeći: apstrakcija, prepoznavanje uzoraka i rad s podatcima, prikazivanje rješenja algoritmom, simulacija i generalizacija (DIMACS, 2021).

Pod konceptima računalnog razmišljanja u ovom radu razmatraju se apstrakcija, algoritamsko razmišljanje, dekompozicija, vrednovanje i generalizacija (Selby i Woollard, 2013). U modelu vrednovanja razvijen je alat koji detaljno razmatra sljedeće koncepte:

- apstrakcija: razumijevanje problema, identifikacija ključnih točaka/ograničenja problema, rad s logičkim vrijednostima, razmišljanje na različitim razinama apstrakcije, rad s varijablama i ulazom/izlazom
- dekompozicija: rastavljanje problema na manje složene probleme, stvaranje novih rješenja na temelju poznatih problema
- algoritamsko razmišljanje: analiziranje i praćenje algoritma, rad s različitim algoritamskim strukturama, predviđanje ponašanja algoritma, primjena strukture grananja, uređivanje i nadogradnja algoritamskih rješenja, stvaranje novih rješenja na temelju postojećih sličnih rješenja.

Navedeni koncepti alata vrednovanja odabrani su u skladu s trenutno važećim Nastavnim planom i programom predmeta Informatika za šesti razred u trenutku provedbe istraživanja.

2.2. Inteligencija i učenje

U današnjem svijetu brzih promjena i tehnološkog napretka vještine kao što su rješavanje problema, analiza valjanosti rješenja te prepoznavanje uzoraka među podatcima sve se više ističu kao važne vještine za zanimanja 'budućnosti' te se za njih traži odgovarajuće mjesto u procesu poučavanja.

Iz prethodnih poglavlja jasno je da su navedene vještine sastavni dio vještina opisanih pod pojmom računalnog razmišljanja, ali i da se ne smije sugerirati da su bitne samo budućim informatičkim stručnjacima već imaju svoju interdisciplinarnu primjenu. Razvojem vještina računalnog razmišljanja pojedincu se omogućuje iskazivanje kreativnosti i inovativnog načina razmišljanja jer ga se priprema za lakšu prilagodbu novim idejama, za stvaranje različitih

rješenja zadanog problema, za pronalazak svog vlastitog puta do rješenja te pojednostavljivanje zadanog problema.

Definicija računalnog razmišljanja povezana je i s pojmom inteligencije. Oba pojma povezuje sposobnost rješavanja problema i apstrahiranje. U računalnoj znanosti naglašavaju se značajke kao što su rukovanje podatcima, apstrahiranje problema, sposobnost dizajniranja i primjene algoritama. U novijoj stručnoj literaturi koja se bavi inteligencijom često se može pročitati da je inteligencija 'modernog' čovjeka potpomognuta umjetnom inteligencijom daleko snažnija i moćnija od inteligencije naših predaka. Većina je stručnjaka suglasna da inteligencija nije stvar već način djelovanja (Zarevski, 2012), no tek se u novije vrijeme u raspravama o inteligenciji veći naglasak stavlja na mentalne produkte, a ne na procese.

Danas su prisutne velike razlike u razmišljanjima i stavovima među stručnjacima u raspravama o inteligenciji. Mnoge razlike u stavovima dolaze zbog različite dobi i spola ispitanika koji se promatraju, ali i zbog međusobne interakcije tih dvaju faktora. Opće je prihvatljiv stav da je inteligencija prisutna kod niza ljudskih aktivnosti: profesionalnih, socijalnih, znanstvenih, sportskih i sl. Pri definiciji inteligencije ističu se sljedeći aspekti: sposobnost apstraktnog mišljenja, adaptabilnost te samokritičnost (Terman, 1916). Gardner naglašava multipersonalnu definiciju inteligencije (lingvistička, logičko-matematička, prostorna, glazbena, tjelesno-kinestetička, interpersonalna, intrapersonalna i prirodoslovna inteligencija) (Gardner, 1983). U novije vrijeme Sternberg definira pojam uspješne inteligencije kao poznavanje jačih i slabijih strana vlastite inteligencije s ciljem što snažnijeg poticanja jačih strana, te poboljšavanja slabijih strana inteligencije (Sternberg, 1996).

Strukturalističkim pristupom definirati inteligenciju znači otkriti strukturu kognitivnih sposobnosti pri čemu se najviše ističe pristup primjene faktorske analize. Takvim pristupom nastoji se prepoznati dovoljan skup faktora (mentalnih aktivnosti) koje objašnjavaju kognitivno djelovanje. Nastavkom ovog pristupa može se smatrati sveobuhvatna klasifikacija kognitivnih sposobnosti korištenjem rezultata testova i nekih drugih indikatora iz psihologije i njoj srodnih područja.

Pristup faktorskom analizom te pristup klasifikacijom i danas je aktualan upravo zbog značajne uporabe računala u znanosti. Upravo su računala s velikim brzinama obrade informacija te značajnim memorijskim kapacitetom omogućila primjenu multivarijatnih regresijskih analiza velikog broja podataka koje klasičnim načinom bez uporabe računala nisu bile moguće. Takvom statističkom metodom omogućuje se kvantitativno izražavanje ovisnosti (korelacije)

među varijablama. Formiraju se modeli kojima se mogu predviđati pojedini podatci koji se ne promatraju ili ne mjere.

Primjer takvog model je model inteligencije Cattel-Horn-Caroll (CHC) koji na temelju velikog broja istraživanja, kroz više od 70 godina, otkriva povezanost među varijablama i temeljnu strukturu koncepata poput "inteligencije". Sa stanovišta kognitivnog pristupa računalno razmišljanje povezano je s tri sposobnosti/faktora CHC modela inteligencije (McGrew, 2009):

- tečnim razmišljanjem (engl. fluid reasoning) odnosno „*sposobnosti izvršavanja namjernih i kontroliranih mentalnih operacija u cilju rješavanja novih problema koji se ne mogu riješiti automatski"*
- obradama grafičkih prikaza (engl. visual processing) odnosno „*sposobnosti generiranja, pohrane, dohvaćanja te izmjene grafičkih prikaza i doživljaja*"
- kratkoročnom memorijom (engl. short-term memory) odnosno „*sposobnosti razumijevanja i zadržavanja svjesnosti o ograničenom broju novih informacija koje se pojavljuju u neposrednim situacijama odnosno situacijama koje su se upravo dogodile*".

Navedeno potvrđuje i istraživanje kriterijske valjanosti testa računalnog razmišljanja (CTt) u odnosu na druge standardizirane psihološke testove (PMA, RP30) koji je primijenjen na uzorku od 1251 učenika petog do desetog razreda u Španjolskoj. Pokazala se statistički srednje značajna povezanost između računalnog razmišljanja i prostorne sposobnosti (r = 0,44), sposobnosti zaključivanja (r = 0,44) te sposobnosti rješavanja problema (r = 0,67) (Romn-Gonzlez i sur., 2017).

U novijem istraživanju koje je imalo za cilj otkriti postoji li povezanost između računalnog razmišljanja i inteligencije rezultati su otkrili snažnu, pozitivnu i linearnu vezu između odgovarajućeg testa računalnog razmišljanja (online test, 20 zadataka Dabar izazova) i testa opće inteligencije (TNI test, treće izdanje) (Boom i sur., 2018). Prema navedenim rezultatima moglo bi se reći da će opća inteligencija biti veća s većom sposobnosti računalnog razmišljanja.

Znači li to da postoji čvrsta veze između računalnog razmišljanja i inteligencije jer same definicije i konstrukti promatranih pojmova nisu jasno razdvojeni? Ili se može reći da su takvi rezultati očekivani jer se iz definicije vidi prirodna veza među pojmovima. Autori istraživanja ističu važnost daljnjeg istraživanja same prirode i definicije računalnog razmišljanja kako bi se u procesu učenja i poučavanja učitelji mogli kvalitetno usredotočiti na prepoznavanje i razvoj onih sposobnosti koje se izravno i jedinstveno odnose na računalno razmišljanje.

Potrebno je analizirati i istražiti kako planirati proces učenja i poučavanja koji će pozitivno utjecati na razvoj vještina računalnog razmišljanja. U procesu obrazovanja inteligencija i učenje često se zajednički promatraju. Inteligencija je za mnoge psihologe sposobnost ili crta ličnosti, a učenje proces. Kaže se da učenje omogućuje razvoj inteligencije, te da inteligencija olakšava učenje (Zarevski, 2012). Među korelacijama koje su potvrđene različitim istraživanjima, ističe se da će pozitivna korelacija između inteligencije i učenja biti snažnija

- što je veća složenost sadržaja koji se uči
- što je veća smislenost sadržaja koji se uči, odnosno lako je moguće uspostaviti kvalitetnu poveznicu s ranije naučenim sadržajem, odnosno postojećim znanjem
- što je vrijeme učenja manje.

Pri planiranju učenja i poučavanja osim navedenog, potrebno je uvažiti i razlike u brzini procesiranja informacija koje postoje između djece i odraslih. Djeca sporije obrađuju podatke od odraslih. Jedan od razloga može biti da im je potrebno više vremena za izvođenje osnovnih mentalnih operacija. Nadalje, mogu biti manje učinkoviti pri organiziranju izvršavanja nekog zadatka, mogu imati slabiju bazu dugoročnog pamćenja te brže gubiti informacije. S razvojem djeca postižu kompletniji nadzor nad vlastitim mišljenjem i ponašanjem, postaju sve temeljitiji u obradi informacija, kodiraju više informacija iz problema od mlađe djece (Sternberg, 2004). Značaj maturacije u razvoju misaonih procesa posebno je opisao Piaget koji smatra da se njihov razvoj događa u fazama:

- senzomotoričkoj (do 2.god) – nadograđivanjem na refleksne akcije i ponašanjem usmjerenim na ponavljanje ili zadržavanje zanimljivih podražaja
- predoperacionalnoj (2.god. - 8.god.) – namjerno eksperimentiranje na fizičkim objektima, uzimanje u obzir više od jedne karakteristike odjednom
- konkretnih operacija (8.god. – 12.god.) – sve sofisticiranije mentalne manipulacije unutarnjih reprezentacija konkretnih objekata
- formalnih apstrakcija (12.god – 16.god.) – apstraktno mišljenje i logičko rezoniranje.

Alternativna stajališta o sazrijevanju misaonih procesa uvelike prihvaćaju Piagetovu teoriju (tzv. neopiagetovski teoretičari). Neki teoretičari uvode petu fazu kognitivnog razvoja koju nazivaju faza pronalaženje problema (Arlin, 1975) pri čemu se misli na otkrivanje problema s kojim se učenik konkretno susreće te odlučivanje o najvažnijim problemima koji su vrijedni ulaganja dodatnog napora. Na primjer, u toj fazi učenik se smatra sposobnim odabrati temu za seminarski rad (pronalaženje problema). Neki od njih bit će sposobni napisati taj seminarski rad (rješavanje problema). Nekolicina psihologa zagovara mišljenje da nakon faze formalnih

operacija dolazi do faze postformalnog mišljenja. Takvo mišljenje omogućava odraslima manipuliranje ćudljivim postupcima te nedosljednostima u svakodnevnim situacijama, ali i razmatranje alternativa te analiziranje prednosti i nedostataka koje one donose.

Nešto drugačiji pristup razvoju misaonih procesa zagovara Vigotsky (1978) koji smatra da se kognitivni razvoj u velikoj mjeri događa **izvana prema unutra** putem apsorpcije znanja iz konteksta (Vygotsky, 1978). Ističe da se djetetovo učenje u većem dijelu odvija kroz interakciju djeteta s okolinom. U obrazovnim procesima posebno ističe potrebu za stvaranjem dinamičkog okružja za vrednovanje u kojem se događa stalna interakcija djeteta i ispitivača (učenika i učitelja), čak i kada je učenik odgovorio pogrešno. Na taj način osigurava se učinkovitije napredovanje učenja jer se učeniku nudi pomoć za stvaranje neophodnih preduvjeta za napredovanje iz trenutne razine razumijevanja.

Takav se pristup danas može primijetiti kroz značajno isticanje važnosti formativnog vrednovanja u procesu učenja i poučavanja, dakle u otkrivanje onoga što učenici znaju dok su još u procesu učenja. Dizajniranje kvalitetnih alata formativnog vrednovanja može biti izuzetno značajno za učitelje, ali i za učenike (Thomas, 2019), jer se pritom mogu dobiti odgovori na pitanja kao što su:

- Je li učenik spreman za nove sadržaje?
- Treba li učeniku/učenicima drugačiji put pri usvajanju novih koncepata?
- Koji su učenici spremni krenuti dalje, a kojima treba drugačiji put?

Mnogi učitelji i stručnjaci vjeruju da je formativno vrednovanje sastavni dio učinkovitog poučavanja. Primijenjene tehnike mogu biti jednostavne poput traženja od učenika da podignu ruku ako smatraju da su razumjeli novouvedeni koncept ili mogu biti sofisticirana, npr. proces pri kojem učenici, primjenom odgovarajuće rubrike, samovrednuju vlastiti uradak, a učitelj to pregledava i komentira. Formativna vrednovanja služe učiteljima kako bi prepoznali potrebe i probleme u učenju, no u mnogim slučajevima takva vrednovanja pomažu i učenicima u razvoju jačeg razumijevanja vlastitih akademskih prednosti i nedostataka. Ako učenici znaju što dobro rade i na čemu trebaju više raditi, to im može pomoći pri preuzimanju veće odgovornosti nad vlastitim učenjem i akademskim napretkom (Formative Assessment, 2014).

Iz navedenog može se zaključiti da je pri procesu planiranja vrednovanja nužno uspostaviti odgovarajuću ravnotežu između stvarnih sposobnosti i mogućnosti učenika te očekivanih ishoda vrednovanja. U ovom radu naglasak je upravo na vrednovanju koncepata računalnog razmišljanja i to apstrakcije, algoritamskog razmišljanja i dekompozicije. Navedeni koncepti

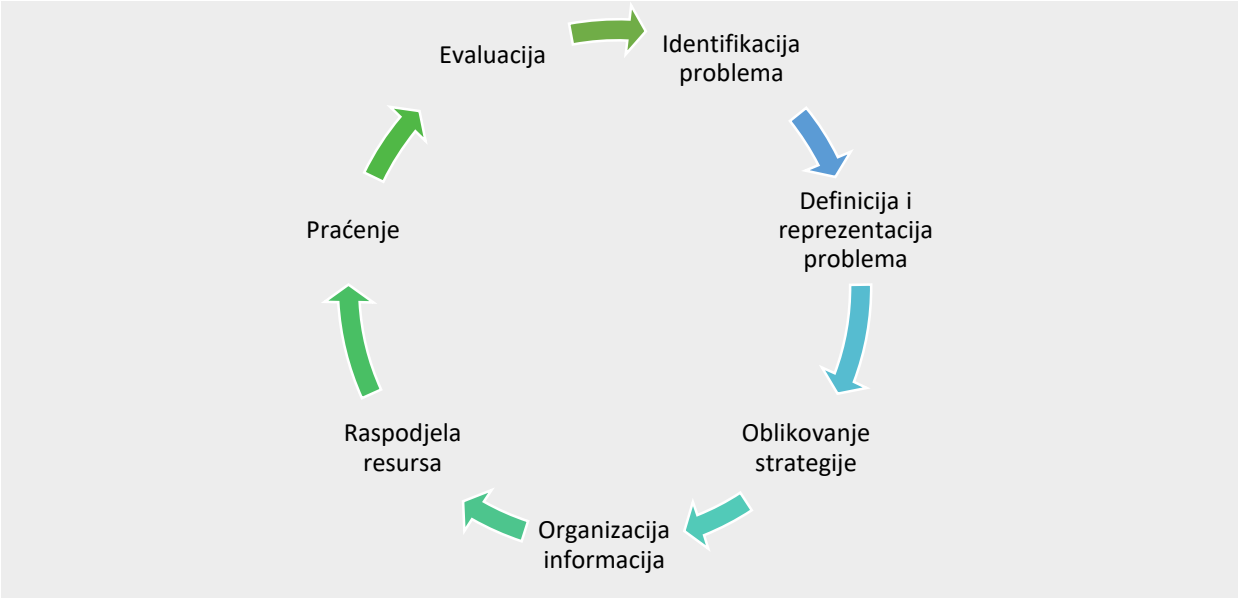
odabrani su u skladu s trenutno važećim Nastavnim planom i programom izbornog predmeta Informatika za šesti razred osnovne škole. Možemo ih ubrojiti u teško mjerljive konstrukte jer se pri njihovom vrednovanju stavlja naglasak na proces promišljanja, način rješavanja problema te pronalazak odgovarajućeg dokaza takvog rješenja.

Način na koji ljudi rješavaju probleme te kako pritom primjenjuju stručnost od posebnog je interesa u području učenja i poučavanja. Danas se promišljanja o načinu rješavanja problema te teorije ljudske spoznaje primjenjuju na strojeve pomoću kojih se to isto promišljanje provjerava. Tehnike rješavanja problema od posebne su važnosti bile i za razvoj umjetne inteligencije. Upravo su istraživanja koja se odnose na razvoj umjetne inteligencije, razvila modele koja se mogu primijeniti pri rješavanju zadanih problema.

2.3. Sposobnost rješavanja problema

Opće prihvaćena definicija računalnog razmišljanja u prvi plan stavlja sposobnost rješavanja problema. Termin rješavanja problema često je prisutan u našoj komunikaciji te se uglavnom intuitivno razumije što se pritom želi postići. Pri rješavanju problema u svakodnevnim situacijama najčešće se koristimo nekom vještinom ili znanjem o tome kako se nešto radi, ali i kreativnim uvidom. Pri tome svladavamo prepreke koje nam ometaju postizanje cilja ili odgovaraju na neko pitanje (Sternberg, 2004).

Kako započeti proces rješavanja nekog problema? Optimalni ciklus rješavanja problem uključuje sljedeće korake: identifikaciju problema, definiciju te prepoznavanje problema, oblikovanje strategije, organizaciju informacija, raspodjelu resursa, praćenje te evaluaciju, no gotovo je nemoguće isključiti fleksibilnost te prilagodbu svakom pojedinom problemu (Slika 4). Rijetko se događa da se pri rješavanju nekog problema slijepo držimo navedenih koraka. Svaki od navedenih koraka ima svoje specifičnosti, zahtjeve te važnost za uspješno postizanje zadanog cilja. Definicija i reprezentacija problema izuzetno su važne i smatraju se ključnim za proces rješavanja problema dok je identifikacija problema jedan od najzahtjevnijih koraka.



Slika 4: Koraci rješavanja problema (Bransford i Stein, 1993)

S druge strane, kvalitetno praćenje i promišljeno trošenje vremena može značajno utjecati na učinkovitost postupka rješavanja problema. U svakodnevnom nastavnom radu, tj. procesu učenja i poučavanja često se spominje sposobnost rješavanja problema kao jedan od obrazovnih

ishoda koji se mora realizirati nastavnim procesom. Kakve to probleme postavljamo pred učenike? Koje strategije učenici primjenjuju za rješavanje problema? U nastavi se najčešće koriste dobro strukturirani problemi iz specifičnih područja (npr. matematike, informatike, fizike), odnosno problemi u kojima je jasno naznačen put do samog rješenja. Primjer takvog problema iz područja programiranja prikazan je na sljedećoj slici (Slika 5). U zadatku je jasno definiran problemski prostor pa pri rješavanju problema neće biti poteškoća s konstruiranjem prikladnih mentalnih prezentacija, odnosno osmišljavanju plana za uzastopno praćenje niza koraka kojima će se približiti rješenju.

4. Ivan je napisao računalni program kojim je želio izračunati i ispisati opseg i površinu kvadrata koji ima duljinu stranice 23. Ivan je siguran da je napisao sve potrebne naredbe, ali nije siguran da ih je složio na odgovarajući način. Analiziraj njegovo rješenje i napiši ispravan redoslijed naredbi.

Ivanovo računalno rješenje	Tvoj prijedlog računalnog rješenja
o=4*a a=23 print('površina je ',p) p=a*a print('opseg je ',o)	

Slika 5: Primjer dobro strukturiranog problema u programiranju

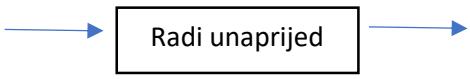
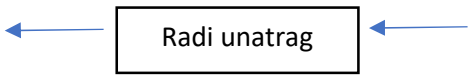
Posebno zanimljiva vrsta problema su tzv. problemi poteza kod kojih pronalaženje rješenja zahtijeva niz poteza. Danas su česte računalne simulacije u kojima je zadatak istraživača izraditi računalni program koji može riješiti zadani problem. Na tom principu, Newell i Simon razvili su model rješavanja problema koji promatra početno (problem) i ciljno (rješenje) stanje unutar zadanog problemskog prostora, a temeljna strategija rješavanja predstavlja razdvajanje zadatka na niz koraka koji dovode do rješenja problema (Newell i Simon, 1972).

Računalnim simulacijama kreiraju se algoritmi, sljedovi operacija koji se mogu opetovano koristiti (Hunt, 1975), koji se uzastopno ponavljaju sve dok se ne zadovolji uvjet, odnosno riješi zadani problem (Sternberg, 2000). Kada čovjek rješava neki problem, njegov se pristup ipak značajno razlikuje od pristupa u računalnim simulacijama zbog ograničenja našeg radnog pamćenja. Čovjek nije u mogućnosti specijalizirati se za vrlo brze izračune većeg broja mogućih kombinacija. Prema Newell i Simon ljudi koriste mentalne kratice za rješavanje problema tzv.

Heuristike neformalne, intuitivne spekulativne strategije koje ponekad dovode do učinkovitog rješenja (Sternberg, 2006).

U situacijama kada nismo u mogućnosti istražiti sve kombinacije mogućih rješenja, ljudski um nastoji pojednostavniti pristup te pribjegava jednoj od sljedećih heuristika: analiza sredstvo-cilj, rad unaprijed, rad unatrag te smisli i provjeri (Tablica 1). Razlike među navedenim pristupima jasno se prepoznaju u početnom koraku pa neke heuristike započinju procjenom krajnjeg cilja, druge analizom problema od početka ili od kraja problema, a neke heuristike stavljaju naglasak na traženje alternative. Nakon definiranja početnog koraka postupak rješavanja može ići u smjeru smanjivanja broja koraka do rješenja, nadalje prema rješenju može se postupno kretati naprijed ili natrag ovisno o početnom koraku tražeći moguće načine postizanja rješenja. Alternativni pristup rješavanja problema ne prati neki sustavni način rješavanja, već za svaku odabranu aktivnost provjerava koliko se uspješno aktivnost može iskoristiti za pronalazak rješenja, na kraju se bira najizravnije rješenje.

Tablica 1: Rješavanje problema: heuristike koje se mogu pri tome primijeniti (Sternberg, 2004)

Heuristika	Definicija	Primjer primjene na svakodnevnom problemu
Analiza sredstvo-cilj	Rješavač analizira problem tako da vidi kraj – cilj koji treba postići – i zatim pokušava smanjiti udaljenost između trenutačnog položaja u problemskom prostoru i krajnjeg cilja u tom prostoru.	Pokušati smanjiti udaljenost između kuće i odredišta
Rad unaprijed	Rješavač kreće od početka i nastoji riješiti problem od početka do kraja 	Analizirati sve zračne putanje koje vode od kuće do odredišta te odabrati one koje izgledaju kao najizravniji put među njima
Rad unatrag	Rješavač počinje od kraja i pokušava riješiti problem radeći unatrag 	Pronaći sve moguće zračne putanje od kuće do odredišta i raditi unatrag kako bi se pronašlo koje od njih su najizravniji put
Smisli i provjeri	Rješavač jednostavno smišlja alternativne načine rješavanja, ne nužno na sustavan način i zatim za svaki tijek aktivnosti provjerava hoće li funkcionirati	Istražiti alternativne putanje, provjeriti koje od njih se mogu iskoristiti te izabrati najizravniju putanju

Za razliku od dobro strukturiranih problema postoje i loše strukturirani problemi kod kojih ne postoji jasan, odmah dostupan put do rješenja. Takvi problemi zahtijevaju jedan postupno drugačiji način sagledavanja problema tzv. pristupom uvida. Takvim pristupom problem je potrebno promatrati potpuno različito od onog kako se problem promatra na početku te različito od nekog općenitog pristupa rješavanju problema.

Uvid nudi specifičan pogled, ponekad i iznenadno razumijevanje problema ili strategije koja može biti od pomoći pri rješavanju problema, a uvijek je rezultat duljeg promišljanja i teškog rada. Iako se način rješavanja problema pristupom uvida još uvijek ozbiljno istražuje, može se reći da poznavanje rješenja nekih izomorfnih problema može biti od koristi. Prema Davidson i Sternberg postoje tri različita uvida (Davidson i Sternberg, 2003):

- uvid putem selektivnog kodiranja – uključuje razlikovanje bitnih od nebitnih informacija
- uvid putem selektivnog uspoređivanja – uključuje nove percepcije načina na koje se nove informacije mogu povezati sa starima
- uvid putem selektivnog kombiniranja – uključuje odabir selektivno kodiranih i uspoređenih djelića relevantnih informacija i kombiniranje tih informacija na nov produktivan način.

2.3.1. Poteškoće pri rješavanju problema

Često se problemi, osobito oni koji zahtijevaju uvid smatraju jako teškim jer zahtijevaju uvođenje novog mentalnog modela u postojeći model prezentacije problema, konteksta ili postupka za njegovo rješavanje. Postojeće mentalne konstrukcije potrebno je kvalitetno i održivo povezati s novim konstrukcijama.

U nastavnom procesu često se bavimo određenim problemima upravo zbog očekivanja da će proučavanje i rješavanje odabranih problema potaknuti stvaranje pozitivnog transfera, odnosno prijenosa znanja ili vještine iz jedne problemske situacije u drugu (Detterman i Sternberg, 1993). Često se pri poučavanju programiranja čuje izreka da nije osnovni cilj da svi učenici postanu programeri, već se želi kod učenika potaknuti razvoj apstrakcije te algoritamskog načina razmišljanja, osnažiti rad s apstraktnim modelima, razvijati sposobnost dekompozicije, odnosno ukratko računalnog razmišljanja jer se takav način razmišljanja pokazao od iznimne koristi i u mnogim drugim znanstvenim disciplinama (Interdisciplinary Computatonal Thinking, 2017). Takva očekivanja upravo su rezultat pozitivnog transfera kojeg očekujemo da se dogodi kao posljedica procesa poučavanja. Nadamo se da će učenici koji su uspješno

primjenjivali dekompoziciju u algoritamskim zadacima uspješno istu vještinu primijeniti i izvan informatičkog područja. U procesu poučavanja uglavnom ne želimo pojavu negativnog transfera, odnosno situaciju kada je rješavanje zadanog problema otežano poznavanjem nekog ranijeg problema.

2.3.2. Stručnost i rješavanje problema

U području učenja i poučavanja od iznimne je važnosti istražiti ulogu stručnosti na sposobnost pojedinca pri rješavanju problema. Istraživanja poučavanja procesa rješavanja problema programiranjem potvrdila su da stručnjaci uspješnije rješavaju probleme od početnika (Bubica i Boljat, 2014) te stvaraju bolje kvalitetnije, održive mentalne modele važnih koncepata (Bubica i Boljat, 2014; Pillay i Jugoo, 2005).

Mnogi stručnjaci prihvaćaju stav da je 'vježbanje dobar put prema savršenstvu', no naglašavaju da ono mora biti promišljeno, namjerno i usmjereno te naglašavati stjecanje novih vještina (Sternberg, 1998). Genetskom nasljeđu ipak se priznaje važnost pri stjecanju nekih vrsta stručnosti npr. kod čitanja.

Kakve su razlike među stručnjacima i početnicima vezano uz organizaciju znanja i kreativnost? Količina znanja, njegova organizacija te uporaba tog znanja predstavlja osnovnu razliku među početnicima i stručnjacima nekog područja (Gobet i Simon, 1996). Bez obzira na područje djelovanja, stručnjaci stvaraju sheme za rješavanje problema koje možemo opisati kao velike, međusobno vrlo dobro povezane jedinice znanja, organizirane u skladu s temeljnom strukturnom sličnošću među jedinicama znanja (Glaser i Chi, 1988).

Početnici uglavnom stvaraju manje, nepovezane jedinice znanja koje se povezuju prema površinskim sličnostima (Bryson i sur., 1991). Primjećuju se razlike i u pristupu postavljanja problema te načinu utroška vremena u postupku rješavanja. Stručnjaci troše značajno više vremena na promišljanje o načinu reprezentacije problema (Lesgold, 1988), a mnogo manje od početnika na samu primjenu strategije za rješavanje.

Stručnjaci se dosta bave pitanjima koja se odnose na informacije vezane uz problem te kako ih usporediti s vlastitim shemama. U ovisnosti o dobivenim odgovorima lako i brzo 'u hodu' prizivaju te primjenjuju odgovarajuću strategiju za rješavanje problema. Od zadane informacije idu naprijed prema nepoznatoj informaciji primjenom odgovarajućih niza koraka koji se izvršavaju odabranim redoslijedom. Pritom primjenjuju strategije koje su utemeljene na njihovim shemama u dugoročnom pamćenju (Chi i sur., 1082). Opetovanim vježbanjem stručnjaci mogu automatizirati različite operacije koje lako pronalaze i izvode dok rade

unaprijed. Ipak, automatiziranost stručnjaka može biti otežavajuća okolnost pri rješavanju problema koji se strukturalno razlikuju od onih kakvima se uobičajeno bave (Frensch i Sternberg, 1989).

2.3.3. Važnost razmatranja mentalnih modela kod učenika

Ljudi prezentiraju svijet koji ih okružuje mentalnim modelima koji mogu nastati percepcijom, imaginacijom te konstruktivnim raspravama (Craik, 1943). Mozak stvara „modele stvarnosti malog opsega“ koji mogu poslužiti za predviđanje i razumijevanje događaja te podržavanje objašnjenja tih događaja. Norman (1983) opisuje mentalne modele s tri osnovna uvjeta:

- postoji sustav uvjerenja – mentalni modeli odražavaju uvjerenja osobe koja ih posjeduje o promatranom sustavu
- sustav je promotriv – postoji određena komunikacija između parametara i stanja mentalnog modela te zadanog sustava koju osoba može promatrati
- postoji snaga predviđanja – svrha mentalnog modela je podržavanje ljudi u razumijevanju i predviđanju ponašanja promatranog sustava (Norman, 1983).

Daljnja istraživanja mentalnih modela prepoznala su neke karakteristike mentalnih modela:

- Mentalni su modeli nedovršeni i pojednostavljeni – učenik ograničen svojim prethodnim znanjem ne može stvoriti cjelovit mentalni model koji pokriva sve detalje promatranog sustava.
- Mentalni modeli su promjenjivi – mijenjaju se i razvijaju kako njihovi vlasnici ostvaruju interakcije sa sustavom.
- Postoji određeno vrijeme kašnjenja u mijenjanju mentalnih modela – pri izmjeni modela stara informacija zamjenjuje se novom, ali to se ne događa trenutno. Stara informacija zadržava se još neko vrijeme u memoriji zajedno s novom informacijom.
- Mentalni modeli su oskudni – ljudi teže izbjeći složenost u mentalnim modelima.
- Mentalni modeli su neznanstveni i često sadrže predrasude – ljudi često zadržavaju svoja ponašanja čak i kad znaju da su pogrešna, a uglavnom to rade jer se zbog toga osjećaju ugodno i sigurno.
- Mentalni modeli imaju nejasne granice – slični modeli mogu se miješati i ispreplitati.

Iako postoji mnogo istraživanja koja se bave ljudskim mentalnim modelima prirodnih fenomena koji ih okružuju, postoji jako malo istraživanja koja se bave upravo mentalnim modelima programera početnika (Ma, 2007; Bubica i Boljat, 2015; Bubica i Boljat, 2014).

U nastavi programiranja učitelju je veoma važno odrediti mentalne modele temeljnih programerskih koncepata koje posjeduju njegovi učenici. Analizirajući odgovore koje su ponudili učenici, učitelj može doći i do mogućih uzoraka takvih odgovora te prepoznati mentalne modele odabranih koncepata koje posjeduju učenici. Analizom modela učitelj procjenjuje svoje sljedeće aktivnosti te uočava koje koncepte ili skup koncepata treba ponovo poučavati (Bubica i Boljat, 2015). Zbog snažne uloge mentalnih modela koje učenici razvijaju na njihov uspjeh u programiranju, razvoj valjanih mentalnih modela razmatra se kao jedan od faktora koji značajno može utjecati na postizanje uspjeha u programiranju.

2.3.4. Kreativnost i rješavanje problema

U promišljanjima učitelja i metodičara o učenju i podučavanju često se može čuti da kod učenika treba poticati kreativnost. Većina stručnjaka opisuje kreativnost kao proces proizvodnje nečeg što je istovremeno i originalno i vrijedno (Sternberg i Lubart, 1996) pa ga možemo prepoznati u nekoj teoriji, plesu, kemijskom elementu, glazbenom uratku, priči, postupku ali i kroz stvaranje originalnih poveznica, ideja i pristupa na temelju kojih se kreiraju i primjenjuju znanja u različitim okružjima. Razvojem kreativnog mišljenja te kreativnosti općenito nastoji se omogućiti stjecanje znanja na višim kognitivnim razinama npr. transformacijom i primjenom znanja te stvaralačkim rješavanjem problema (Ministarstvo znanosti i obrazovanja, 2019a; Ministarstvo znanosti i obrazovanja, 2019b, Ministarstvo znanosti i obrazovanja, 2019d).

U procesu stvaranja algoritamskih rješenja kreativnost je jedna od ključnih značajki. Iako pojam algoritma kao konačnog niza uputa koje se izvršavaju prema unaprijed određenom redoslijedu djeluje kao da nema nikakvu poveznicu s kreativnim procesom, stvarnost je upravo suprotna. Koraci koje čine određeni algoritam, rješenje zadanog problema, nisu unaprijed zadani. Postupak pronalaženja odgovarajućih koraka koji će izgraditi neki algoritam nastaju kreativnim promišljanjem o problemu. Za neki problem moguće je kreirati više algoritamskih rješenja koja se razlikuju u različitim obilježjima. Rješenja mogu imati različitu složenost, brzinu izvršavanja, razinu usklađenosti rješenja zadanom problemu i slično.

Kako u procesu podučavanja potaknuti učenike na kreativno stvaranje? Kreativni pojedinci imaju sposobnost proizvoditi više od ostalih (Weisberg, 1999). Oni su voljni raditi dugo i vrijedno, proučavaju radove svojih prethodnika i suvremenika kako bi postali stručnjaci u svom području. Nadograđuju svoje znanje i divergiraju od njega da bi stvorili inovativne pristupe i proizvode. Često posjeduju snažnu intrinzičnu motivaciju, imaju osobnu želju da riješe problem ili jednostavnu uživaju u kreativnom procesu (Amabile, 1996).Prema (Sternberg, 2000)

predlaže se model kategorija kreativnih doprinosa kojima se može unaprijediti pojedino područje tj. ona su rezultat kreativnog vodstva svojih stvaralaca. To su sljedeći kreativni doprinosi:

- replikacija – napor da se pokaže koliko je određeno područje tamo gdje bi trebalo biti
- redefinicija – napor da se redefinira trenutačni položaj područja
- unapređenje – pokušaj da se područje unaprijedi u smjeru u kojem se već kreće
- izrazito unapređenje
- preusmjeravanje – pokušaj da se područje sa sadašnjeg područja pomakne u novom i različitom smjeru
- rekonstrukcija – pokušaj da se područje pomakne na položaj na kojem je bilo ranije tako da se s te točke može krenuti naprijed
- ponovni početak – pokušaj da se područje pomakne na drugu i ranije nedostignutu početnu točku te da se s te točke krene u novom smjeru
- integracija – pokušaj da se područje pomakne tako da se ujedine aspekti dvaju ili više prošlih vrsta kreativnih doprinosa koji su se ranije smatrali različitima ili čak suprotnima, a sada se smatraju sintetiziranima.

Danas se škola, kao važna odgojno-obrazovna ustanova, s razlogom doživljava kao mjesto razvoja kreativnog potencijala kod učenika (Somolanji i Bognar, 2008). Kreativnost se najčešće iskazuje u pojedinim područjima (Sternberg, 1999) pa je važno omogućiti učenicima upoznavanje svoje kreativnosti djelovanjem u različitim nastavnim predmetima. Iz tog razloga nastavni se proces sve više prilagođava individualnim potrebama učenika, njihovom individualnom stilu i dinamici učenja. Ipak još uvijek se nedovoljna pažnja posvećuje konstruktivističkom pristupu podučavanju kojim se želi kreirati stvaralačko okružje za učenika (Pejić i sur., 2007).

Kako potaknuti kreativnost kod učenika? Postoje kreativne tehnike razmišljanja koje se koriste u svrhu poticanja kreativnog potencijala pojedinca ili grupe pojedinaca pri rješavanju problema ili generiranju ideja potrebnih za razvoj novih proizvoda ili korisničkih usluga. Moguće je pronaći bilješke za oko 200 tehnika kreativnog razmišljanja koje su opisane u literaturi, a među njima najviše su istražene sljedeće tehnike: metoda 635 (engl. Brainwriting), oluja mozgova (engl. Brainstorming) i sinektika (engl. Synectics).

Metoda 635 tehnika je kreativnog mišljenja koja pojedincu dopušta iznošenje više ideja istovremeno. Sudionici primaju informacije paralelno, čime se značajno povećava broj predloženih ideja u istom vremenu.

Osnovna obilježja ove tehnike su (Michalko, 2010):

- jasno definiranje problema
- sudionici svoje ideje zapisuju na kartice (svaku ideju na zasebnu karticu) i prosljeđuje je drugima na čitanje
- svaki sudionik čita ideju s kartice, promišlja i eventualno zapisuje novu ideju na karticu, postupak se može ponavljati više puta
- nakon pola sata stvaranja ideja kartice se slažu na ploču, duplikati se uklanjaju, a ideje se mogu organizirati i po nekim kategorijama
- sudionici dogovorenim oznakama obilježavaju ideje koje im se sviđaju.

Ova tehnika može se uspješno primijeniti i na individualno rješavanje problema pri čemu se koriste pitanja koja potiču na promišljanje i traženje odgovora: „*Koja je opcija prihvatljiva? Što je najbolje za mene? Što trebam učiniti?"*. Odgovore je potrebno zapisati bez analiziranja. Nakon zapisivanja svih odgovora te njihovog čitanja pretpostavlja se da će se rješenje problema prepoznati u nekom od odgovora ili će se pojaviti neka nova ideja kao rješenje.

Oluja mozgova (engl. Brainstorming) je tehnika stvaranja slobodnog okružja u kojem se sudionike stimulira na stvaranje kreativnih ideja i zamisli. Unutar manje grupe, sudionici razgovaraju o problemu, jedan po jedan predlažu svoje ideje i sugestije.

Osnovno obilježje ove tehnike je da se sudionici suzdržavaju od kritiziranja sugestija i ideja prilikom njihovog izlaganja te da se očekuje da će veći broj predstavljenih ideja rezultirati kvalitetnim rješenjem problema.

Sinektika je tehnika kreativnog mišljenja kod koje se stimuliranje misaonih procesa događa kroz primjenu metafora i poredbi. Osnovna obilježja ove tehnike su:

- kreativno razmišljanje može se opisati i naučiti: ljudi mogu biti kreativniji ako razumiju kako kreativnost nastaje
- inovacije u znanosti i umjetnosti su slične i kontrolirane istim psihičkim procesima: važan element kreativnosti je prihvaćanje naizgled nevažnog, stavljanjem većeg naglaska na osjećaje umjesto razum
- individualna i grupna kreativnost su slične.

Tehnika je razvila konkretne vježbe i smjernice vođenja sastanka koje pomažu pri formiranju konstruktivnih ideja za rješavanje problema. Osmišljena je kako bismo krenuli u smjeru nelogičnosti čime dobivamo priliku vidjeti stvari na novi, kreativni način, izraziti sebe i pristupiti problemu na nekonvencionalan način (Nolan, 2010).

Iz razmatranja o inteligenciji, učenju i sposobnosti rješavanja problema može se utvrditi da je moguće unaprijediti sposobnost rješavanja problema osobito većom razinom stručnosti (korištenje postojećih znanja), primjenom kreativnog uvida te tehnikama za poticanje kreativnosti. Kvalitetnim vrednovanjima potrebno je analizirati postojeća znanja učenika kako bi se prepoznali mentalni modeli koje posjeduju te otkrili problemi u njihovom razumijevanju.

2.4. Faktori koji značajno utječu na uspjeh u rješavanju problema programiranjem (prediktori)

Digitalni uređaji dio su naše svakodnevnice i teško je pronaći područje u kojem njihova primjena ne donosi neke prednosti. Svakodnevno se prezentiraju računalne i mobilne aplikacije koje nam mogu značajno olakšati obavljanje svakodnevnih poslovnih ili osobnih aktivnosti. Usprkos popularnosti računalnih i mobilnih aplikacija te činjenici da su sadržaji programiranja danas sastavni dio informatičkih kurikula u osnovnim i srednjim školama, ali i u raznim klubovima koji populariziraju informatičke sadržaje, osobito programiranje, učenicima programerski sadržaji nisu posebno atraktivni i često ih smatraju zahtjevnim i neprivlačnim.

Zašto se rješavanje problema programiranjem smatra tako zahtjevnim sadržajem za učenje i poučavanje? Koji su to faktori koji mogu značajno unaprijediti učenje programiranja? Postoji veći broj istraživanja koja su pokušala odgovoriti na ova pitanja. Kroz rezultate takvih istraživanja prepoznati su faktori koji su pokazali značajnu snagu predviđanja uspjeha u programiranju. Takve faktore smatramo prediktorima uspjeha u programiranju jer su pokazali snažan, pozitivan ili negativan utjecaj na izvedbu programiranja.

Faktori koji su pokazali pozitivan utjecaj na izvedbu programiranja su: matematičko predznanje (Wilson i Shrock, 2001; White i Sivitanides, 2003), skiciranje prostornih mapa (Fincher, i sur., 2006; Kranch, 2011; Watson i Godwin, 2014), atribucija uspjeha (Wilson i Shrock, 2001), stilovi učenja prema Gregorcu (Watson i Godwin, 2014), prvi programerski jezik (Pillay i Jugoo, 2005), vještine prostorne vizualizacije (Fincher, i sur., 2006), visok postotak uzastopnih uspješnih prevođenja programa (Watson i Godwin, 2014), strategija dubinskog pristupa učenju (Fincher, i sur., 2006), uporaba održivih mentalnih modela (Bornat i sur., 2008; Kranch, 2011; Kranch, 2012), sposobnost rješavanja problema (Pillay i Jugoo, 2005), vježbanje jezičnih vještina (Siegmund, i sur., 2014).

Faktori kao što su visok postotak uzastopnih pogrešaka u vježbama programiranja (Watson i Godwin, 2014), veći broj sati utrošen na ispravljanje pogrešaka (Watson i Godwin, 2014), strategija površinskog pristupa učenju (Fincher, i sur., 2006) te velik broj sati utrošen na igranje računalnih igara (Kranch, 2011) pokazali su negativan utjecaj na vještine rješavanja problema programiranjem.

Istraživanja su prepoznala i faktore koji unatoč očekivanjima nisu izrazili snagu programerskog prediktora. U takve faktore ubrajamo stilove učenja prema Kolbu (Pillay i Jugoo, 2005), spol (Pillay i Jugoo, 2005; Byrne i Lyons, 2001; Chmura, 1998), prethodno programersko iskustvo

(Pillay i Jugoo, 2005; Allert, 2004; Byrne i Lyons, 2001), broj godina programiranja (Watson i Godwin, 2014) i broj upoznatih programerskih jezika (Watson i Godwin, 2014).

Jedan od razloga zbog kojeg još uvijek nema snažnijih rezultata na području predviđanja uspjeha u programiranju može se potražiti u činjenici da predmet Informatika te sadržaji programiranja nisu široko rasprostranjeni u K12 obrazovanju, već se uglavnom poučavaju na sveučilištima te osnovnim i srednjim školama nekih država. Istovremeno, svakodnevni pregled natječaja za posao u mnogim zemljama potvrđuje kontinuiranu potrebu industrije za računalnim stručnjacima, osobito programerima. Računalna industrija je ta koja sve više naglašava važnost prepoznavanja prediktora programiranja kako bi se što prije posvetili onim studentima koji imaju neku predispoziciju za postizanje uspjeha u ovom području.

Kako opisati uspjeh pri rješavanju problema programiranjem? Koji su to faktori zbog kojih neki učenici lako usvajaju koncepte programiranja, dok su drugima ti isti koncepti iznimno zahtjevni? U većini istraživanja uspjeh pri rješavanju problema programiranjem analiziran je kroz uvodne kolegije programiranja koji se provode pod različitim nazivima, s različitim obimom sadržaja koji pokriva zadanu temu, ali i s različitom težinom sadržaja koji se implementira. Zbog ovih razlika teško je govoriti o usklađenosti svakog uvodnog kolegija te usklađenosti rezultata istraživanja. U većini istraživanja uspjeh u uvodnom kolegiju programiranja promatrao se kroz različite kvantitativne mjere poput stope odustajanja od kolegija, trajanje (duljina) kolegija, te izvedbe učenika na ispitima tijekom kolegija i završnom ispitu.

Također, učeničko postignuće negdje se promatralo iz perspektive učitelja istraživanjem razumijevanja koje učenici pokazuju vezano uz sadržaje koji se podučavaju, istraživanjem poznavanja određenih kvalitativnih karakteristika učenika, nadalje razumijevanjem povezanosti uspjeha u programiranju s prethodnim programerskim iskustvima učenika. Sve navedene aktivnosti imale su zajednički cilj, razumijevanje uloge učitelja u razvoju učenika (Kinnunen i sur., 2007).

2.4.1. Faktori koji mogu pozitivno utjecati na postizanje uspjeha pri rješavanju problema programiranjem

Utjecaj općih faktora

Broj istraživanja koja su proučavala faktore značajne za postizanje uspjeha je velik. Najbolji pokazatelji uspjeha u svim disciplinama smatraju se samouvjetovani uspjeh, osobni stav, oduševljenje određenim sadržajem te opća akademska motivacija. Ovi faktori stvarno imaju snažan utjecaj na uspješnost učenika. Za prepoznavanje specifičnih pokazatelja uspješnosti pri rješavanju problema programiranjem, nužno je bilo provesti osjetljivija istraživanja u području programerske izvedbe. Razni pokušaji predviđanja uspjeha u programiranju obično su započinjali intuitivnim pretpostavkama samih istraživača.

Poznavanje područja iz kojeg dolazi zadatak programiranja nametnuo se kao jedan od prvih dobrih prediktora (Adelson i Soloway, 1985). Ipak, daljnja istraživanja procesa dizajniranja računalnog rješenja pokazala su da bi upravo ne preveliko poznavanje određenog područja moglo olakšati programiranje specifičnih rješenja jer se omogućuje kritički osvrt na problem (Berry, 1995). Ako je pojedinac dovoljno izvan problema te može uočiti svu njegovu specifičnost, može napraviti i bolje računalno rješenje. Danas se ovaj pristup često koristi pri planiranju zadataka programerskih timova (Mehrotra, 2011).

U ranijim istraživanjima prediktora programiranja najčešće su razmatrani sljedeći faktori: stručnost u govornom jeziku, broj korištenih i analiziranih programskih jezika, matematičke sposobnosti, glazbene sposobnosti, akademski stupanj, opća inteligencija, samopouzdanje i slično. Među svim navedenima faktorima kao najmoćniji prediktori uspjeha pri rješavanju problema programiranjem istaknuli su se stupanj ugode, matematička sposobnost te atribucija uspjeha ovisna o sreći (Cantwell Wilson i Shrock, 2001).

Poznato je da postoji snažna korelacija uspjeha pri rješavanju problema programiranjem sa sposobnošću rješavanja problema u drugim znanostima kao što su matematika i fizika (Pillay i Jugoo, 2005). Izvrsno matematičko znanje potvrdilo se kao snažan prediktor uspjeha u programiranju u nekoliko istraživanja (Byrne i Lyons, 2001; White i Sivitanides, 2003; Bennedsen i Caspersen, 2005). Detaljnija istraživanja matematičkih sposobnosti kao prediktora uspjeha u programiranju istaknula su da dobro usvajanje diskretne matematike te matematičkog računa može pozitivno utjecati na uspjeh u programiranju (Pioro, 2006).

Utjecaj pristupa i stilova učenja

Kako programeri početnici usvajaju teške koncepte na uvodnim kolegijima programiranja? Istraživanja koja su se bavila proučavanjem iskustava programera početnika ističu nekoliko zahtjevnih tema i uobičajenih zabluda na koje treba osobito paziti u radu s početnicima (Lahtinen i sur., 2005). Pokazalo se da upravo inicijalizacija varijabli te rad s petljama, uvjetima, pokazivačima i rekurzijom izazivaju najviše zabludi kod programera početnika. Nadalje, početnici često programiranju pristupaju 'redak po redak' umjesto da koriste smislene programske strukture. Imaju poteškoće u različitim pitanjima povezanim s izgradnjom algoritma odnosno računalnog rješenja jer, iako poznaju sintaksu i semantiku pojedinih naredbi, ne znaju ih kombinirati u valjane algoritme odnosno računalna rješenja.

Koje strategije početnici primjenjuju za rješavanje problema koji se temelje na teškim konceptima? Zanimljivi su rezultati istraživanja koja ističu da kvalitetno komentiranje programskih blokova (blokova naredbi) kao i sposobnost skiciranja detaljnih prostornih karata ima značajnu pozitivnu korelaciju s postizanjem uspjeha u programiranja. Studenti koji su za zadanu udaljenost (od adrese A do adrese B) bili sposobni stvoriti prostornu mapu koja sadrži sve poznate objekte te bogate i precizne upute postizali su bolje rezultate u programiranju od onih koji su mogli nacrtati samo smjer kretanja ili kreirati jednostavnu, ne pretjerano detaljnu prostornu mapu (Simon i sur., 2006; Kranch, 2011; Watson i Godwin, 2014). Očigledno različite navigacijske strategije mogu pozitivno utjecati na kvalitetu programskog koda i tako stvoriti kod programera konceptualizaciju prethodnog računalnog iskustva.

Ranija fenomenografska istraživanja sedamdesetih godina istaknula su dva različita pristupa koje primjenjuju početnici: dubinski pristup učenju kod kojeg učenici teže razvoju stvarnog razumijevanja onoga što uče i površinski pristup učenju u kojem studenti samo žele odraditi zadatak koji su dobili od svojih učitelja. Istraživanja koja su proučavala strategije učenja pokazala su da je dubinski pristup učenju pozitivno povezan sa završnom ocjenom kolegija uvodnog programiranja dok površinski pristup učenju sugerira negativnu korelaciju s istim ocjenom (Simon i sur., 2006).

Studenti uvodnih kolegija programiranja često dolaze s različitim računalnim znanjem i iskustvom. Čini se prirodnim očekivati da će studenti s manje prethodnog iskustva u radu na računalu imati manje samopouzdanja u prvim programerskim zadacima, a u skladu s tim i skromnije očekivanje uspjeha. S druge strane, iako je broj istraživanja koji se bavi utjecajem stilova učenja prema Gregorcu na programiranje manji, rezultati su pokazali snažan pozitivan

utjecaj na uspjeh u programiranju za apstraktno/slučajnu dimenziju što ukazuje na činjenicu da bi stilovi učenja prema Gregorcu mogli biti dobri prediktor uspjeha pri rješavanju problema programiranjem (Watson i Godwin, 2014). Valjan zaključak vezano uz utjecaj stilova učenja na izvedbu programiranja mogla bi ponuditi meta-analiza postojećih istraživanja.

Utjecaj faktora povezanih s programiranjem

Usvajanje programerskih vještina zahtijeva od učenika osmišljavanje učinkovitog modela računala, konstrukciju potpuno novog kognitivnog modela kroz kojeg učenik objašnjava kako radi računalo, predviđa i opisuje što se događa prilikom izvođenja određenog računalnog programa (Ben-Ari, 1998). Učenje programiranja uključuje stvaranje održivih mentalnih modela osnovnih programskih koncepata. Pri poučavanju programiranja učiteljima je vrlo važna mogućnost prepoznavanja mentalnih modela osnovnih koncepata programiranja koje učenici posjeduju. Učenici koji stvaraju održive (valjane, nadogradive) mentalne modele vjerojatno će postići znatno bolje rezultate u rješavanju programskih zadataka od onih s mentalno neodrživim (nevaljanim, nenadogradivim) modelima (Ma, 2007; Ma i sur., 2009; Bornat i Simon, 2008).

Jedno od zanimljivih istraživačkih pitanja računalne znanosti zasigurno je izbor prvog programskog jezika. Trenutno, postoji približno 9000 poznatih programskih jezika koji su dokumentirani u cijelom svijetu, dok ih je oko 50 koji se mogu smatrati popularnim i koji se najčešće upotrebljavaju (Fowler, 2020). Koji bi od ovih jezika trebao na najbolji način upoznati učenike s odgovarajućim programerskim konceptima, ali i zadržati interes učenika prema programiranju kao važnom dijelu informatike? Je li izbor prvog programskog jezika jedan od faktora koji bi to mogao značajno odrediti uspjeh učenika u programiranju? Istraživanje uspjeha programera početnika ukazuje da prvi programski jezik snažno utječe na uspjeh u programiranju (Pillay i Jugoo, 2005). Unatoč velikom broju postojećih programskih jezika, neki od njih su jako popularni dok drugi jednostavno nestaju mnogo prije nego što bi mogli postati poznati. Analiza programskih jezika koji se najčešće upotrebljavaju pokazala je da u stvarnoj uporabi postoji mali broj programskih jezika. Nadalje, na odabir prvog programskog jezika najčešće utječu sljedeći faktori: postojanje biblioteka otvorenog koda, postojanje gotovog koda za učenje, prethodno iskustvo u programiranju u odabranom jeziku pri čemu pristupačnost semantike, pouzdanost i značajke samog programskog jezika postaju manje važne (Rabkin, 2013). Zbog snažnog utjecaja prvog programskog jezika na uspjeh početnika učitelji bi trebali ovom pitanju posvetiti posebnu pažnju.

2.4.2. Faktori koji se nisu istaknuli kao faktori koji utječu na postizanje uspjeha pri rješavanju problema programiranjem

Pored faktora koji su pokazali pozitivan utjecaj na postizanje uspjeha pri rješavanju problema programiranjem važno je spomenuti i rezultate istraživanja koja su prepoznala faktore koji nemaju takav utjecaj ili njihov utjecaj još uvijek nije dovoljno istražen.

Istraživanja koju su se fokusirala na stilove učenja učenika, na primjer Colbove stilove učenja i stilove učenja prema Gregorcu, nisu ponudila jedinstvene rezultate. Prema Colbu, različiti su stilovi učenja prikladni za pojedina okružja učenja pa tako i za učenje programiranja pri kojem učenik pokušava riješiti problem, primjenjuje različite vještine rješavanja problema te razumije i prepoznaje odnose između promatranih koncepata. Istraživanje utjecaja Colbovih stilova učenja nije dalo čvrst zaključak o njihovom utjecaju na uspjeh u programiranju, ali je pokazalo da su studenti koji su odabrali kolegij programiranja većinom razvili Colbov konvergentni stil učenja i postizati bolje rezultate u usporedbi s drugim studentima (Byrne i Lyons, 2001).

Iako se pokazalo da ranije iskustvo rada na računalu nije značajno za postizanje uspjeha u programiranju važno je istaknuti da pozitivno prethodno iskustvo programiranja osnažuje pozitivnu samoprocjenu studentskog rada (Kinnunen i Simon, 2014) koja se općenito smatra pozitivnim faktorom uspjeha u bilo kojoj aktivnosti.

Za razliku od izbora prvog programskog jezika, ostali faktori povezani s programskim jezicima kao što su: broj uvedenih (korištenih) programskih jezika (Watson i Godwin, 2014), ukupan broj godina učenja programiranja (Watson i Godwin, 2014) te prethodna iskustva programiranja (Byrne i Lyons, 2001; Allert, 2004; Pillay i Jugoo, 2005) do sada nisu pokazali značajan utjecaj na postizanje uspjeha pri rješavanju problema programiranjem.

Iako se čini iznenađujućim da je nekoliko istraživanja potvrdilo da prethodno iskustvo u programiranju nije važan faktor za postizanje uspjeha, ne može se reći je programsko iskustva potpuno nebitno. Istraživanje utjecaja prethodnog programskog iskustva na uspjeh u programiranju pokazalo je da su studenti s prethodnim iskustvom ipak postigli nešto bolji uspjeh od onih bez prethodnog iskustva (Porter i sur., 2014). Pokazalo se da čak i nakon negativnog iskustva programiranja student može imati pozitivan osjećaj o svojoj učinkovitosti (Kinnunen i Simon, 2014). Ipak, iako se prirodno očekuje da pozitivna iskustva u programiranju vode do pozitivnog osjećaja samoučinkovitosti važno je naglasiti da se uzrok takvog osjećaja može pronaći i u nekim drugim faktorima kao što su izbor prvog programskog jezika,

primijenjene strategije učenja i poučavanja, motivacija učenika ili nekim drugim faktorima povezanima s učiteljima.

Istraživanja koja su proučavala utjecaj spola na kvalitetu programiranja uglavnom su motivirana činjenicom da svijetom programiranja dominiraju osobe muškog spola. Bez obzira na očekivanja da je spol važan faktor, nekoliko je istraživanja potvrdilo upravo suprotno, odnosno da spol nije faktor koji utječe na izvedbu programiranja (Byrne i Lyons, 2001; Allert, 2004; Pillay i Jugoo, 2005; Pioro, 2006). Također, pokazalo se da će žene češće od muškaraca odustati od nekih tehničkih sadržaja računalne znanost poput programiranja (Sheard, i sur., 2008).

2.4.3. Utjecaj učitelja i načina poučavanja na postizanje uspjeha pri rješavanju problema programiranjem

U radu s učenicima koji su prvi put susreću s programiranjem učiteljima je izuzetno važno prepoznati način na koji učenici usvajaju koncepte koji se poučavaju. Iskusni programer rješava problem primjenom strategija izvedenih iz iskustva rješavanja problema iz prošlosti, što ne vrijedi za programere početnike (Raadt i sur., 2004; Kranch, 2012). Početnici uglavnom mogu vrlo dobro razumjeti sintaksna pravila programskog jezika i pisati jednostavne programe, ali često imaju poteškoća pri pronalasku odgovarajućih rješenja problema ako nemaju iskustvo jednog stručnjaka.

Uočavanje razlika u načinu na koje početnici i stručnjaci analiziraju i rješavaju probleme moglo bi značajno pomoći učiteljima. Stručnjaci stvaraju mentalne prikaze, fleksibilne i navigacijske mentalne prezentacije, koje su više od zbroja informacija dobivenih čitanjem programa.

Za razliku od njih, programeri početnici grade znanje stalnim popunjavanjem i povezivanjem dijelova znanja, što je obično dugotrajan proces. Postoji procjena da je za takav proces stvaranja vještog programera potrebno oko 10 godina (Winslow, 1996.).

U posljednje vrijeme čuju se i razmišljanja da bi intuitivni učitelj mogao razviti programera sa solidnom razinom razumijevanja rada programa u roku od dvije do četiri godine uz uvjet izuzetno strpljivog i posvećenog rada sa studentom (Schulte i sur., 2010). Iz svega navedenog poželjno je razmatrati i istražiti način poučavanja programera početnika kao jedan od faktora koji značajno može utjecati na uspjeh pri rješavanju problema programiranjem.

Koliko je važan učitelj za postizanje uspjeha u podučavanju programera početnika? Iako učitelj teško može značajno utjecati na većinu osobnih obilježja učenika, jako je važno da učitelj dobro poznaje svoje učenike kako bi pronašao odgovarajuće metode rada te olakšao učenicima

usvajanje izazovnih programerskih koncepata. Veći broj istraživanja ističe pozitivne rezultate učinkovitosti novih pristupa, metodologija i okružja za učenje i poučavanje, ali često se zanemaruju važnost kvalitete i motivacije učitelja te koliko je točno taj faktor utjecao je na izvedbu istražene metode.

U kojoj mjeri motivacija učitelja i entuzijazam doprinose uspjehu nekih metoda i pristupa? Ljudski faktor je iznimno bitan u obrazovnom sustavu (Thompson i Barnes, 2009). Iskustva uspješnih škola ističu da je angažiranje pravih ljudi da postanu učitelji te njihov razvoj do učinkovitog odgajatelja kao i osiguravanje sustava za učenje koje nudi najbolju moguću pomoć za svakog učenika od presudne važnosti za kvalitetu obrazovanja (Ambrosio i Martins, 2011; McKinsey i Company, 2007).

Prvo istraživanje McCrackenove skupine pokazalo je koliko su učitelji slabo upoznati sa stvarnim razumijevanjem sadržaja koje se događalo u mislima učenika i potaknulo je neka nova istraživanja tog područja (McCracken i sur., 2001). Kasnija fenomenografska međunarodna istraživanja među učiteljima, provedena na nekoliko sveučilišta, pokazala su da bi učitelji istraživači mogli ponuditi mnoštvo odgovora na pitanja koja se odnose na uspjeh učenika pri rješavanju problema programiranjem te da su najuspješniji među učiteljima bili upravo oni koji su njegovali razvojno stajalište te učitelji koji su uključivali i koristili sve razine razumijevanja za stvaranje interakcije između učitelja i učenika (Kinnunen i sur., 2007). Takvi učitelji koriste različite pristupe u nastavi, koriste obrazovne materijale prilagođene različitim stilovima učenja, nude učenicima odgovarajuće primjere i probleme za učinkovit rad, prate i vrednuju rad učenika, ukratko takvi učitelji stvaraju pozitivno okružje za učenje u svojoj učionici.

2.5. Položaj računalnog razmišljanja u kurikulima K12 obrazovanja, u RH i drugdje

Sve veći napori, koji se u posljednje vrijeme ulažu u dogovor o jedinstvenoj definiciji računalnog razmišljanja, rezultirali su konkretnijim djelovanjem u smjeru promoviranja te vrednovanja računalnog razmišljanja, ali uglavnom na preddiplomskoj razini obrazovanja. Jedan od vodećih izazova poučavanja i promoviranja računalnog razmišljanja danas sve više je istraživanje K12 obrazovanja te mogućnosti uključivanja računalnog razmišljanja u kurikule računalne znanosti kao i kurikule nekih drugih predmeta osnovnih i srednjih škola. Istraživanja takve vrste još uvijek su nedovoljna.

Možemo istaknuti nekoliko značajnih projekata koji su krenuli u tome smjeru, a uglavnom predstavljaju zajedničko djelovanje istaknutih članova računalne znanosti, nacionalnih državnih tijela te organizacija kao Microsoft i Google (Johnson, 2016; Center for Computational Thinking, 2012; Exploring Computational Thinking, 2014) ili su usmjereni na uvođenje računalnog razmišljanja u škole (Computing in the Core, 2016; Computer Science for High School, 2016).

Kurikule računalnog razmišljanja može se promatrati i sa stanovišta razine programiranja koja je uključena u sam proces poučavanja. Takvi kurikuli nude različite načine uključivanja programiranja u poučavanje računalnog razmišljanja. Računalno razmišljanje može se uključiti u računalne, ali i temeljno ne-računalne kolegije te interdisciplinarne kolegije (npr. računalna biologija, računalna ekonomija..) čime programiranje i dalje ima važnu ulogu u računalnim kolegijima, ali prvenstveno služi kao alat za istraživanje područja računalne znanosti, te ostalih znanosti i društva općenito (Blackwell i sur., 2008; Developing Computational Thinking, 2015; Miller, i sur., 2014; Dierbach, i sur., 2011).

Drugim pristupom predlaže se integriranje računalnog razmišljanja kroz različite kolegije općeg obrazovanja tako da su kolegiji ipak razvijeni s ciljem poučavanja koncepata i vještina društvenih kolegija, ali s elementima računalnog razmišljanja. Takav pristup gotovo da ne inzistira na programiranju kao temeljnoj vještini (Blackwell i sur., 2008; Dierbach, i sur., 2011; Djurdjevic-Pahl i sur., 2017; Interdisciplinary Computational Thinking, 2017).

Treći pristup zagovara razvoj potpuno novog kolegija o računalnom razmišljanju koji koristi problemski pristup s naglaskom na znanstvenom istraživanju računalnim metodama u kojem programiranje ima važnu ulogu (Blackwell i sur., 2008).

Kakva je pojavnost računalnog razmišljanja u kurikulima K12 obrazovanja zemalja EU i šire? Neke države imaju dužu povijest poučavanja računalne znanosti, druge su tek nedavno

pristupile uvođenju vještina računalnog razmišljanja u svoje kurikule, dok ima i onih država koje su tek u fazi promišljanja o načinu integriranja računalnog razmišljanja planiranjem novih ili preuređivanjem postojećih kurikula. Sljedeća tablica prikazuje načine uključivanja računalnog razmišljanja, odnosno nekog dijela njegovih vještina u kurikule K12 obrazovanja (Tablica 2).

Tablica 2: Računalno razmišljanje u nekim kurikulima K12 obrazovanja

Država	**Oblik uključenosti računalnog razmišljanja u kurikul**
Australija	- Koncepti računalnog razmišljanja razvijaju se kroz kolegij Digitalne tehnologije, počevši od prve godine obrazovanja (Acara, 2015).
Velika Britanija	- Računalno razmišljanje standardni je dio novog nacionalnog kurikula računalne znanosti osnovnih i srednjih škola (Computing at School, 2013; Computing at School, 2014) .
Sjedinjene Američke Države	- Ne postoji zajednički pristup na nacionalnoj razini već opća preporuka CSTA organizacije za uvođenje računalnog razmišljanja u K12 obrazovanje od prve godine obrazovanja (CSTA, 2016).
Litva	- Vještine računalnog razmišljanja razvijaju se postupno kroz kolegij Informatika počevši od 5. razreda osnovne škole (modeliranje, algoritamsko razmišljanje, principi rada digitalne tehnologije, upravljanje podatcima, algoritmi i programiranje, virtualna komunikacija, sigurnost) (Dagiene i Stupiriene, 2016).
Novi Zeland	- Vještine i koncepti računalnog razmišljanja integrirane su u kolegij Tehnologija od prve godine učenja srednje škole (računalno modeliranje, apstrakcija…) (The New Zeland Curriculum Online, 2014).
Izrael	- Predmet Računalna znanost između ostalog uključuje algoritamsko razmišljanje, rješavanje problema, kreativnost (European Schoolnet, 2015).
Malta	- Planira se uvođenje sadržaja programiranja kao obvezan sadržaj za sve učenike jer je „računalno razmišljanje temeljna vještina svih, a ne samo računalnih znanstvenika“ (European Schoolnet, 2015). Računalno razmišljanje sastavni je dio osnovnoškolskog kurikula kroz predmet Digitalna pismenost i Informacijsko-komunikacijska tehnologija (Bocconi i sur., 2016).
Poljska	- Predmet Informatika pojavljuje se od prvog razreda osnovne škole, a od 2016. godine uvodi se novi kurikul koji ima glavni cilj motivirati učenike da primjenjuju računalno razmišljanje te da se što više angažiraju u ostalim predmetima primjenjujući strategije rješavanja problema (Bocconi i sur., 2016).

Austrija	-	Kurikul predmeta Informatika (u završnim razredima osnovne škole) stavlja naglasak na koncepte računalnog razmišljanja kao što su modeliranje, apstrakcija te rješavanja problema.
Hrvatska	-	Računalno razmišljanje i programiranje uvodi se kao jedna od četiri domene novog kurikula predmeta Informatika počevši od prve godine obrazovanja (apstrakcija, dekompozicija, algoritamsko razmišljanje i programiranje, upravljanje i prezentacija podataka, …) (Brođanac, i sur., 2016; Ministarstvo znanosti i obrazovanja, 2018).

Danas postoji veći broj mrežnih stranica, sponzoriranih od strane industrije, s mnogim resursima za učitelje sa sadržajima koji naglašavaju razlikovanje te prepoznavanje uzoraka, apstrakciju, modele, algoritme te analizu podataka i vizualizaciju ili nude primjere računalnog razmišljanja u drugim znanostima (Romero i sur., 2017). Nadalje, postoje i primjeri kurikula u kojima se računalno razmišljanje koristi za poučavanje matematike (Bootstrap World, 2015; Djurdjevic-Pahl i sur., 2017).

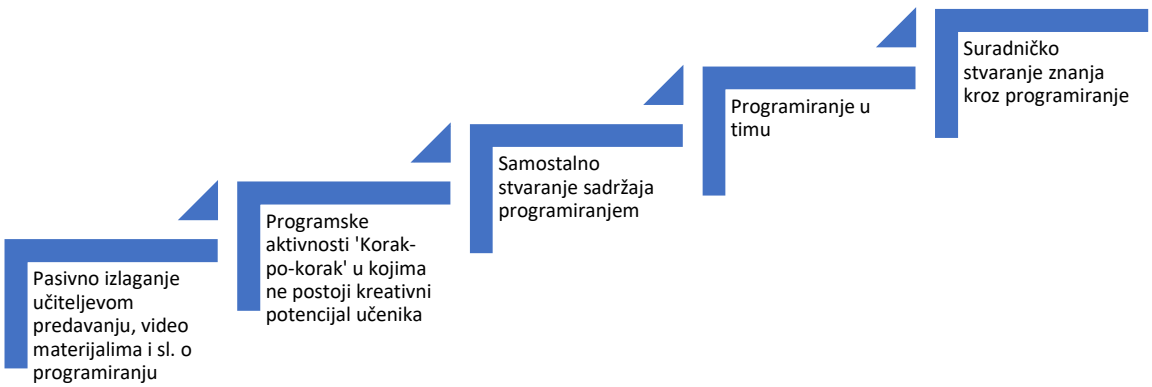
U novijim raspravama o definiciji računalnog razmišljanja, konceptima koji ga opisuju te njegovoj uključenosti u školskim kurikulima, može se zaključiti da je računalno razmišljanje uspješno našlo svoje mjesto u K-12 obrazovanju.

Iako se početno predstavilo kao 'opće korisna vještina rješavanja problema' koja bi se trebala prenijeti na ostale discipline, usprkos velikim očekivanjima i željama, istraživanja još uvijek nisu potvrdila takve rezultate (Tedre i Denning, 2016). Novi pristup poučavanju, koji napušta primjenu jednog univerzalnog pristupa i zagovara pristup poučavanju u kojem se stavlja veći naglasak na individualni stil učenja svakog učenika, jedan je od izazova pedagoškog okvira računalnog razmišljanja (Schulte, 2013).

2.6. Pristupi poučavanju računalnog razmišljanja

Usprkos mnogim izazovima definiranja računalnog razmišljanja rezultati istraživanja, koja proučavaju različite nastavne metode za uvođenje i pojačavanje učenja programiranja, ističu da takva nastavna praksa mora biti iskustvo učenja koje će učenik opisati kao učinkovito, ali i zabavno (Garneli i sur., 2015). Novija istraživanja stavova učitelja računalne znanosti o pristupima poučavanju koje oni primjenjuju u svojoj nastavnoj praksi, posebno ističu razvoj vještina računalnog razmišljanja uz kontekstualno učenje, upravljanje programskim kodom, suradničko učenje te učenje na daljinu (Sentance i Csizmadia, 2015).

U nastojanju da se pronađe još jedan opravdan razlog poučavanja programiranja, koje mnogi smatraju zahtjevnim za većinu učenika, često se navodi mogućnost programiranja da potakne kreativnost kod učenika. Takva tvrdnja mogla bi biti istinita, ako bi se kreativno programiranje promatralo kao pedagoška strategija, a ne samo kao tehnički alat ili skup tehnika kodiranja koje je potrebno naučiti (Romero i sur., 2017). Kreativnim programiranjem učenik se uključuje u proces osmišljavanja i razvoja originalnog rješenja kroz proces kodiranja. Pri tom procesu učenika se potiče na korištenje alata za kodiranje, ali i kreiranje te korištenje znanja iz drugih područja. Romero i sur. (2017) prepoznaju pet razina kreativnog angažmana pri poučavanju programiranja prema aktivnostima učenika (Slika 5) (Romero i sur., 2017).

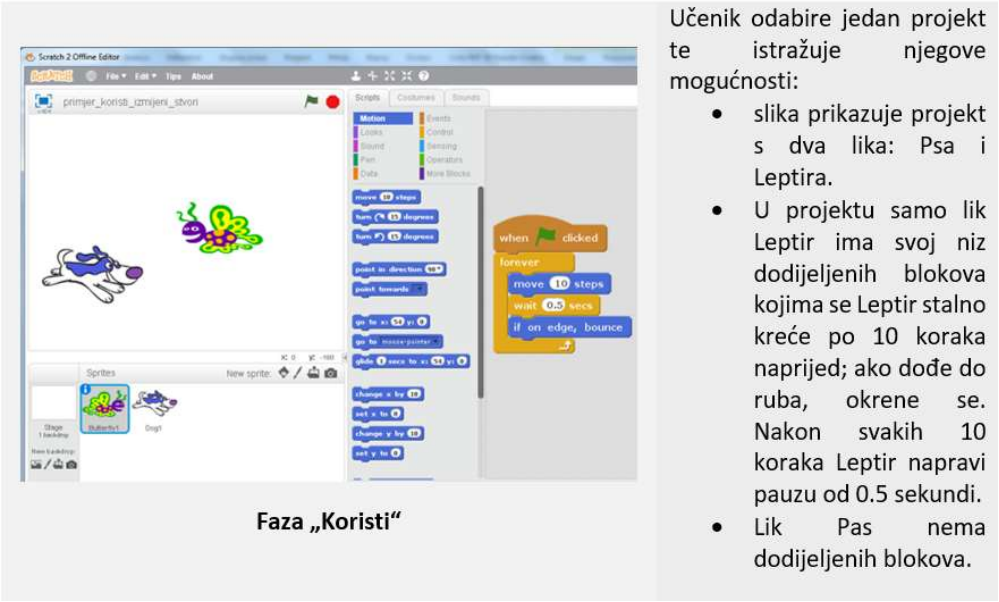


Slika 6: Razine kreativnog angažmana učenika pri aktivnosti Učenje programiranjem

U daljnjem tekstu opisat ćemo neke od pristupa koji su se pokazali kao mogući primjeri dobre prakse u poučavanju računalnog razmišljanja.

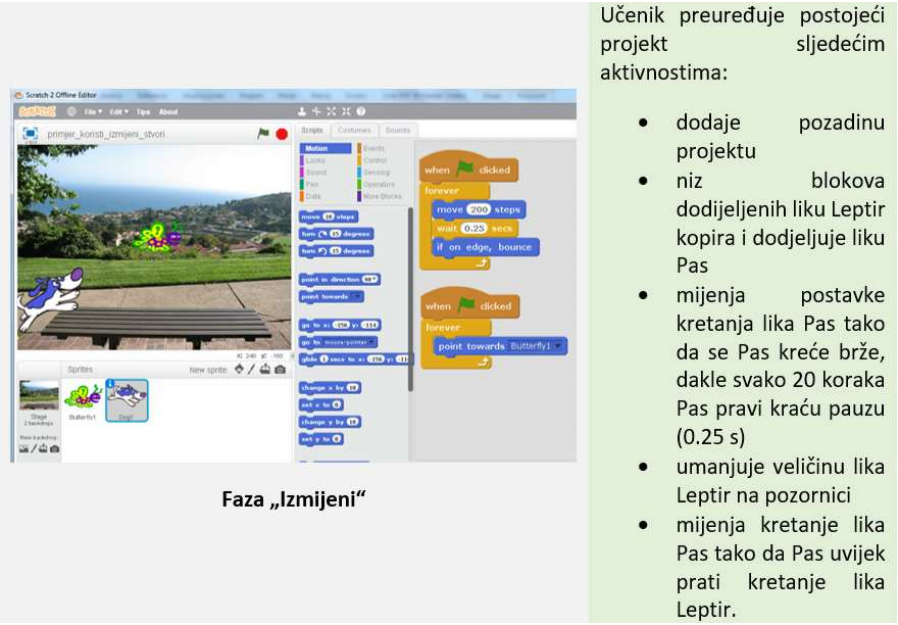
2.6.1. Poučavanje računalnog razmišljanja pristupom koristi-izmijeni-stvori

Trodijelni proces „**koristi–izmijeni–stvori**“ (engl. Use-Edit-Create) primjer je nastavne prakse poučavanja koja ima za cilj potaknuti učenika na računalno razmišljanje koristeći se pri tome bogatim računalnim okružjima (Lee i Ko, 2011; Armoni i sur., 2015; Meerbaum-Salant i sur., 2015). Bogata računalna okružja smatraju se sva ona okružja u kojima se podržane apstrakcije i mehanizmi mogu provjeravati, dalje prilagođavati, ali i njima upravljati. Dakle sva ona okružja u kojima neki korisnik može razviti vještine računalnog razmišljanja i postupno postati stvaratelj. Primjeri takvih okružja su SimCity, StarLogo TNG, Storytelling Alice, Scratch i ostali. Pristup „koristi – izmijeni – stvori“ temelji se na pretpostavci da „skaliranje“ pojačava duboke interakcije dok istovremeno promovira usvajanje i razvoj računalnog razmišljanja. U prvoj fazi tzv. „koristi“ (engl. Use) učenicima je dostupan neki gotov primjer računalnog koda, računalnog modela, računalne igre i sl. (Slika 7).



Slika 7: Primjer pristupa "Koristi - izmijeni - stvori" – faza Koristi

Pretpostavlja se da će kroz neko vrijeme učenici postupno započeti „izmjenu“ tog koda, modela ili igre, čime započinje druga faza procesa „Izmijeni“. Najčešće se u početku izmjene odnose na neka jednostavna obilježja kao što je izmjena boje lika ili boje scene te općenito nekih vizualnih obilježja, a daljnjim usavršavanjem učenik proizvodi i izmjene koje se odnose na kretanja likova (Slika 8).

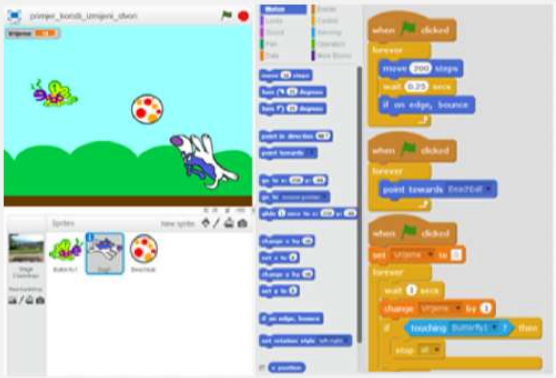


Slika 8: Primjer pristupa "Koristi - izmijeni - stvori" – faza Izmijeni

Završna faza, faza „stvori“ (engl. Create), nastaje stalnim mijenjanjem i popravljanjem, odnosno uljepšavanjem računalnog koda (modela, igre …) čime se razvijaju nove vještine, a tuđi kod postaje vlastiti (Slika 9). U primjeni ovog pristupa izuzetno je važno u svim fazama održavati primjerenu razinu izazova kako bi se smanjila napetost i povećao rast učenikovih sposobnosti i vještina te učenika motiviralo na angažiranje u sve zahtjevnijim izazovima.

Ako je razina izazova neprimjerena razvoju učenikovih vještina, aktivnosti mogu postati preteške za učenike ili naprotiv izazvati dosadu i odustajanje kod učenika (Repenning i Ioannidou, 2008). Opisani pristup poučavanju ne smije se primjenjivati kao slijedni niz navedenih faza već kao oblik cikličkog procesa u kojem se neke faze i višestruko ponavljaju.

Učenici koriste, istražuju i mijenjaju gotova računalna rješenja pri čemu kroz navedene faze prolaze željenom brzinom. Neki će se odmah baciti na stvarno programiranje, odnosno uređivanje postojećeg ili stvaranje vlastitog rješenja dok će drugi više vremena posvetiti analizi i istraživanju postojećeg rješenja prije nego se ohrabre za stvaranje novog rješenja. Takav pristup može povećati samopouzdanje učenika dok istovremeno razvija vještine računalnog razmišljanja (Zhang, 2019).



Faza „Stvori“

Učenik dalje mijenja projekt koji sada ima sve manje sličnosti s početnim projektom:

- dodaje lik Lopta i njemu pridružuje blokove naredbi kojima se Lopta kreće nekom slučajnom putanjom po pozornici.
- mijenja i unapređuje mogućnosti kretanja lika Pas tako da Pas prati kretanje lika Lopta
- uvodi završetak projekta dodirom likova Pas i Leptir
- ponovo mijenja pozadinu
- uvodi brojač koji u lijevom gornjem vrhu ispisuje vrijeme izvođenja projekta

Slika 9: Primjer pristupa "Koristi - izmijeni - stvori" (Scratch): faza Stvori

Predstavljeni pristup poučavanja „Koristi-izmijeni-stvori“ moguće se realizirati i u nekom tekstualnom programskom jeziku. Sljedeće slike prikazuje primjer takvog pristupa u programskom jeziku Python, u radu s nizom podataka (Slika 10).

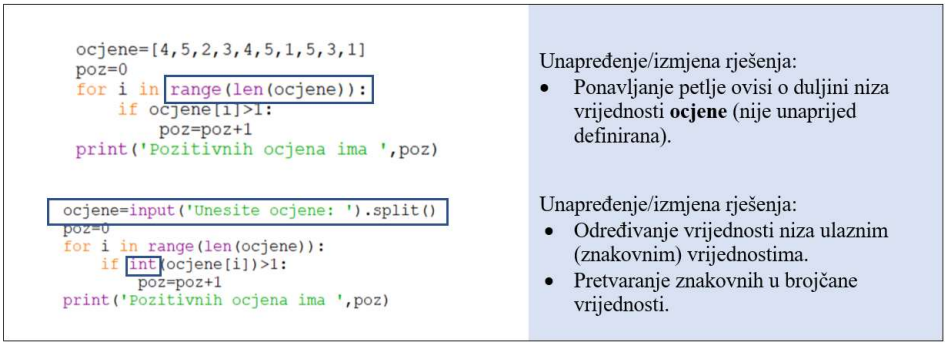
```
ocjene=[4,5,2,3,4,5,1,5,3,1]
poz=0
for i in range(10):
    if ocjene[i]>1:
        poz=poz+1
print('Pozitivnih ocjena ima ',poz)
```

Karakteristike rješenja:

- Definiranje vrijednosti niza **ocjene** unutar računalnog rješenja.
- Ponavljanje petlje određeno je konstantnom vrijednošću.
- Uporaba brojača (poz) za određivanje ukupnog broja pozitivnih ocjena.

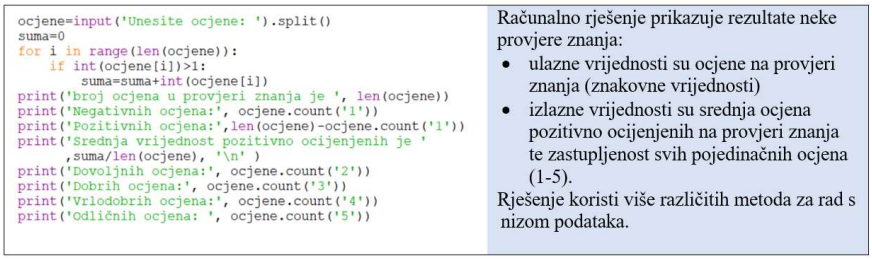
Slika 10: Primjer pristupa Koristi – Izmijeni – Stvori (Python): faza Koristi

U početnoj verziji računalnog rješenja predstavlja se način definiranja niza vrijednosti te pretraživanje tog niza s ciljem prebrojavanja pozitivnih vrijednosti, faza Koristi (Slika 10). U sljedećoj fazi prepoznaju se ograničenja računalnog rješenja: vrijednost niza određuje se unutar samog rješenja, odnosno rješenje je prilagođeno zadanom nizu vrijednosti. Omogućavanjem upisivanja vrijednosti niza kroz ulazne (znakovne) vrijednosti računalno rješenje radit će za proizvoljni niz upisanih ocjena, faza Izmijeni (Slika 11).



Slika 11: Primjer pristupa Koristi – Izmijeni – Stvori (Python): faza Izmijeni

Daljnjim unapređenjem računalnog rješenja uvodi se nova funkcija (count) kojom se prebrojavaju ocjene, a petlja se koristi samo za zbrajanje pozitivnih ocjena. Prepoznaju se nove mogućnosti: prikaz srednje ocjene, broja pozitivnih i negativnih ocjena te zastupljenosti pojedinačnih ocjena zadane provjere znanja, faza Stvori (Slika 12).



Slika 12: faza Stvori - programski jezik Python

Treba naglasiti da se završno rješenje gotovo uvijek može dalje nadograđivati u cilju stvaranja boljeg rješenja npr. pretvaranje niza znakovnih vrijednosti u niz brojčanih vrijednosti (funkcija list(map(int.ocjene)) i slično.

2.6.2. Poučavanje računalnog razmišljanja primjenom grafičkog okružja za programiranje

Mnoga istraživanja poučavanja programiranja te koncepata računalnog razmišljanja bave se primjenom nekog grafičkog okružja. Istraživači obrazovanja računalne znanosti slažu se da visoko skalirana okružja za programiranje nude početnicima nježniji put prema uspjehu, no neminovno je da pojedina okružja zbog svoje ograničene izražajnosti neće moći držati korak s jačim konvencionalnijim jezicima. Neka su se okružja pokazala uspješnima jer omogućuju

učenicima postizanje boljih rezultata pri usvajanju osnovnih koncepata računalnog razmišljanja kroz programiranje upravo primjenom pristupa posredovanog transfera (engl. Mediated transfer) (Dann i sur., 2012). Jedan od načina kojim se pokušava doskočiti takvom problemu je razvoj računalnog razmišljanja koristeći apstrakciju, skaliranje te „igranje uloga“. Takvim pristupom omogućava se prijelaz između različitih okružja, odnosno omogućava se učenicima napredovanje iz jednog u drugo okružje odgovarajućom brzinom (Marghitu, i sur., 2010).

Nešto drugačiji pristup predstavlja pedagoški model kojim se omogućava skaliranje radom u Microsoft Kodu, Alice te LegoMindstorm NTX-g okružjima (Touretzky i sur., 2013). Ovaj pedagoški model uvijek predviđa za početak rad u okružju Microsoft Kodu jer ono omogućuje učenicima slaganje njihovih vlastitih programa, organiziranih u obliku sljedova naredbi *Kad/Radi* (engl. *When/Do***)**. Posebnost ovog pristupa je u potpunoj zaštiti učenika od sintaksnih pogrešaka jer učenici sljedove slažu birajući ikone iz sadržajnih izbornika umjesto da ih samostalno tipkaju. Također, takva okružja ne spominju pojam varijable koji je programerima početnicima jako zahtjevan, već umjesto njega koriste objekt *Rezultat* (engl. Score) koji ima za cilj sačuvati vrijednosti koje se koriste tijekom izvođenja. Nakon primjene okružja za programiranje Kodu autori predlažu nastavak s Alice okružjem te nadalje završetak s Lego Mindstorm NTX-G okružjem.

U povratnim informacijama učenici su izjavili da im je Alice okružje ponekad zahtjevno i izazovno zbog većeg broja prozora, manje intuitivnog povuci-spusti okružja te potrebnog tipkanja u odnosu na Kodu okružje. Ipak, važno je naglasiti da, iako su mnogi učenici treću fazu rada doživjeli kao veoma zahtjevnu, čak 71% učenika izjavilo je da bi vjerojatno ili sigurno željelo raditi s robotima ako bi za to imali prilike u nekom dužem periodu (Touretzky i sur., 2013).

2.6.3. Poučavanje računalnog razmišljanja bez rada na računalu

Za razvoj vještina računalnog razmišljanja mogu se koristiti i aktivnosti koje ne uključuju rad s računalom. Primjer takvih aktivnosti su tzv. *unplugged* aktivnosti (Bell i sur., 2015) za simuliranje algoritama bez uporabe računala koje su se učenicima pokazale izuzetno zabavne, ali i učinkovite za ilustriranje važnih koncepata računalne znanosti. *Unplugged* aktivnosti čine skup gotovih scenarija učenja kojima se kroz igre i zagonetke, primjenom većeg broja kinestetičkih i grupnih aktivnosti, nastoji približiti računalna znanost mlađim učenicima, ali i informatičkim početnicima starije dobi. Ovaj oblik poučavanja računalnog razmišljanja privlači sve veću pažnju zajednice obrazovanja računalne znanosti, ali i globalnih organizacija, stoga

ne iznenađuje mnoštvo dostupnih besplatnih mrežnih nastavnih materijala u obliku video uradaka, scenarija učenja, pomoćnih nastavnih materijala (CS Education Research Group, 2015). Novija istraživanja govore u prilog činjenici da CS Unplugged aktivnosti zaista potiču računalno razmišljanje te da učenici usvajaju koncepte računalnog razmišljanja koji su utkani u *Unplugged* aktivnosti (Rodrigez, 2015). No, iako se ovaj oblik aktivnosti često uspješno primjenjuje u raznim obrazovnim grupama nakon redovne nastave ili kroz neke projekte i kampove, njihovo uključivanje u redovnu nastavu ipak ukazuje na određene nedostatke, kako u strukturi nastave tako i u samom nastavnom sadržaju te sredstvima rada (Rodriguez i sur., 2016; Imberman i sur., 2014; Sentance i Csizmadia, 2015).

Kako bi se razvoj računalnog razmišljanja potaknuo u što ranijoj dobi učenika, neki zagovaraju njegovo uvođenje kroz predmet Matematika, pri čemu se kod učenika nastoji razviti apstrakcija te logičko razmišljanje bez pisanja računalnih programa (Djurdjevic-Pahl i sur., 2017). Takvim pristupom nudi se niz matematičkih nastavnih jedinica koje su utemeljene na nekoj aktivnosti računalnog razmišljanja. Tako se na primjer, predlaže vrednovanje apstrakcije kroz aritmetiku i prostornu koordinaciju, prepoznavanje uzoraka kroz geometriju i umjetnost, algoritama kroz kriptografiju i geometrijske slagalice i sl.
Primjenom računalnog razmišljanja, a osobito modeliranja i simulacija u prirodoslovnim i društvenim znanostima istaknula se poveznica između računalstva i ostalih nastavnih predmeta. Takvim se pristupom vještine i teme računalnog razmišljanja uključuju kao sastavni dio drugih predmetnih kurikula, a istovremeno pronalazi nov način učenja odabranih (neinformatičkih) nastavnih tema primjenom interdisciplinarnog pristupa (Interdisciplinary Computational Thinking, 2017). Moguće je pronaći nastavne materijale i scenarije poučavanja kojima se npr. razmatranjem modela neurona analizira način rada mozga, evolucijskim modelima pomažu objasniti biološke pojave, računalnim modelima olakšava razumijevanje klimatskih promjena i slično.

Svim navedenim pristupima poučavanju računalnog razmišljanja predstavljeni su primjeri dobre prakse poučavanja za koje postoje istraživanja koja ističu njihovu učinkovitost ili se radi o novijim pristupima čija se učinkovitost još uvijek istražuje. Može se primijetiti da se poučavanje računalnog razmišljanja može smjestiti u programerske, ali i neprogramerske kontekste, no kako je računalno razmišljanje svojim konceptima ključno vezano uz informatičke sadržaje, očita je nešto veća zastupljenost metoda poučavanja koja uključuju rad s računalom i/ili računalnim aplikacijama, najčešće grafičkim okružjima za programiranje.

Sve se više naglašava mogućnost i sloboda učenika da pri učenju napreduje brzinom koja je u skladu s njegovim kognitivnim mogućnostima. U nastojanju da se motivacija i angažman učenika pri poučavanju zadrže što dulje, učiteljima se predlažu pristupi koji će učenicima biti zanimljivi, zabavni i neprezahtjevni na početku, a ujedno im omogućiti osnaživanje u procesu učenja te jačanje samopouzdanja. Tek nakon stvaranja 'ugodne zone učenja' za učenike, preporučuje se uvođenje složenijih sadržaja u proces poučavanja.

2.7. Vrednovanje računalnog razmišljanja

Vrednovanje je ključ svakog učenja (Jeager, 1998), pa se upravo kvalitetno planiranim te pravodobno provedenim vrednovanjem može jednostavno otkriti kako učenici uče. Opće je poznato da na uspjeh pri učenju mogu utjecati različiti faktori, od metoda i strategija koje ovise o učitelju, ali i učeniku, nadalje društvenih faktora te faktora vezanih za mjesto učenja do nekih osobnih obilježja samog učenika.

Vrednovanje programerskih koncepata kroz vrednovanje poznavanja novih definicija te naredbi programskog jezika daleko je jednostavnije od vrednovanja načina na koji učenici primjenjuju računalstvo i programske jezike za rješavanje problema te dizajniranje različitih računalnih uradaka. Računalno razmišljanje sastoji se od malih zadataka ili praksi koje se koriste pri rješavanju određene vrste problema pri čemu zadatci mogu nuditi drugačiji način pristupa problemu ili temi. Vrednovanje računalnog razmišljanja tjera nas na pronalaženje dokaza kod učenika o postojanju dubljeg razumijevanja samog problema koji se rješava, te razumijevanja načina stvaranja kodiranog rješenja kojeg učenik primjenjuje. Na primjer, sposobnost učenika da rastavi složeni problem na manje dijelove ukazuje na njegovo razumijevanje problema kao cjeline, dok njegovo apstrahiranje, pojednostavljivanje ideja ukazuje na njegovu svijest o ključnim točkama promatranog problema.

Postoje pristupi vrednovanju razvoja računalnog razmišljanja na kojima se trenutno radi ili su još uvijek u razvoju, a mogu poslužiti kao dobar temelj razvoja općeg pristupa vrednovanju računalnog razmišljanja.

2.7.1. Vrednovanja računalnog razmišljanja uz primjenu programiranja i/ili računala

Vrednovanje računalnog razmišljanja temeljeno na dokazu

Vrednovanje računalnog razmišljanja koji se temelji na dokazu (engl. Evidence-Centered Design, ECD) predstavlja pristup vrednovanja u obrazovanju koji koristi dokazne argumente (Almond i sur., 2002; Mislevy i sur., 2003; Mislevy i Haertel, 2006). Takvim se pristupom pri vrednovanju stavlja naglasak na dokaz o kompetencijama kao osnovi za izgradnju izvrsnih zadataka vrednovanja. Vrednovanje temeljeno na dokazu može se opisati s pet različitih slojeva: analiza i modeliranje domene (područja), konceptualni okvir vrednovanja te primjena i objava vrednovanja. Navedenim slojevima obuhvaćene su aktivnosti poput analize domene (područja) vrednovanja, odabira i administriranja zadataka, interakcije sa studentima radi predstavljanja ili preuzimanja radnih materijala/uradaka, vrednovanje odgovora zadataka uz prikupljanje dokaza znanja. Ovaj pristup ističe složenost znanja i vještina te druge značajke i

ponašanja koja bi trebala biti vrednovana, poput osnovnog ICT znanja, mogućih radnih proizvoda i zapažanja te promjenjivih obilježja povezanih s vrednovanjem (Mislevy i Riconscente, 2005). Kao najveća prednost ovog pristupa vrednovanja ističe se upravo stvaranje alata/instrumenta vrednovanja koji mogu bolje odražavati i mjeriti ono što se događa u učionici te prikupljanje rezultata vrednovanja koji su snažno potkrijepljeni dokaznim argumentima (Hendrickson i sur., 2013). Pristup dizajnu temeljen na dokazima obično je prikazan u pet slojeva koje detaljno prikazuje sljedeća tablica (Tablica 3).

Tablica 3: Slojevi pristupa vrednovanju temeljenom na dokazima (ECD)

Slojevi dizajna usmjerenog prema dokazima	Osnovna uloga	Primjeri i ključni koncepti
Analiza područja	- Prikupljanje dovoljnog broja informacija o odabranom području računalnog razmišljanja, znanju koje se stvara, usvaja pa čak i komunicira	- Apstrakcija, automatizacija - Alati (programski jezici) - Predstavljanje (pričanje priča)
Modeliranje područja	- Izraziti argument vrednovanja u narativnom obliku na osnovu informacija prikupljenih pri analizi područja	- Specifikacija vještina, znanja te ostalih obilježja koje treba vrednovati
Konceptualni okvir vrednovanja	- Izraziti argument vrednovanja u strukturama i specifikacijama zadataka, testova, procedura vrednovanja te modela mjerenja	- Modeli učenika, dokaza te zadataka - Rubrike mjerenja - Modeli mjerenja - Uzorci testa - Specifikacije testa
Primjena vrednovanja	- Primjena vrednovanja	- Zadatci, materijali zadataka - Poboljšavanje postupaka vrednovanja pilot testiranjem - Prilagodba modela mjerenja
Objava vrednovanja	- Uskladiti interakcije učenika i zadataka - Bodovanje zadataka i testova - Izvještavanje	- Predstavljanje zadataka - Vrednovanje rezultata

U sklopu PACT projekta (engl. Pricipled Assessment of Computational Thinking, SRI Education Group) (SRI Education, 2020) također se koristio ECD pristup pri kreiranju

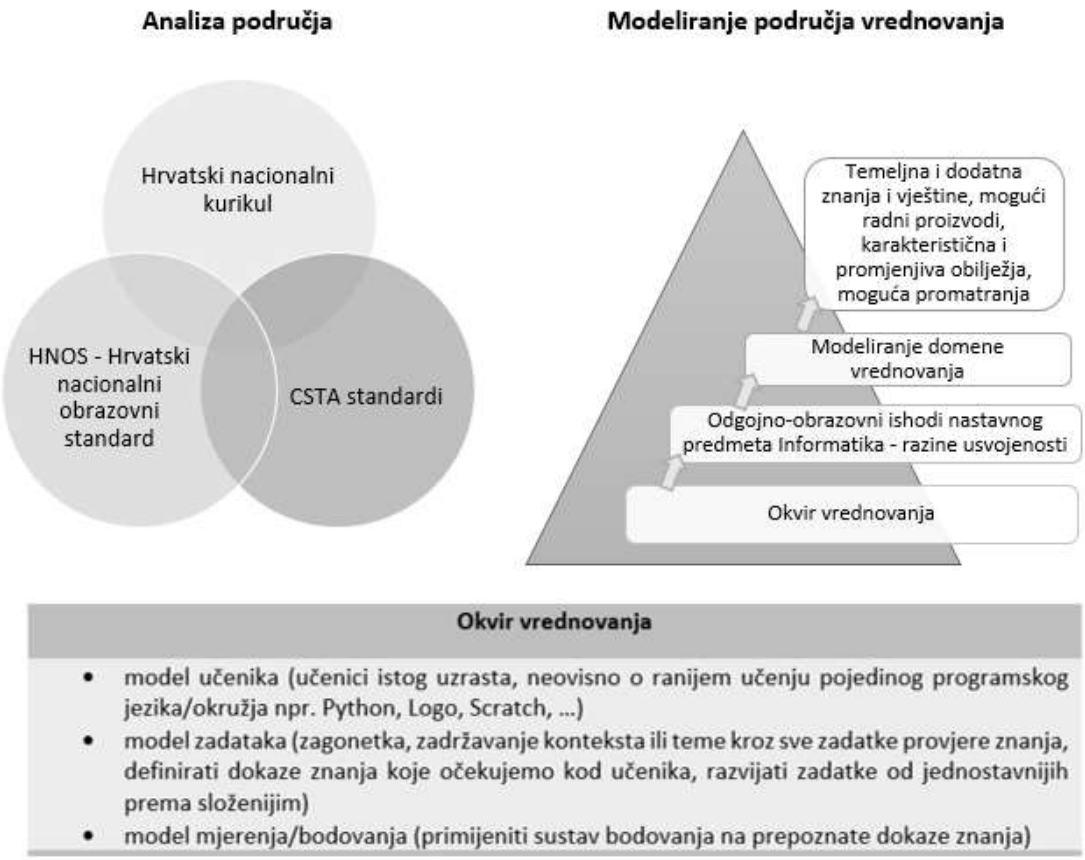
vrednovanja s ciljem da se osnovni koncepti računalnog razmišljanja prikažu posebnim obrascima i to: strukturiranim opisom domene (Gdje se učenje odvija?), dokazom (Što tražimo u pojedinom zadatku?) te argumentima (Što će proizaći iz određenog dokaza?). Takvi obrasci trebali bi naglasiti primjenu i preispitivanje vještina koje se koriste tijekom rješavanja računalnih problema, a ne ocjenjivanje znanja o konceptima potrebnim za primjenu takvih vještina. U svom radu Istraživačka skupina Stanford Research Institute (SRI) kroz projekt PACT opisuje moguće načine primjene vrednovanja za svaki sloj ECD pristupa kako bi se stvorila cjelovita praksa vrednovanja teško mjerljivih konstrukata računalnog razmišljanja (Bienkowski, 2015; Bienkowski, 2016; Bienkowski i sur., 2015; Bienkowski, i sur., 2017;). Takav pristup uvijek započinje propitivanjem o složenosti znanja, vještina te ostalih obilježja koja bi se trebala vrednovati. Također, za takav pristup vrednovanju važno je odrediti kakva bi ponašanja ili izvedbe trebale otkriti takve konstrukte te kakve bi zadatke trebalo zadati kako bi se uočila takva ponašanja. Ovim pristupom stvara se strukturiran opis dokaza za neko područje, koji se može koristiti za proizvodnju zadataka vrednovanja.

Jedan primjer vrednovanja temeljenog na dokazu, razvijen je u skladu s novim hrvatskim kurikulom predmeta Informatika, a s ciljem formativnog i sumativnog vrednovanja računalnog razmišljanja u stvarnoj razrednoj situaciji (Bubica i Boljat, 2018). Vrednovanje ističe analizu područja domene, modeliranje te okvir područja vrednovanja (Slika 13), zatim provedbu vrednovanja te analizu rezultata (Slika 14). Za razliku od PACT metodologije koja stavlja naglasak na vrednovanje računalnog razmišljanja interdisciplinarnim pristupom, kroz nekoliko STEM predmeta, ovaj pristup vrednovanju stavlja naglasak na vrednovanje u stvarno razrednoj situaciji, unutar jednog predmeta, predmeta Informatika.

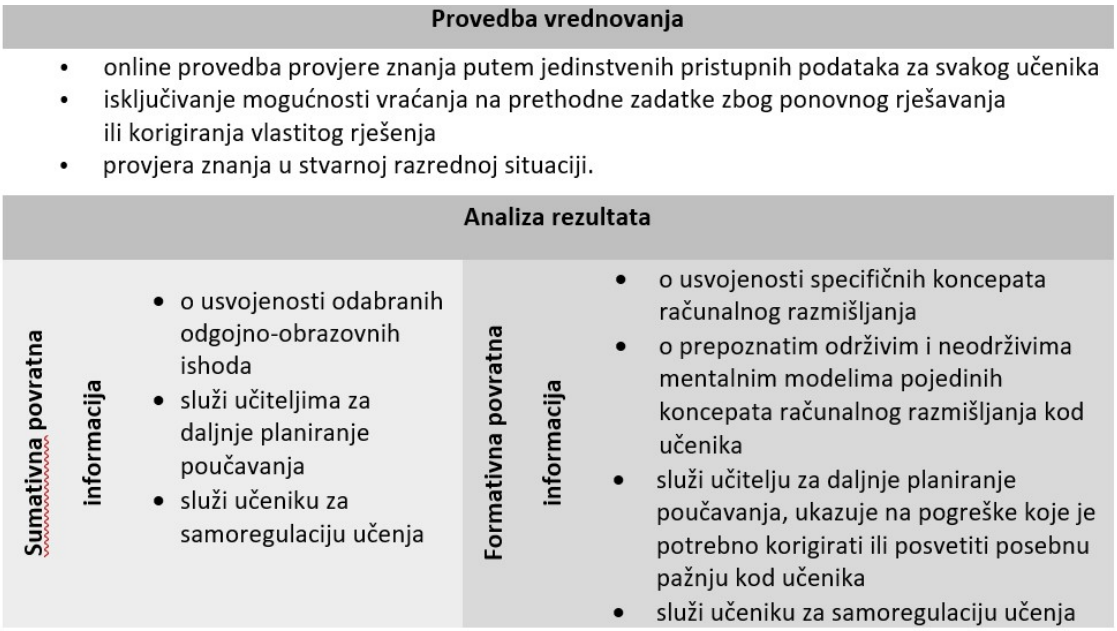
Prvi rezultati istraživanja pokazali su da je predloženi pristup podjednako prikladan za dječake i djevojčice. Učenici s boljim matematičkim ili općim akademskim postignućima vjerojatno će postići bolji rezultat na predloženom modelu vrednovanja. Navedeni su rezultati u skladu s relevantnim istraživanjima i govore u prilog primjerenosti takvog modela za opću primjenu u nastavnoj praksi. Primjer vrednovanja računalnog razmišljanja pokazao je prihvatljiv indeks pouzdanosti. Visoko podudaranje rezultata vrednovanja koje su proveli istraživač i dodatni neovisni ocjenjivač, govori u prilog veće pouzdanosti alata.

Kako bi se ispitala sadržajna valjanost, odabrani su rezultati istraživanja dijela učenika te uspoređeni s rezultatima koje su postigli na zadacima računalnog razmišljanja izazova Dabar (Suradnici u učenju, 2020). Iako je predloženi model vrednovanja pokazao bolje karakteristike

(koeficijent pouzdanosti, usklađenost pojedinog vrednovanja s ishodima predmetnog kurikula, međusobna korelacija dvaju vrednovanja itd.) u odnosu na vrednovanje računalnog razmišljanja zadacima natjecanja Dabar, za pouzdani zaključak trebalo bi provesti istraživanje na većem uzorku učenika.



Slika 13: Model vrednovanja temeljenog na dokazu – analiza i modeliranje te okvir područja vrednovanja



Slika 14: Model vrednovanja temeljenog na dokazu – provedba vrednovanja te analiza rezultata

PACT metodologija koja uz ECD pristup stvara cjelovitu praksu vrednovanja teško mjerljivih konstrukata računalnog razmišljanja primijenjena je i u "VELA vrednovanju" (Grover, 2020), gdje je razvijen instrument za mjerenje usvajanja uvodnih koncepata programiranja (razredi 6.-8.). Uzorci dizajna vrednovanja kao i mjerni instrument namijenjeni su mjerenju razumijevanja pojmova poput varijabli, izraza, petlji, uvjeta i apstrakcija. Ti pojmovi predstavljaju osnovne pojmove većine uvodnih tečajeva programiranja, bez obzira na programski jezik ili okruženje koje se pri poučavanju koristi.

Osim ovih koncepata, vrednovanje VELA također obuhvaća i zadatke s neispravnim dijelom koda koji se, uz prethodno navedene koncepte, koriste za vrednovanje razumijevanja pojmova poput događaja. U sklopu vrednovanja razvijene su rubrike i dizajnerski obrasci koji mogu poslužiti ostalim učiteljima kao dobri predlošci za razvijanje vlastitih vrednovanja te osiguravanje šire primjene instrumenta vrednovanja. Vrednovanje kvalitete učenja od iznimne je važnosti za učenike i učitelje, a dobro planirano vrednovanje te kvalitetna povratna informacija učitelja može poboljšati stjecanje kompetencija i omogućiti učenicima samovrednovanje svog napretka (Pellegrino, 2020).

Vrednovanje računalnog razmišljanja primjenom dizajniranja

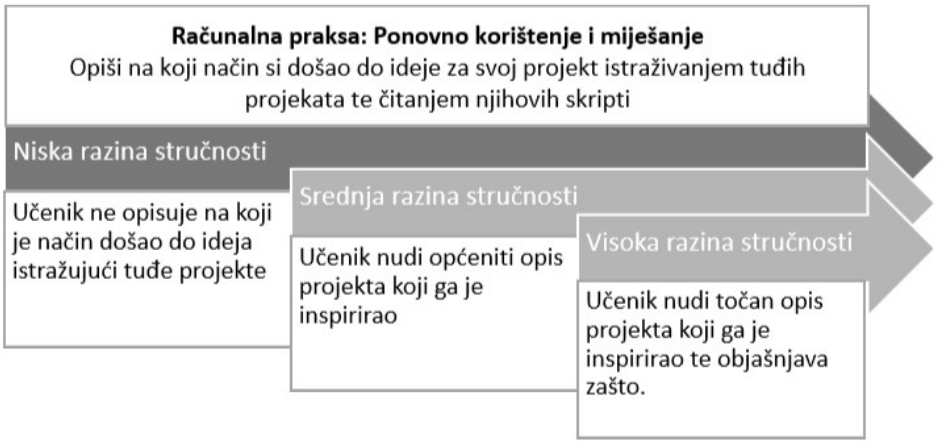
U nastojanju da se potaknu što više razine kreativnog programiranja kod učenika u procesu poučavanja i vrednovanja, mnogi zagovaraju primjenu dizajniranja. Kreativnim pristupom programiranju stvaraju se originalna računalna rješenja koja imaju efikasnu implementaciju te manju složenost npr. manji broj varijabli, operacija, grananja, ugniježdenih petlji…, a rješenje i

dalje udovoljava svim zahtjevima koji su početno definirani. Pristup poučavanju koji uključuje dizajniranja zagovaraju autori Brennan i Resnick (Brennan, i Resnick, 2012) te predlažu vrednovanja sljedećim postupcima:

- analizom portfolija projekata
- intervjuima temeljenim na dokumentaciji i uradcima
- razvojem dizajna scenarija.

U pristupu vrednovanju koji koristi intervju temeljen na dokumentaciji i uradcima, učenici sudjeluju u razgovoru o njihovim računalnim proizvodima i praksi. Intervjui se provode u tri vremenske točke: na uvodnim predavanjima o Scratch-u, u srednjem dijelu procesa učenja te u završnoj fazi učenja pri čemu rubrike za ocjenjivanje promatraju fluidnost računalne prakse i to kroz: testiranje i ispravljanje pogrešaka, eksperimentiranje i ponavljanje, apstrahiranje i moduliranje te ponovno korištenje i miješanje/preslaganje.

Stručnost računalne prakse procjenjuje se kroz tri razine: niske, srednje te visoke razine stručnosti. Sljedeća slika (Slika 15) prikazuje primjer rubrike vrednovanja za računalnu praksu Ponovno korištenje i miješanje (engl. Reusing and remixing) (Brennan i Resnik, 2012).



Slika 15: Primjer rubrike vrednovanja za računalnu praksu

U pristupu vrednovanju dizajniranjem scenarija učenici se susreću s nizom projekata u kojima sudjeluju kroz četiri različite perspektive i to: propitivanje, proširivanje, popravljanje i ponovno miješanje. Učenicima su ponuđena tri skupa od po dva Scratch projekta različite složenosti. Projekti u pojedinim skupinama jednaki su s obzirom na složenost, uključene računalne koncepte i praksu. Istovremeno projekti imaju različitu tematiku kako bi se angažirali učenici različitih interesa.

Osnovni zadatak svakog učenika u radu na nekom projektu je:

- objasniti cilj pojedinog projekta (propitivanje)
- opisati kako se taj projekt može proširiti/poboljšati (proširivanje)
- popraviti neku pogrešku (popravljanje) te
- nadopuniti projekt dodajući mu neko obilježje (ponovno miješanje).

U dijelu vrednovanja koji se bavi dokumentacijom, pored stvaranja kreativnih projekata od učenika se očekuje i stvaranje uočljivih tragova njihovog rada na projektu. Takvi tragovi mogu se realizirati u obliku papirnatog ili digitalnog dnevnika, ali i primjenom Scratch-ovog svojstva komentiranja kojim se mogu objasniti neka obilježja projekta. Također, tragovi se mogu ostvariti i prikazima zaslona kojima će se grafički predstaviti projekt, njegova namjena ili osnovne prednosti i nedostatci.

Iako se primjenom ovog pristupa napredovanje učenika od prvog prema trećem pristupu pokazalo produktivno, ipak sa stanovišta vrednovanja, niti jedan od navedenih pristupa nije se pokazao kao posebno učinkovit za razumijevanje nastale promjene u načinu računalnog razmišljanja kod učenika (Brennan i Resnick, 2012). Primjena neke kombinacije navedenih pristupa možda bi ponudila nešto bolje rezultate, no istovremeno pojavila bi se dodatna ograničenja u obliku potrebnog vremena za realizaciju procjene, broja učenika koji sudjeluju u procjeni i sl. zbog čega bi se pristup izuzetno teško proveo u stvarnom razrednom okružju.

Online vrednovanje računalnog razmišljanja – alat Dr. Scratch

U mnogim do sada navedenim istraživanjima, koja se bave vrednovanjem računalnog razmišljanja, spominje se grafičko okružje Scratch. Osim što se radi o bogatom grafičkom okružju za uvođenje programiranja, koje putem mrežne zajednice nudi mnoštvo primjera i tutorijala za učenje i istraživanje, ovo je okružje specifično u odnosu na ostala okružja po tome što ima vlastiti alat za vrednovanje projekata u skladu s konceptima računalnog razmišljanja: besplatni online alat Dr. Scratch (Dr.Scratch, 2019). Koncepti kao što su: logika, prezentacija podataka, paralelizam, sinkronizacija, interakcija korisnika, kontrola tijeka, apstrakcija, udvostručeni blokovi, neispravni nazivi, dijelovi neaktivnog koda (tzv. mrtvog koda) te inicijalizacija obilježja likova, vrednuju se kroz tri razine usvojenosti (Slika 16). Primjena svakog od navedenih koncepata u računalnom rješenju opisana je odgovarajućim brojem bodova (0-3) pri čemu se određena razina usvojenosti pojedinog koncepta vrednuje s jedan, dva ili tri boda. Ako koncept nije uključen u navedeno rješenje ili sadrži neku temeljnu pogrešku vrednuje se s nula bodova. Uz opisano bodovanje predlaže se moguća nadogradnja koncepta s

konkretnim primjerom takve nadogradnje u obliku Scratch programa. Na primjer, ako je koncept Logika vrednovan s nula bodova, pretpostavlja se učenik nema nikakva saznanja o tom konceptu i navodi se primjer jednog Scratch bloka koji sadrži odluku (logičku vrijednost) te opisuje djelovanje takvog bloka. Na sljedećoj razini (1 bod) pretpostavlja se usvojenost jednostavne naredbe grananja (if…) te nudi konkretan primjer primjene složene naredbe grananja (if…else...) s opisom djelovanja takve naredbe u konkretnom slučaju. Nadalje, uspješna uporaba složenog grananja u računalnom rješenju vrednovat će se s dva boda te predložiti i opisati uporaba logičkih operatora kojima se unutar naredbe grananja mogu obaviti višestruka vrednovanja logičkih uvjeta. Takva razina uporabe logike boduje se s tri boda.

Koncepti	Dr. Scratch – način vrednovanja
Logika	Mogu li se projekti ponašati različito ovisno o situaciji?
Prezentacija podataka	Postoje li potrebe informacije o samim likovima kako bi sve funkcioniralo na ispravan način, npr. pozicija, usmjerenje te veličina likova i sl.?
Paralelizam	Postoji li mogućnost istovremenog odvijanja stvari?
Sinkronizacija	Događaju li se aktivnosti likova željenim redoslijedom?
Interakcija korisnika	Postoji li mogućnost da korisnik svojim akcijama provocira neke nove situacije u projektu?
Kontrola tijeka	Primjenjuju li se različite algoritamske strukture kojima se kontrolira ponašanje likova u projektu?
Apstrakcija	Je li primijenjena vještina dijeljenja problema na manje dijelove koji rješavaju određena pitanja?
Neispravni nazivi	Imenovanje likova
Neaktivan kod	Postoje li dijelovi koda koji se nikada ne izvršavaju?
Inicijalizacija atributa	Jesu li inicijalizacije obilježja pojedinih objekata u projektu obavljene na ispravan način?

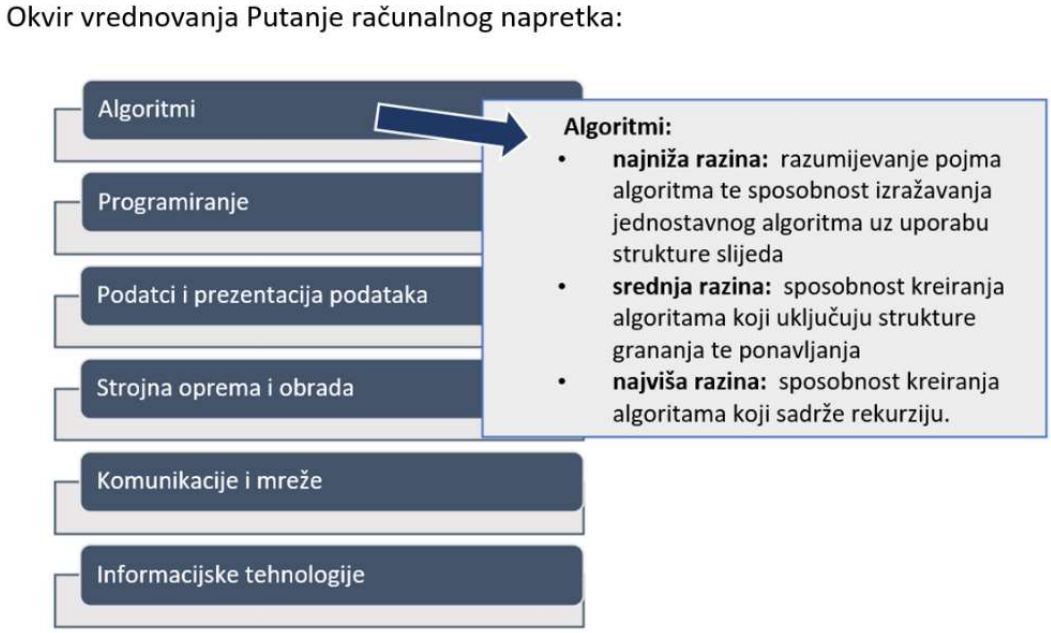
Slika 16: Koncepti vrednovanja računalnog razmišljanja uključeni u alat Dr. Scratch

Ovaj pristup vrednovanju primjer je formativno-iterativnog pristupa koji nudi brzu povratnu informaciju. Dr. Scratch je koristan alat jer omogućuje korisniku-učeniku promišljanje o algoritamskom rješenju te nudi korisne savjete kojima se rješenje može unaprijediti (Romero i sur., 2017). Ipak, pri vrednovanju računalnih igara ovim alatom neka istraživanja ističu nemogućnost alata da pri vrednovanju vodi računa i o faktorima kao što su npr. vrijeme uloženo u kreiranje igre, broj blokova koda, broj nepoznanica koji se koristi i sl. te predlažu kombiniranje alata s nekim drugim oblikom kvalitativnog vrednovanja (Hoover i sur., 2016).

2.7.2. Vrednovanja računalnog razmišljanja uz djelomična programiranja ili bez primjene programiranja, odnosno računala

Zanimljiv pristup podržavanju rada učitelja u razrednom okružju pri poučavanju računalnog razmišljanja predstavlja okvir vrednovanja Putanje računalnog napretka (engl. Computing Progression Pathways). Takvim pristupom identificiraju se glavna područja računalnog znanja te nude pokazatelji povezanih razina usvojenosti odabranih područja (Slika 17) (Dorling i Walker, 2014).

Umjesto stavljanja naglaska na samu vještinu i sadržaj koji se poučava, koristi se opisani okvir vrednovanja te predlaže način rada koji uvijek započinje s definiranjem teme nastavne jedinice te propitivanjem „Zašto učimo odabrane sadržaje/vještine? (faza Zašto)“ (Curzon i sur., 2014). Na temelju okvira Putanje računalnog napretka identificiraju se ishodi učenja koji učenicima mogu omogućiti razumijevanje te identificiranje koncepata računalnog razmišljanja koji su povezani s navedenim ishodima učenja (faza Kako).



Slika 17: Okvir vrednovanja Putanje računalnog napretka

Odabiru se tehnike kojima se odabrani koncepti računalnog razmišljanja nastoje uključiti u proces poučavanja te na kraju provodi vrednovanje (faza Što). Iako su autori ovog pristupa ponudili primjere navedenog okvira kroz niz nastavnih aktivnosti, kritike opisanog pristupa

naglašavaju nedovoljan naglasak na dublje razumijevanje u fazama Kako? i Zašto? (Rodriguez i sur., 2017).

U raspravama o pristupima vrednovanju koji ne uključuju programiranje ili su smješteni izvan programerskog pa čak i informatičkog konteksta, posebno mjesto zauzimaju zadaci izazova Dabar. Svaki takav zadatak predstavlja zanimljivu priču, odnosno zagonetku koja se oslanja na odabrani ključni koncept računalnog razmišljanja, osmišljen za promicanje računalnog razmišljanja i informatike kod učenika svih dobnih skupina.

Dabar, glavni lik svakog zadatka, pokušava riješiti zadani problem ili zagonetku, pri čemu svaki zadatak ima različit kontekst (priču). Svake godine stručnjaci informatičkog obrazovanja pripremaju i distribuiraju zadatke za izazov Dabar, imajući u vidu da svaki zadatak mora sadržavati koncepte računalnog razmišljanja ili informatike primjerene dobi učenika. Iako postoje dokazi da su zadaci izazova Dabar učinkovito sredstvo u razvoju računalnog razmišljanja (Dagiene i Stupiriene, 2016; Araujo i sur., 2017; Curran, 2019), danas ih više spominjemo kao jedan od načina vrednovanja računalnog razmišljanja.

U posljednje se vrijeme sve više raspravlja o učinkovitosti vrednovanja koncepata računalnog razmišljanja kroz zadatke izazova Dabar. Takvo vrednovanje još uvijek se smatra zahtjevnim (Dagiene i Stupiriene, 2016). Postoje istraživanja u kojima se raspravlja o tome mogu li se koncepti računalnog razmišljanja učinkovito ocjenjivati pomoću zadataka izazova Dabar (Araujo i sur., 2017), s obzirom na to da još uvijek nema dovoljno odgovarajućih istraživanja koja bi podržala trenutnu praksu razvoja dobrih zadataka.

Danas je moguće pronaći više objavljenih istraživanja o vrednovanju računalnog razmišljanja koja se provode primjenom računala ili uz papir i olovku, a mogu se koristiti u različitim kontekstima. Učinkovitost nekih pristupa još uvijek se istražuje (Werner i sur., 2012; Romero i sur., 2017; González, 2015).

Sve navedene primjere vrednovanja računalnog razmišljanja možemo smatrati i predstavnicima pojedinih pristupa vrednovanju. Pristupi se razlikuju u vrstama grafičkih okružja i programskih jezika koji se primjenjuju u poučavanju i vrednovanju, ali i u tome zagovaraju li računalno vrednovanje ili vrednovanje bez uporabe računala, npr. putem intervjua, pisanim uradcima i provjerama ili nekim drugim kvalitativnim metodama.

Iako su se neki pristupi pokazali kao mogući primjeri dobre prakse, oni su uglavnom relativno novi i nedovoljno istraženi te je potrebno njihovo daljnje ispitivanje kako bi se utvrdila njihova valjanost i pouzdanost.

3. MODEL VREDNOVANJA RAČUNALNOG RAZMIŠLJANJA TEMELJEN NA DOKAZU

Odgovarajuća pedagoška praksa, koja naglašava konstruktivistički pristup učenju i stavljanje učenika u središte procesa učenja, trebala bi razviti kompetencije poput neovisnosti, samopouzdanja, odgovornosti i poduzetništva. Iskustva učenja moraju biti takva da potiču učenika na aktivno sudjelovanje i ulaganje velikog napora u svom kreativnom radu te da se timskim radom i suradnjom stvara snažna motivacija za učenje. Kurikul nastavnog predmeta Informatika koji se temelji na ishodima učenja umjesto unaprijed definiranom nastavnom sadržaju omogućava realizaciju učenja i poučavanja usmjerenog prema učeniku te razvoj njegovih potencijala. Takvim pristupom pruža se fleksibilnost u poučavanju i daje sloboda učiteljima u osmišljavanju procesa učenja i poučavanja.

U anketi u kojoj je sudjelovalo 623 hrvatska učitelja Informatike čak 42,88% učitelja odabralo je upravo vrednovanje kao temu računalnog razmišljanja koju najlošije poznaje. To ukazuje na potrebu stvaranja okvira vrednovanja računalnog razmišljanja usklađenog s predmetnim kurikulom te primjerenog provedbi u stvarnom razrednom okružju (Ministarstvo znanosti i obrazovanja, 2018).

Ovo istraživanje predstavit će jedan model vrednovanja teško mjerljivih koncepata računalnog razmišljanja koji je temeljen na dokazima (ECD), ima uporište u ishodima učenja i novom kurikulu predmeta Informatika, te stavlja naglasak na konstruktivistički pristup učenju. Takav model vrednovanja koji je temeljen na dokazu nudi dobro strukturiran (usklađen s temeljnim dokumentima), a opet dovoljno fleksibilan (neovisan o programskom jezicima koji se koriste procesu poučavanja) okvir za stvaranje vrednovanja u kojima je dokaz o kompetenciji osnova za izgradnju kvalitetnog zadatka vrednovanja.

Model vrednovanja prvenstveno ističe složenost znanja i vještina, ali uvažava i neke druge značajke koje se razmatraju u tom procesu (osnovno ICT znanje, mogući radni proizvodi i zapažanja, promjenjiva obilježja povezana s vrednovanjem). Predstavlja se model stvaranja kvalitetnih zadataka, dokaza i mjerenja koji će učenicima biti zanimljivi i poticajni usprkos činjenici da često uključuju veoma zahtjevne koncepte.

Model predlaže uzorke dizajna koncepata računalnog razmišljanja za izradu odgovarajućih zadataka vrednovanja kojima se konkretiziraju pojedine prakse, znanja i vještine povezane s odabranim konceptom računalnog razmišljanja.

Za razliku od PACT projekta u kojem je naglasak na vrednovanju koncepata računalnog razmišljanja u interdisciplinarnom pristupu, predstavit će se model koji je u potpunosti usklađen s predmetnim kurikulom, prilagođen primjeni u stvarnom razrednom okružju te omogućava vrednovanje neovisno o programskom jeziku ili okružju koji se koristi u procesu poučavanja.

U sljedećim poglavljima predstavit će se model vrednovanja u skladu sa slojevima pristupa temeljenog na dokazima: analizom i modeliranjem područja vrednovanja, opisom konceptualnog okvira vrednovanja, primjeni vrednovanja te objavom rezultata vrednovanja.

3.1. Analiza područja vrednovanja

Analiza područja vrednovanja ima za cilj pronaći i istražiti sve relevantne materijale koji se tiču ciljanih ishoda učenja. Svaka provedba vrednovanja u školi mora biti usklađena s relevantnim dokumentima koji se odnose na proces poučavanja i vrednovanja: „Na svim su razinama kurikulnih dokumenta definirana i razrađena odgojno-obrazovna očekivanja ili ishodi kojima su jasno određene kompetencije (znanja, vještine i stavovi) koje učenici trebaju steći odgojno-obrazovnim procesom te je opisan napredak koji trebaju ostvariti učenjem i poučavanjem. U nacionalnim kurikulima nastavnih predmeta, osim što je uz svaki odgojno-obrazovni ishod određena razrada ishoda, opisane su razine usvojenosti te dane preporuke za ostvarivanje ishoda. Jasno definiranje ishoda i pripadajućih razina njihove usvojenosti omogućuje kriterijsko vrednovanje s obzirom na to u kojoj je mjeri učenik usvojio očekivane odgojno-obrazovne ishode, a umanjuje potrebu za normativnim vrednovanjem koje se temelji na međusobnome uspoređivanju učenika u određenim skupinama” (Ministarstvo znanosti i obrazovanja, 2019c).

Ishodi učenja razvijenog okvira vrednovanja temelje se na nekoliko dokumenata, ali prvenstveno na Hrvatskom nacionalnom obrazovnom standardu (HNOS) (Ministarstvo znanosti i obrazovanja, 2006), standardima definiranim od strane Svjetske udruge učitelja

informatike, CSTE Standards (CSTA, 2016) te na Kurikulumu nastavnog predmeta Informatika (Ministarstvo znanosti i obrazovanja, 2018) (Slika 18).

Slika 18: Analiza područja

HNOS definira način na koji je predmet Informatika uključen u hrvatsko osnovno, srednje i visoko obrazovanje dok kurikul predmeta Informatika te standardi Svjetske udruge učitelja informatike definiraju obrazovne ishode. Hrvatski kurikul predmeta Informatika nadalje razrađuje obrazovne ishode za svaku razinu obrazovanja (Tablica 4) uz detaljnu specifikaciju pojedinih razina usvojenosti za svaki obrazovni ishod (Tablica 5).

U sklopu predmetnog kurikula definirano je područje aktivnosti svake domene te ishodi koje ona uključuje: „Računalno razmišljanje temeljni je pristup kojim se razvija sposobnost rješavanja problema i programiranja. Pritom je naglasak na usvajanju procesa stvaranja aplikacije od početne ideje do konačnoga proizvoda, a ne isključivo na usvajanju sintakse i semantike programskoga jezika.

Aktivnosti i sadržaji ishoda iz domene Računalno razmišljanje i programiranje razvijaju inovativnost, stvaralaštvo i poduzetnost te daju vrijedna znanja koja se mogu ugraditi u budući profesionalni život.“ (Ministarstvo znanosti i obrazovanja, 2018).

Tablica 4: Ishodi domene Računalno razmišljanje i programiranje za učenike 5. i 6. razreda - Kurikulum nastavnog predmeta Informatika

Razred/uzrast	Ishodi – domena Računalno razmišljanje i programiranje
5. razred/11 god.	Nakon pete godine učenja predmeta Informatika u domeni Računalno razmišljanje i programiranje učenik: B. 5. 1 koristi se programskim alatom za stvaranje programa u kojemu se koristi ulaznim i izlaznim vrijednostima te ponavljanjem B. 5. 2 stvara algoritam za rješavanje jednostavnoga zadatka, provjerava ispravnost algoritma, otkriva i popravlja pogreške.
6. razred/12 god.	Nakon šeste godine učenja predmeta Informatika u domeni Računalno razmišljanje i programiranje učenik: B. 6. 1 stvara, prati i preuređuje programe koji sadrže strukture grananja i uvjetnoga ponavljanja te predviđa ponašanje jednostavnih algoritama koji mogu biti prikazani dijagramom, riječima govornoga jezika ili programskim jezikom B. 6. 2 razmatra i rješava složeniji problem rastavljajući ga na niz pod-problema.

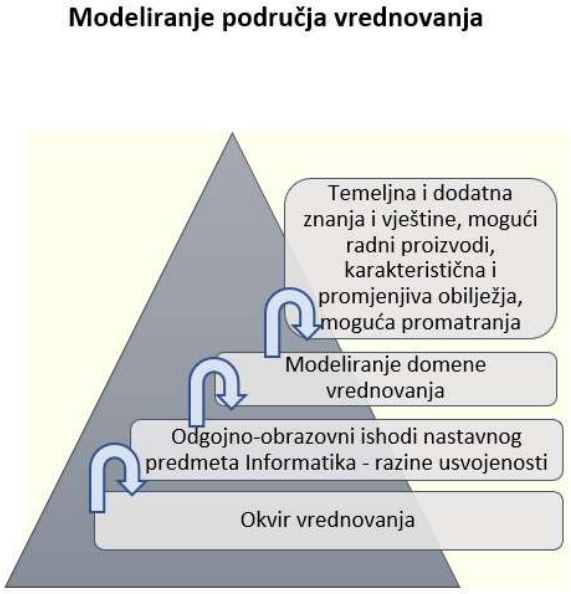
Tablica 5: Detaljna razrada ishoda B.6.1. kurikula predmeta Informatika

Oznaka ishoda	B.6.1
Razred	6.
Domena	Računalno razmišljanje i programiranje
Ishod	B.6.1. Nakon šeste godine učenja predmeta Informatika u domeni Računalno razmišljanje i programiranje učenik stvara, prati i preuređuje programe koji sadrže strukture grananja i uvjetnog ponavljanja te predviđa ponašanje jednostavnih algoritama koji mogu biti prikazani dijagramom, riječima govornog jezika ili programskim jezikom
Zadovoljavajuća razina	Učenik opisuje problem te prepoznaje ulazne i izlazne vrijednosti te algoritamske strukture koje se upotrebljavaju za rješavanje problema, samostalno planira i slaže niz uputa (naredbi) kao rješenje problema primjenom samo algoritamske strukture slijeda i ponavljanja (s određenim brojem ponavljanja).
Dobra razina	Učenik samostalno ili uz pomoć učitelja analizira zadani problem te predlaže koje algoritamsko rješenje. Rješenje problema prikazuje riječima govornoga jezika, dijagramom ili naredbama programskoga jezika te samostalno planira i slaže niz uputa kao rješenje problema primjenom algoritamskih struktura slijeda i grananja.
Vrlo dobra razina	Učenik samostalno predlaže program/algoritam kao rješenje problema, predviđa ponašanje algoritma te provjerava ispravnost algoritma prateći njegovo ponašanje ili izvođenjem programa sa zadanim primjerima. Samostalno ili uz pomoć učitelja slaže niz uputa za rješenje problema koristeći se uvjetnim ponavljanjem.
Iznimna razina	Učenik samostalno stvara program/algoritam kao rješenje problema koje uključuje niz uputa (naredbi) primjenom svih algoritamskih struktura, predviđa

	odgovarajuće ulazne (testne) primjere te kritički provjerava ispravnost rješenja i prema potrebi preuređuje svoje rješenje.
Razrada ishoda	Učenik interpretira problem te prepoznaje ulazne vrijednosti i algoritamske strukture koje se upotrebljavaju za rješavanje problema, samostalno planira i slaže niz uputa (naredbi) kao rješenje problema primjenom algoritamskih struktura slijeda, grananja i ponavljanja. Učenik samostalno ili uz pomoć učitelja analizira zadani problem te predlaže neko algoritamsko rješenje, rješenje problema prikazuje dijagramom, riječima govornoga jezika ili naredbama programskoga jezika. Predviđa ponašanje algoritma te provjerava ispravnost algoritma prateći njegovo ponašanje (olovkom) ili testiranjem programa (algoritma) nekim ulaznim vrijednostima (na računalu). Učenik predviđa odgovarajuće ulazne (testne) primjere te kritički provjerava ispravnost rješenja i prema potrebi preuređuje svoje rješenje.
Preporuke	Pokazati praćenje ponašanja algoritma jednostavnim pokaznim primjerima (učitelj se koristi svojim primjerima ili postojećim demosadržajima, npr. online videosadržajima, alatima vizualizacije programa (npr. http://www.pythontutor.com/visualize.html#mode=display, https://scratch.mit.edu/help/videos/…) Angažirati učenike u istraživanju ponašanja nekih algoritama samostalnim uređivanjem i mijenjanjem pokaznih primjera. Samostalno ili u parovima učenici izrađuju niz uputa (naredbi) kao rješenje nekog problema. Učenici samostalno ili uz pomoć učitelja rješavaju jednostavne probleme koji upotrebljavaju ulazne vrijednosti i algoritamske strukture slijeda, grananja i ponavljanja, npr. ispisivanje određenoga niza brojeva ili brojeva s određenim svojstvima (parni, pozitivni i sl.), računanje s nizom ulaznih vrijednosti, traženje najveće/najmanje od triju vrijednosti (najviše tri vrijednosti), stvaranje scenarija koji uključuju dijaloge među likovima (objektima) te ponavljanje izvršavanja nekih aktivnosti, npr. kretanje likova, mijenjanje različitih obilježja likova (objekata), korištenje koordinatnim sustavom s cjelobrojnim koordinatama, računanje opsega, površine trokuta i četverokuta, računanje postotnoga iznosa. Zanimljivi sadržaji mogu se pronaći na mrežnim stranicama Code week, Hour of code, App studio, Code Academy i sl.
Poveznice:	Matematika: ishodi A.6.1, B.6.1, D.6.1, D.6.2, D.6.3, D.6.5.

3.2. Modeliranje područja vrednovanja

Modeliranje područja vrednovanja uključuje proces identificiranja ključnih elementa kojima se opisuje područje vrednovanja. Svaki proces vrednovanja koji se primjenjuje u odgojno-obrazovnom procesu osnovnih i srednjih škola RH propisan je temeljnim dokumentima koje propisuje MZO (Slika 19). Pravilnik o načinima, postupcima i elementima vrednovanja učenika u osnovnim i srednjim školama (Ministarstvo znanosti i obrazovanja, 2019c) definira proces vrednovanja, načine praćenja te metode i elemente koji se pri tom procesu koriste. Područje vrednovanja opisuju razine usvojenosti odgojno-obrazovnih ishoda predmetnog kurikula (Ministarstvo znanosti i obrazovanja, 2018).



Slika 19: Modeliranje područja vrednovanja

Prema ECD-u pristupu, modeliranje domene (područja) vrednovanja organizirano je u pet slojeva: osnovna i dodatna znanja, vještine i značajke, mogući radni proizvodi, karakteristična i promjenjiva obilježja te moguća promatranja. Svaki sloj opisan je smjernicama/pitanjima koja olakšavaju primjenu modela za stvaranje jedinki vrednovanja (Tablica 6).

Tablica 6: Modeliranje područja vrednovanja prema ECD pristupu

Smjernice za modeliranje područja vrednovanja	
Osnovna znanja, vještine i obilježja	Istaknuti osnovne algoritamske strukture uključene u zadatke vrednovanja (slijed, grananje, ponavljanje). Istaknuti ključne sposobnosti i razumijevanja uključene u zadatke vrednovanja npr.: • osnovna namjena/zadatak računalnog rješenja • stvaranje novih rješenja nadograđivanjem postojećih • stvaranje cjelovitog rješenja • prepoznavanje manjih dijelova (cjelina) zadanog rješenja redizajniranje rješenja
Dodatna znanja, vještine i obilježja	Dodatna znanja, vještine i obilježja prepoznati uz pomoć sljedećih pitanja: • Je li neophodno poznavanje programiranja (neovisno o programskom jeziku ili okružju)? • Je li neophodno poznavanje/rad u nekom zadanom programskom jeziku? • Je li neophodna vještina pretraživanja interneta? • Je li neophodna vještina preuzimanja sadržaja s mrežnih stranica? • Zahtijeva li se posjedovanje ili rukovanje nekih osobnim korisničkim računom? • Zahtijeva li se posjedovanje nekih predznanja i vještina iz predmeta Informatika? Kojih? • Zahtijeva li se posjedovanje znanja i vještina iz nekih drugih predmeta/disciplina? Kojih?
Mogući radni proizvodi	Svi oblici radnih proizvoda koji mogu biti uključeni u zadatke vrednovanja: • računalno rješenje • opis računalnog rješenja • predviđanje izvršavanja računalnog rješenja • usporedba računalnih rješenja • usporedba dijela računalnih rješenja • usporedba računalnih strategija • grafički prikazi kojima se opisuje algoritamsko rješenje • vrednovanje ispravnosti računalnog rješenja • vrednovanje ispravnosti dijela računalnog rješenja
Karakteristična obilježja	Obilježja koja su značajna i zajednička jedinkama vrednovanja: • način prikazivanja računalnog rješenja • kontekst koji se koristi u zadatcima vrednovanja, • vrsta grafičkih prikaza kojima se postiže konkretizacija problema te olakšava razumijevanje, • vrsta specifičnih grafički prikazi npr. tablice za prikaz odabranih obilježja, vrijednosti, slučajeva, testnih primjera i sl. • povezanost zadataka vrednovanja.
Promjenjiva obilježja	Istaknuti obilježja jedinki vrednovanja koja se mogu mijenjati u ovisnosti o ishodu vrednovanja npr.: • jesu li uključene različite razine rješavanja problema (vrednovanje djelomičnog ili/i potpunog rješenje)?

	• jesu li uključeni različiti načini prikazivanja rješenja? • je li uključeno vrednovanje rješenja od strane učenika npr. usporedba dvaju rješenja, vrednovanje ispravnosti rješenja i sl.? • uključuju li jedinke više koncepata računalnog razmišljanja? Takvim izmjenama kreiraju se jedinke različite težine ili mjerenje nekih dodatnih znanja.
Moguća promatranja	Sva promatranja kojima se vrednuje odabrani koncept/koncepti računalnog razmišljanja. Promatranja opisati s obzirom na dokaze koji će se koristiti u postupcima vrednovanja. Smjernice za odabir promatranja: • razlikuju li odabrani dokazi razinu adresiranja problema? • razlikuju li odabrani dokazi složenost računalnog rješenja? • prepoznaju li dokazi odgovarajući primjenu algoritamskih struktura u rješenju? • prepoznaju li dokazi različite stupnjeve uspješnog pronalaska pogrešaka u procesu popravljanje rješenja? • prepoznaju li dokazi različite stupnjeve popravljanja pogrešaka?

Primjer modeliranja područja vrednovanja prikazan je na razvoju alata vrednovanja računalnog razmišljanja za učenike šestog razreda osnovne škole (Tablica 7).

Tablica 7: Primjer modeliranja područja vrednovanja računalnog razmišljanja 6.razreda (OŠ)

Osnovna znanja, vještine i obilježja	Razumijevanje da se računalno rješenje može dizajnirati za više namjera. Sposobnost stvaranja rješenja kombiniranjem manjih dijelova koji vode prema rješenju problema u cjelini. Sposobnost naknadnog uređivanja dizajna računalnog rješenja. Sposobnost kodiranja kompletnog rješenja. Sposobnost uporabe algoritamskih struktura (slijed, grananje i ponavljanje) u rješenju.
Dodatna znanja, vještine i obilježja	Znanje pojedinog programskog jezika (nije neophodno). Prijava na mrežne stranice s osobnim korisničkim podatcima (prijava s računom AAI). Osnovne vještine pregledavanja mrežnih stranica (otvaranja zadane mrežne stranice, osnovna navigacija unutar zadanog mrežnog mjesta).
Mogući radni proizvodi	Računalno rješenje. Usporedba višestrukih računalnih rješenja ili strategija. Opis ili objašnjenje računalnog rješenja. Predviđanje rezultata izvršavanja računalnog rješenja.
Karakteristična obilježja	Predstavljanje računalnog rješenja grafičkim prikazima i algoritamskim rješenjem pisanim jezikom sličnim govornom jeziku koji strukturom podsjeća na pseudo jezik (engl. pseudocode). Svaki zadatak vrednovanja ima zajednički kontekst tj. zagonetku koja se razvija od početnog do završnog zadatka prema principu od najlakšeg prema najtežem).

	U svakom zadatku vrednovanja glavni lik zagonetke nailazi na neki problem koji treba riješiti kako bi uspješno prošla kroz labirint. Zadatcima se opisuju, provjeravanju, korigiraju postojeći te kreiraju novi načini kretanja po labirintu.
Promjenjiva obilježja	Razine računalnog rješenja. Reprezentacija računalnog rješenja. Je li učenik došao do rješenja, vrednovao rješenje ili usporedio računalna rješenja?
Moguća promatranja	Stupanj do kojeg računalno rješenje adresira problem. Razina složenosti računalnog rješenja. Ispravnost računalnog rješenja. Primjerenost uporabe algoritamskih struktura u rješenju. Stupanj do kojeg proces popravljanja pronalazi i/ili popravlja pogreške.

Modeliranjem područja vrednovanja opisuju se i uzorci dizajna koji će se koristiti u vrednovanju. Uzorci dizajna sadrže sve ili samo neke konstrukte (znanja, vještine, praksu) koje želimo vrednovati i mogu se koristiti kao dokazi (znanja, prakse) ili poslužiti kao osnova za izradu dokaza. Uzorci se mogu koristiti i kao aktivnosti ili zadaci koji imaju za cilj potaknuti demonstraciju određenih kompetencija. Dovoljno su općeniti da mogu usmjeriti vrednovanje neovisno o načinu njegove provedbe (vrednovanje na računalu, tradicionalno vrednovanje papir-olovka, vrednovanje simulacijom, igrom i sl.). Sljedeća tablica prikazuje uzorke dizajna koji su uključeni ovaj model modela vrednovanja (Tablica 8).

Tablica 8: Tablica uzoraka dizajna modela vrednovanja računalnog razmišljanja

Koncept	Uzorak dizajna (oznaka uzorka)
Apstrakcija	(A1) opisati osnovna obilježja problema (A2) prepoznati ograničenja zadanog problema (A3) vrednovati logičke izraze (A4) koristiti se logičkim operatorima (A5) kreirati logički izraz za zadani uvjet (problem) (A6) razlikovati konstantne i promjenjive veličine u algoritmu/rješenju (A7) primijeniti varijablu u algoritmu za praćenje promjenjivih obilježja problema (A8) definirati i/ili pratiti izmjenu vrijednosti varijable u algoritmu/rješenju
Algoritamsko razmišljanje	(AL1) pratiti i predvidjeti izvršavanje algoritamskog rješenja koje ne sadrži petlje (AL2) predvidjeti izvršavanje algoritamskog rješenja koje sadrži petlje (AL3) prepoznati/identificirati osnovno obilježje petlje (ponavljanje) te način zaustavljanja rada petlje (AL4) prepoznati/identificirati dijelove algoritma koji sadrže odluke

	(AL5) prepoznati/opisati/razlikovati kako djeluje jednostavna i složena struktura grananja (AL6) stvoriti novi algoritam (AL7) nadograditi algoritam zbog uočene pogreške ili ispunjanja zahtjeva zadanog problema
Dekompozicija	(D1) prepoznati dijelove zadanog problema koji je jednostavniji za rješavanje ili nam je rješenje tog dijela već poznato (D2) razlučiti pod-cjeline algoritamskog rješenja (D3)potprogramom prikazati pod-cjelinu algoritamskog rješenja
Vrednovanje	(V1) identificirati valjane (i nevaljane) ulazne vrijednosti algoritamskog (računalnog) rješenja (V2) identificirati ulaznih vrijednosti kojima se provjeravaju ograničenja algoritamskog (računalnog) rješenja (V3) identificirati izvor run-time pogreške (V4) objasniti uzrok run-time pogreške (V5) opisati sustavnu metodu prepoznavanja pogrešaka u radu algoritma/rješenja
Generalizacija	(G1) objasniti kako se uporabom parametara može poopćiti zadano rješenje (G2) prepoznati uzorak (među podatcima, dijelom koda, obilježjima..) analizom sličnih problema ili procesa (G3) poopćiti prepoznati uzorak u cilju stvaranja rješenja primjenjivog na slične probleme ili procese

U sljedećem poglavlju, kroz opis konceptualnog okvira modela vrednovanja prikazat će se i primjer uporabe uzoraka dizajna za razvoj jednog alata vrednovanja.

3.3. Konceptualni okvir modela vrednovanja

Vrednovanje računalnog razmišljanja uvelike ovisi o kontekstu unutar kojeg se provodi samo vrednovanje. Važno je promišljati je li za provođenje vrednovanja računalnog razmišljanja nužno koristiti neki programski jezik ili okružje. Pitanje povezanosti računalnog razmišljanja i programiranja mora se definirati prema kontekstu primijenjenog vrednovanja. Postoje različiti pristupi uključivanju programiranja u nastavni proces, a time i u proces vrednovanja računalnog razmišljanja. Razlikujemo ih prema ulozi koju imaju programiranje i računalno razmišljanje u kurikulu predmeta (Astrachan i sur., 2009). U ovom istraživanju vrednovanje računalnog razmišljanja provodi se pristupom koji nije prilagođen pojedinom programskom jeziku ili okružju. Takvim pristupom nastoji se omogućiti šira primjena alata vrednovanja te realizirati vrednovanje koje ističe koncepte i praksu koja se vrednuje, a ne sintaksu pojedinog programskog jezika ili okružja.

Prema ECD-u pristupu (Hendrickson i sur., 2013), osnovni zadatak okvira vrednovanja je pomoći dizajneru alata pri odabiru odgovarajućih modela zadataka vrednovanja. Svaki dizajner nekog alata vrednovanja trebao bi provjeravati, a na kraju i potvrditi svoj rad odgovarajućim pitanjima koji se odnose na relevantnost, specifičnost te skalabilnost čestica (zadataka vrednovanja), istovremeno vodeći računa o odgovarajućim statističkim parametrima odabranih čestica. Okvir vrednovanja trebao bi prvenstveno pružiti informacije o dokazima znanja koje želimo vrednovati kod učenika, nadalje informacije o modelu učenika i modelu zadataka, informacije o zapaženim karakteristikama, modelima mjerenja te eventualnim specifikacijama ispitivanja.

3.3.1. Model učenika

U ovom okviru vrednovanje računalnog razmišljanja provodi se kroz pristup koji nije ovisan o programskom alatu ili okruženju. Ovaj se pristup može koristiti za vrednovanje usvojenih ishoda učenja u stvarnim nastavnim situacijama u odabranoj fazi obrazovanja. Upravo neovisnost programskog alata ili okruženja trebala bi omogućiti širu primjenu u procesu vrednovanja računalnog razmišljanja. Također, neovisnost vrednovanja o programskim jezicima koji se uglavnom koriste u redovnoj nastavnoj praksi u RH omogućava naglašavanje koncepata, a ne sintakse određenog programskog alata ili mogućnosti okružja. Iz istog je razloga razvijeni alat vrednovanja moguće koristiti i s učenicima koji imaju malo ili gotovo nimalo znanja programiranja.

3.3.2. Model zadataka

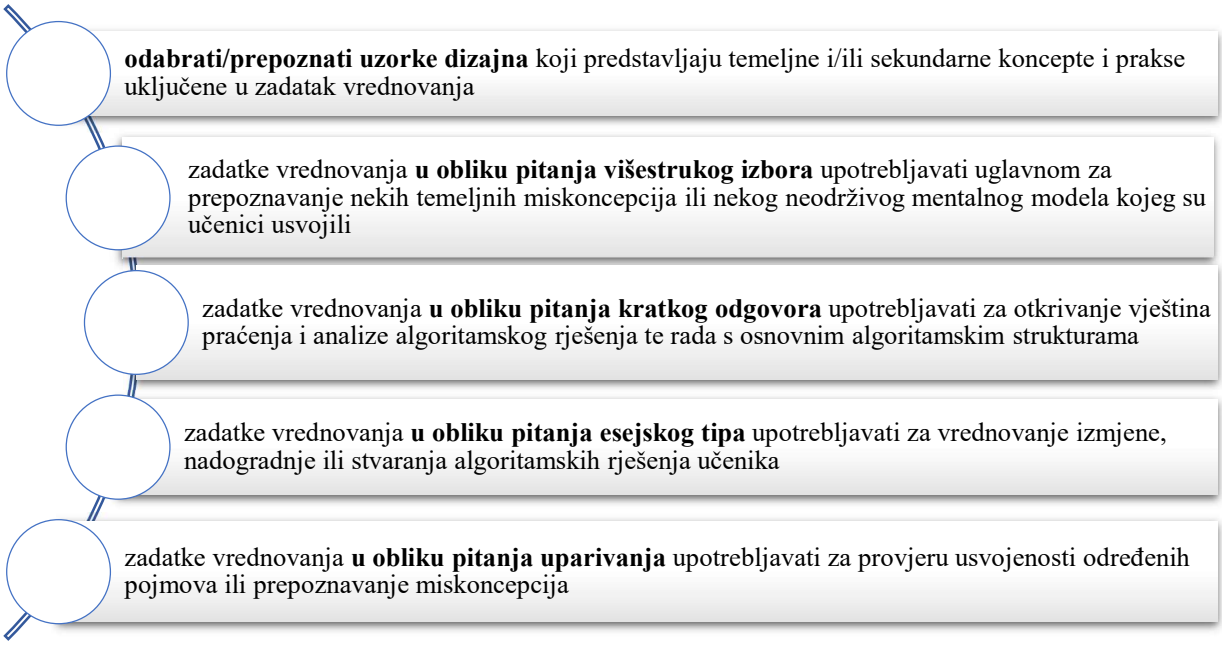
Pri izradi okvira vrednovanja, od iznimne je važnosti prepoznati moguće radne proizvode (artefakte) te opažanja koja će se koristiti pri vrednovanju kao dokazi učeničkog znanja. Pri tom procesu od velike pomoći mogu biti odgovori na neka osnovna pitanja koja se odnose na sam proces vrednovanja:

- *Treba li radni proizvod učenika biti računalno rješenje, usporedba više računalnih rješenja ili čak opis ili objašnjenje nekog rješenja?*
- *Treba li pri vrednovanju razmatrati obilježja kao što su složenost, ispravnost (korektnost) ili učinkovitost računalnih rješenja?*
- *Treba li u rješenju prepoznati odgovarajuću uporabu algoritamskih struktura?*
- *Treba li analizirati u kojoj mjeri prijedlog rješenja zaista rješava zadani problem, odnosno treba li analizirati u kojoj mjeri učenik uspješno pronalazi i ispravlja pogreške?*

Odgovori na navedena pitanja moraju ovisiti o dobi učenika, o domeni i cilju vrednovanja te stvarnoj situaciji u učionici u kojoj se vrednovanje primjenjuje. Iz tih razloga, ova pitanja predstavljaju osnovu za stvaranje budućih čestica (zadataka) vrednovanja. U ovom su istraživanju kreirani zadaci koji su prilagođeni učenicima s malim ili gotovo nikakvim programskim vještinama. U nastojanju da se potakne motivacija i interes učenika te njihova sposobnost, glavni lik (pčelica Maja) rješava neki problem (Lee i Ko, 2011).

Svaki zadatak krije u sebi vrednovanje jednog ili više koncepata računalnog razmišljanja. Ti koncepti, skriveni u zagonetkama, odabrani su i usklađeni s očekivanim predmetnim ishodima (Tablica 4, Poglavlje 3.1.) i detaljnom analizom domene (Poglavlje 3.1.).

Predloženim pristupom vrednovanju nastoji se stvoriti model koji će se dovoljno odmaknuti od ograničenja određenog programskog jezika te omogućiti vrednovanje koncepata računalnog razmišljanja, a ne sposobnosti kodiranja u zadanom programskom jeziku. Model oblikovanja zadataka vrednovanja prikazan je sljedećom slikom (Slika 20).



Slika 20: Oblikovanje zadataka vrednovanja – model zadataka

Sljedeća tablica prikazuje strukturu razvijenog alata vrednovanja za potrebe istraživanja (Tablica 9). Alat vrednovanja sadrži deset čestica. Za svaku česticu vrednovanja istaknuta je vrsta pitanja koja se koristila u alatu vrednovanja te temeljni koncepti i prakse računalnog razmišljanja koji su uključeni u vrednovanje. Pri analizi koncepata i praksi računalnog razmišljanja korištenih u pojedinom zadatku posebno su istaknuti oni koncepti i prakse koje se smatraju temeljnim, te posebno oni koji se smatraju sekundarnim za odabrani zadatak.

Gotovo je nemoguće smatrati da se pri rješavanju nekog zadatka predstavljenog algoritamskim rješenjem može reći da zadatak ne vrednuje analiziranje i praćenje algoritma. No, kao što se u matematici, na primjer pri vrednovanju rješavanja linearne jednadžbe s jednom nepoznanicom, sekundarno oslanjamo na vještinu računanja s osnovnim matematičkim operacijama, tako ćemo i pri vrednovanju računalnog razmišljanja prepoznati koji su to koncepti te prakse računalnog razmišljanja koje se primarno vrednuju odabranim zadatkom, a koji su to koncepti i prakse sekundarno prisutni u istom zadatku vrednovanja. Takav prikaz nužan je i za daljnju analizu kvalitete provedenog vrednovanja i kreiranje modela vrednovanja.

Nadalje, za svaku jedinku vrednovanja istaknuta je oznaka uzorka dizajna koja je uključena u vrednovanje. Posebno su istaknute oznake uzoraka dizajna temeljnih i sekundarnih koncepata i praksi računalnog razmišljanja.

Tablica 9: Struktura alata vrednovanja primijenjenog u istraživanju

Redni broj zadatka	Vrsta pitanja	Temeljni koncepti i prakse računalnog razmišljanja uključeni u zadatak	Sekundarni koncepti i prakse računalnog razmišljanje uključeni u zadatak	Oznaka uzorka dizajna Temeljni (sekundarni)
Zadatak1	Pitanje kratkog odgovora	Razumijevanje problema (apstrakcija)	Analiziranje i praćenje algoritma (Algoritamsko razmišljanje) Rad sa strukturama slijeda, grananja te ponavljanja (Algoritamsko razmišljanje)	A1+A3 (AL2+AL3+AL4)*
Zadatak2	Pitanje višestrukog odgovora	Identificiranje ključnih obilježja (ograničenja) problema (Apstrakcija)	Predviđanje ponašanja algoritma (Algoritamsko razmišljanje)	A2+A3 (AL2+AL3)*
Zadatak3	Esejsko pitanje (rad s algoritamskim rješenjem)	Rad s logičkim vrijednostima (Apstrakcija)	Primjena strukture grananja (Algoritamsko razmišljanje)	A3+A5 (AL2+AL3+AL4+ AL5+AL7)*
Zadatak4	Pitanje višestrukog odgovora	Rad s različitim razinama apstrakcije (Apstrakcija)	Razumijevanje problema (Apstrakcija)	A1+A2 (AL2+AL3+AL4)*
Zadatak5	Pitanje višestrukog odgovora	Rad s grananjem (Algoritamsko razmišljanje)	Analiza i praćenje algoritma (Algoritamsko razmišljanje)	AL4+AL5 (AL1)*
Zadatak6	Pitanje višestrukog odgovora	Predviđanje ponašanja algoritma (Algoritamsko razmišljanje)	Analiza i praćenje algoritma (Algoritamsko razmišljanje)	AL2 (AL3)*
Zadatak7	Esejsko pitanje (rad s algoritamskim rješenjem)	Uređivanje i nadogradnja algoritamskog rješenja (Algoritamsko razmišljanje) Razumijevanje problema (Apstrakcija)	Analiza i praćenje algoritma (Algoritamsko razmišljanje) Rad s varijablama i izlazom (Apstrakcija)	A2+AL7 (A6+A7+AL2+ AL3)*
Zadatak8	Pitanje višestrukog odgovora	Rastavljanje problema na jednostavnije već poznate probleme (Dekompozicija)	Analiza i praćenje algoritma (Algoritamsko razmišljanje)	D1 (AL2)*
Zadatak9	Esejsko pitanje (rad s algoritamskim rješenjem)	Stvaranje novih rješenja na temelju poznatih rješenja problema (Algoritamsko razmišljanje, dekompozicija)	Rad s različitim razinama apstrakcije (Apstrakcija) Rad s varijablama i izlazom (Apstrakcija)	D3 (A1+A3+A7)*

Zadatak10	Pitanje kratkog odgovora	Predviđanje ponašanja algoritma (Algoritamsko razmišljanje) Rad s varijablama i izlazom (Apstrakcija)	Analiza i praćenje algoritma (Algoritamsko razmišljanje)	A2+A8+AL2 (AL3)*

(..)* - oznake uzoraka dizajna sekundarnih koncepata i praksi računalnog razmišljanja

3.3.3. Model dokaza i vrednovanja

Dizajn i primjena visokokvalitetnih vrednovanja vrlo su zahtjevni i dugotrajni. Prema ECD pristupu (Hendrickson i sur., 2013) pretpostavke i hipoteze učitelja o znanju i sposobnosti svojih učenika predstavljat će dokaz o njihovom načinu rješavanja zadataka. Svaki takav dokaz trebao bi pokazati je li učenik usvojio očekivane ishode učenja. U procesu stvaranja modela dokaza, ključno je prepoznati sve moguće dokaze učeničkog znanja koji su povezani s očekivanim ishodima.

U prikazanoj strukturi alata vrednovanja (Tablica 9 Poglavlje 3.3.2.) može se primijetiti da su tri čestice esejski zadaci koji se ne mogu vrednovati automatski (zadatak3, zadatak 7, zadatak 9). U tim se zadacima od učenika očekuje uređivanje ili nadogradnja postojećeg algoritma, odnosno stvaranje novog dijela algoritma čime se gotovo onemogućava automatsko vrednovanje rješenja. Za takve je zadatke kreiran model vrednovanja u kojem se prepoznaju i ističu očekivani dokazi znanja koje bi učenici mogli prikazati pri rješavanju zadatka. U svakom zadatku razmatrani su dokazi znanja koji uvijek idu od najlošijeg mogućeg slučaja (učenik nije uopće pokušao riješiti zadatak) te se postupno razvijaju (djelomično rješavanje zadatka) do najboljeg slučaja (učenik je uspješno riješio zadatak).

U nastojanju da se vrednovanjem prikupe što kvalitetnije informacije o stvarnoj razini usvojenog znanja kod učenika, za svaki dokaz znanja definiran je uočeni problem u razumijevanju koncepata računalnog razmišljanja te mentalni model koji takvo (ne)razumijevanje opisuje. Također, svaki dokaz znanja opisan je brojem bodova koji će se koristiti u svrhu vrednovanja naučenog. Odabranim skupom čestica koje uključuju različite koncepte računalnog razmišljanja te vrstu pitanja kojom je čestica realizirana, prikazan je primjer modela vrednovanja čestica (Tablica 10-15).

Tablica 10: Model vrednovanja za česticu – zadatak 3

Dokazi znanja	Mentalni model	Uočeni problem	Bodovi
Nije uopće napravljena izmjena.	Učenik ne zna što treba napraviti ili uopće se ne uočava pogreška	Nerazumijevanje problema Algoritamsko razmišljanje (nesposobnost analize i praćenja koda)	0
Napravljena je izmjena uvjeta, ali pogrešna.	Učenik je prepoznao pogrešku, ali je ne zna ispraviti.	Algoritamsko razmišljanje (rad s logikom, struktura grananja) Nerazumijevanje problema	1
Napravljena je ispravna izmjena uvjeta, ali je napravljena i još neka nepotrebna izmjena koja se nije tražila i koja kao posljedicu ima neočekivano ponašanje (kretanje lika).	Učenik prepoznaje pogrešku i djelomično je uspješno ispravlja.	Algoritamsko razmišljanje (praćenje i analiza koda) Djelomično nerazumijevanje problema	1.5
Napravljena je izmjena samo u jednoj naredbi grananja koja je usklađena s zadanim smjerom kretanja i postavljenim uvjetom.	Učenik prepoznaje pogrešku i djelomično je uspješno ispravlja.	Djelomično nerazumijevanje problema	1.5
Sve izmjene su ispravno napravljene: • Ispravna izmjena uvjeta.	Učenik uspješno koristi logiku i brzo korigira pogrešku.	-	2
Sve izmjene su ispravno napravljene: • Ispravna izmjena smjera kretanja u obje grane grananja.	Učenik ne mijenja logički uvjet već uspješno ispravlja kretanja u obje grane naredbe if.	Rad s logičkim vrijednostima	

Tablica 11: Model vrednovanja za česticu - zadatak 7

Dokazi znanja	Mentalni model	Uočeni problem	Bodovi
Nije napravljena nikakva izmjena akcije ***hodaj.***	Učenik ne zna ni gdje ni kako napraviti nadogradnju programa.	Nerazumijevanje problema Algoritamsko razmišljanje	0
Napravljena je izmjena u broju koraka (npr. idi 4 koraka, idi 3 koraka…a nije napravljena izmjena koja uključuje povećanje koraka pri kretanju.	Učenik ne prepoznaje varijable kao vrijednosti kojima se može pratiti kretanje Maje kroz labirint, Razumije da treba napraviti neku izmjenu uz samo kretanje.	Rad s varijablama (apstrakcija) Djelomično nerazumijevanje problema	
Napravljena je izmjena o povećavanju brojača za 1, ali na krivom mjestu npr. u akciji Labirint.	Učenik prepoznaje potrebu za uporabom varijable za praćenje kretanja Maje po labirintu no ne prepoznaje odgovarajuće mjesto u programu za takvu promjenu.	Algoritamsko razmišljanje (problem s praćenjem koda) Djelomično nerazumijevanje problema	1

Napravljena je izmjena o povećanju brojača, ali samo na nekim od očekivanih mjesta u uputama kretanja npr. samo pri kretanju desno i gore.	Učenik prepoznaje potrebu za uporabom varijable te je uspješno primjenjuje na nekim od očekivanih mjesta.	Djelomično nerazumijevanje problema Algoritamsko razmišljanje (praćenje koda)	2
Napravljene su sve izmjene o povećanju brojača na očekivanim mjestima.	Učenik uspješno primjenjuje varijable u zadanom problemu.	-	3

Tablica 12: Model dokaza i vrednovanja za česticu - zadatak 9 – uvjet grananja

Dokazi znanja			**Mentalni model**	**Uočeni problem**	**Bodovi**
Uvjet grananja (ako je žaba gore)	**Ispravno napisan uvjet**	**Pogrešno napisan uvjet**			
	x	Nije naveden uvjet grananja if naredbe.	Učenik ne zna napisati uvjet.	Rad s logičkim vrijednostima (apstrakcija) Nerazumijevanje problema	0
		Nije naveden uvjet koji uključuje žabu.	Učenik piše neki uvjet koji ne uključuje žabu.		
	x	Naveden je uvjet koji uključuje žabu, ali je provjerava u krivom smjeru npr. lijevo, ili samo navodi npr. 'ako je žaba onda ….'ili neka druga greška uvjeta.	Učenik spominje žabu u uvjetu, ali provjerava na krivom smjeru ili samo spominje žabu, a ne navodi vrijednost povezanu sa žabom.	Rad s logičkim vrijednostima (apstrakcija) Djelomično nerazumijevanje problema	0.5
	Umjesto naredbe grananja uspješno je primijenjena petlja npr. dok je žaba gore …	x	Učenik primjenjuje uvjetnu petlju umjesto naredbe grananja.	Algoritamska struktura ponavljanja/grananja	1
	DA	x	Učenik se uspješno služi logikom i logičkim vrijednostima u naredbi grananja.	-	1

Tablica 13:Model dokaza i vrednovanja za česticu - zadatak 9 – grana DA

Grana DA (naredbe koje se rade ako je uvjet ispunjen – 'žaba je gore')	Ispravno napisana grana DA	Pogreška u grani DA	Mentalni model	Uočeni problem	Bodovi
	x	Nisu uopće navedene naredbe za kretanje i povećavanje brojača.	Učenik uopće ne zna kako realizirati ni kretanje ni brojanje koraka koji se pri kretanju rade.	Nerazumijevanje problema	0
	x	Napisane su obje naredbe ali se jedna ili obje vrijednosti (korak i brojač) povećavaju za 1.	Učenik je svjestan da mora povezati kretanje i povećavanje vrijednosti brojača, no povremeno koristi krivu vrijednost (1 umjesto 2).	Algoritamsko razmišljanje (vještine praćenja koda) Djelomično nerazumijevanje problema	0.5
	x	Ispravo je napisana samo jedna naredba (za kretanje ili povećavanje brojača) dok druga naredba ili nije napisana ili je pogrešna.	Učenik ne prepoznaje brojač kao vrijednost koja mora pratiti kretanje Maje kroz labirint i koja se mora mijenjati pri svakom njenom kretanju.	Apstrakcija (rad s varijablama)	
	DA	x			1

Tablica 14: Model dokaza i vrednovanja za česticu - zadatak 9 – grana NE

Grana NE (naredbe koje se rade ako uvjet nije ispunjen – 'žaba je gore')	Ispravno napisana grana NE	Pogreška u grani NE	Mentalni model	Uočeni problem	Bodovi
	X	Nisu uopće navedene naredbe za kretanje i povećanje brojača.	Učenik uopće ne zna kako realizirati ni kretanje ni brojanje koraka koji se pri kretanju rade.	Nerazumijevanje problema	0
	X	Napisane su obje naredbe, ali se jedna ili obje vrijednosti povećavaju za 2.	Učenik je svjestan da mora povezati kretanje i povećavanje vrijednosti brojača, no povremeno koristi krivu vrijednost (1 umjesto 2).	Algoritamsko razmišljanje (vještine praćenja koda) Djelomično nerazumijevanje problema	0.5
	X	Ispravo je napisana samo jedna naredba (za kretanje ili povećavanje brojača) dok druga naredba ili nije napisana ili je pogrešna.	Učenik ne prepoznaje brojač kao vrijednost koja mora pratiti kretanje Maje kroz labirint i koja se mora mijenjati pri svakom njenom kretanju.		
	DA	X		-	1

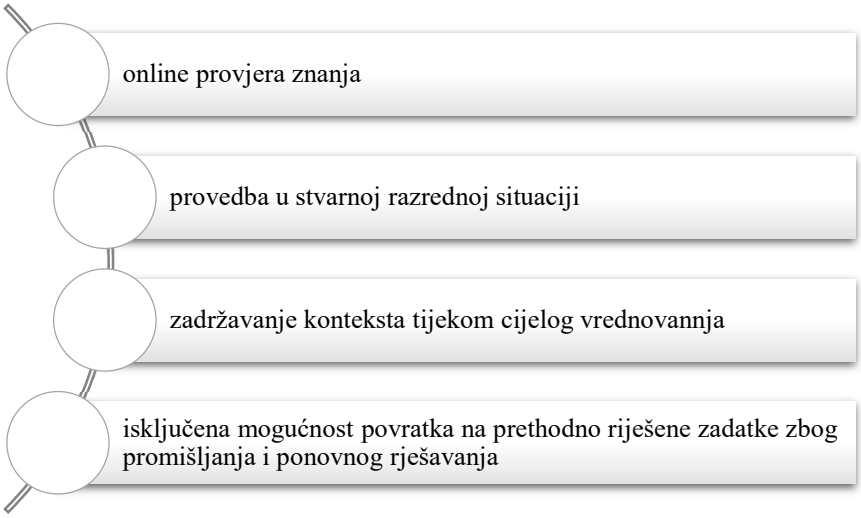
Tablica 15: Model dokaza i vrednovanja za česticu - zadatak 9 – ostali slučajevi

	Dokazi znanja	Mentalni model	Uočeni problem	Bodovi
Ostali slučajevi	Naveden je neki niz naredbi koje su nepovezane i uglavnom predstavljaju jednostavan slijed kretanja (gore, dolje, lijevo ili desno) za neki broj koraka.	Učenik piše samo nekoliko naredbi kretanja ili povećanja brojača koje djeluju jednokratno.	Nerazumijevanje problema Algoritamsko razmišljanje (struktura grananja)	0
	Nije napisana naredba grananja već niz naredbi kretanja i povećanja brojača za 1 (odnosno za 2) kao da se jednom izvršavaju naredbe iz grana DA i NE.	Učenik stvara jednu ili niz od nekoliko naredbu kretanja ili izmjena brojača – ne poznaje strukturu grananja.	Logika (apstrakcija) Algoritamsko razmišljanje (struktura grananja)	0,5
	Učenik je napisao dvije if naredbe umjesto jedne if..else…naredbe : promatrati svaki dio naredbe (uvjet, naredbe) te bodovati prema gore navedenoj uputi (primjer točnog rješenja): *ako je žaba gore* *idi dva koraka gore* *povećaj broj koraka za 2* *ako žaba nije gore* *idi jedan korak gore* *povećaj broj koraka za 1*	Učenik koristi dvije if.. umjesto jednu if..else naredbu.	Još uvijek nije dobro usvojio rad s logičkim vrijednostima (apstrakcija)	3

3.4. Primjena vrednovanja

Na temelju predstavljenog modela kreiran je alat vrednovanja u obliku online provjere koja se sastoji od deset zadataka. Iako je poznato da će se pouzdanost testa povećati s većim brojem zadataka uključenih u vrednovanje (Cohen i sur., 2000), pri kreiranju vrednovanja mora se voditi računa da će se vrednovanje provoditi u stvarnoj nastavnoj situaciji. Iz tog razloga, broj zadataka alata mora biti usklađen s duljinom nastavnog sata u školi te istovremeno zadatci moraju udovoljiti svim ishodima koji se žele ispuniti vrednovanjem.

Prema definiranom modelu kontekst zadataka vrednovanja zadržan je kroz cijelu strukturu alata kako bi se učenicima olakšalo razumijevanje zadataka. Nadalje, zadaci su organizirani na način da lakši zadaci dolaze na početku alata. Upravo takva struktura testa, koja zadržava kontekst zadataka kroz cijelo vrednovanje te postupno razvija zadatke (većina zadataka predstavlja određenu nadogradnju prethodnog zadatka) zahtijeva kreiranje online provjere koja će isključiti mogućnosti vraćanja na prethodno riješene zadatke zbog ponovnog razmatranja ili rješavanja (Slika 21).



Slika 21: Primjena vrednovanja

Smještanjem alata vrednovanja na sustav za udaljeno učenje omogućio se jednostavan pristup te sudjelovanje učenicima iz različitih dijelova RH, što je od iznimne važnosti za istraživanje. U sklopu sustava za udaljeno učenje Loomen (Loomen, 2018) otvoren je novi kolegij s naslovom Računalno razmišljanje (Bubica, 2018). Za svaku školu koja je sudjelovala u istraživanju kreirana je posebna grupa zaštićena pristupnom lozinkom. Učiteljima koji su bili uključeni u istraživanje, dodijeljene su odgovarajuće dozvole te pristupne lozinke kojima su mogli upravljati pridruživanjem svojih učenika kolegiju.

U sklopu kolegija kreirana je uvodna anketa i alat vrednovanja za učenike dok je za učitelje kreirana izlazna anketa. Svaki je učitelj pregledao alat vrednovanja prije njegove provedbe u svom razredu kako bi provjerio odgovaraju li sadržaji koji se provjeravaju, sadržajima koji su učenici usvajali u redovnom nastavnom procesu.

Učitelji koji su sudjelovali u istraživanju pozvani su da se uključe u vrednovanje putem e-mail pozivnice od strane istraživača. Također, uz pozivnicu, učitelji su dobili osnovne upute o načinu provedbe istraživanja, te sve potrebne dokumente (Listić za praćenje aktivnosti učenika tijekom rješavanja testa, PRILOG 4). Budući da su svi nastavni sadržaji uključeni u istraživanje bili u skladu s trenutnim Nastavnim planom i programom, što je u skladu sa zahtjevima 'in situ' istraživanja, posebna suglasnost za primjenu istraživanja nije bila potrebna.

3.5. Objava rezultata vrednovanja

Završna faza vrednovanja računalnog razmišljanja temeljnog na dokazu predstavlja objavu rezultata vrednovanja. Za tu svrhu, svim sudionicima istraživanja, učenicima i učiteljima, i nakon provedenog vrednovanja omogućen je pristup e-kolegiju.

U cilju očuvanja rezultata istraživanja nije omogućen postupak ponovnog rješavanja testa, već su ovlasti ograničene samo na pregled odgovora i zadataka. Nakon analize svih rezultata svim učiteljima bit će poslan izvještaj u kojem se prezentiraju rezultati njihovih učenika. Cjeloviti rezultati istraživanja bit će naknadno predstavljeni i objavljeni na različitim javnim mjestima.

4. METODOLOGIJA ISTRAŽIVANJA

4.1. Predmet i cilj istraživanja

Istraživanje je provedeno u prirodnom razrednom okružju u periodu od ožujka do svibnja 2018. godine. U tom dijelu nastavne godine vrijedila je pretpostavka da će učenici uglavnom završiti ili biti u završnoj fazi obrade nastavne cjeline programiranja prema još uvijek važećem Nastavnom planu i programu za izborni predmet Informatika za šk. god. 2017./2018. (Ministarstvo znanosti i obrazovanja, 2006). Istraživanje se realiziralo putem sustava za udaljeno učenje Loomen jer svi učenici osnovnih i srednjih škola u Republici Hrvatskoj imaju jedinstven pristup sustavu Loomen putem svojih AAI@EduHr identiteta (Srce, 2008). Takvim pristupom osigurala se autentičnost te jednoznačnost sudionika istraživanja.

U vrijeme provedbe istraživanja predmet Informatika bio je izborni predmet za učenike od 5. do 8. razreda osnovne škole. Broj učenika koji su uključeni u nastavu izbornog predmeta Informatika najveći je upravo u petom razreda i opada u višim razredima. Nadalje, svaki učitelj samostalno odabire vrijeme podučavanja sadržaja programiranja tijekom nastavne godine pa neki učitelji to rade na početku nastavne godine, neki sredinom, a neki na samom kraju. Kako bi se u uzorak istraživanja uključio što je moguće veći broj učenika za koje vrijedi da su tijekom svog nastavnog rada upoznali barem osnovne programerske koncepte, odabrani su učenici šestog razreda, a istraživanje je provedeno u drugoj polovici drugog obrazovnog razdoblja.

Vrednovanje računalnog razmišljanja bilo je ograničeno na koncepte algoritamskog razmišljanja, dekompozicije te apstrakcije i to u razinama usvojenosti koje su u skladu s ishodima učenja kurikula predmeta Informatika za šesti razred osnovne škole (Ministarstvo znanosti i obrazovanja, 2018). Obrazovni ishodi koji su predviđeni u domeni Računalno razmišljanje i programiranje za učenike šestih razreda osnovne škole, odnose se na stvaranje algoritamskih rješenja koja uključuju algoritamske strukture slijeda, grananja i ponavljanja što je u skladu s tada još uvijek važećim Nastavnim planom i programom predmeta Informatika. Prema predmetnom kurikulu ostali koncepti računalnog razmišljanja predviđeni su za vrednovanje u višim razredima osnovne škole ili u srednjoj školi.

Uz vrednovanje računalnog razmišljanja prikupljani su i osobni podatci ispitanika putem online upitnika u sklopu posebnog kolegija. Alat vrednovanja sadržavao je deset pitanja (pitanja višestrukih odgovora, pitanja kratkih odgovora i esejska pitanja za odgovore kojima se uređuju algoritamska rješenja) s predviđenim trajanjem od jednog nastavnog sata. S obzirom da je vrednovanje pratilo redovni nastavni sadržaj, njegovom primjenom nije se poremetila provedba

redovnog obrazovnog procesa. Može se smatrati da je istraživanje ispunilo zahtjev „in situ“ istraživanja. Ovim modelom vrednovanja ponudio se online alat vrednovanja koji je javno dostupan učenicima i učiteljima za vrednovanje koncepata računalnog razmišljanja (algoritamskog razmišljanja, apstrakcija, dekompozicija) u stvarnom razrednom okružju/situaciji.

Cilj istraživanja bio je vrednovati usvojenost ishoda računalnog razmišljanja i programiranja u razrednom okružju šestih razreda učenika osnovne škole. Postavljene su sljedeće hipoteze:

H1: Postoji korelacija između uspjeha kojeg postižu učenici šestih razreda osnovnih škola pri vrednovanju koncepata računalnog razmišljanja i programiranja.

H2: Postoji korelacija između uspjeha kojeg postižu učenici šestih razreda osnovnih škola pri vrednovanju koncepata računalnog razmišljanja i uspjeha iz matematike.

H3: Ovaj alat vrednovanja računalnog razmišljanja bolji je prediktor uspjeha u programiranju od akademskog uspjeha, ocjene iz matematike te prvog programskog jezika.

4.2. Istraživačka pitanja

- Može li se uspjeh pri usvajanju koncepata računalnog razmišljanja, kao što su algoritamsko razmišljanje, apstrakcija te dekompozicija, smatrati prediktorom uspjeha programera početnika?
- Je li predloženi alat vrednovanja prikladan alat za vrednovanje koncepata računalnog razmišljanja (dekompozicije, apstrakcije, algoritamskog razmišljanja) namijenjen učenicima šestih razreda osnovne škole?

4.3. Paradigma i metodološki pristup

Planirano istraživanje ubraja se u naturalistička istraživanja kojima je osnovno obilježje odbacivanje gledišta udaljenog tzv. objektivnog opažača te da razumijevanje pojedinca mora doći iznutra. Istraživane su pojave koje nisu jedinstvene. Naturalističkim pristupom podatci se često prikupljaju *in situ*, što u našem slučaju znači prikupljanje podataka u realnom razrednom okružju učenika.

Provedeno istraživanje moralo se uklopiti u redovnu realizaciju nastavnih sadržaja. U istraživanjima u obrazovanju često se koriste testovi kao jedna od moćnijih strategija prikupljanja podataka i istraživanja. U ovom istraživanju koristio se test, napravljen od strane istraživača, čija se primjena provela na odabranoj populaciji učenika šestih razreda.

4.4. Nacrt istraživanja

Osnovna ideja predloženog modela vrednovanja prvi je put testirana u uvodnom istraživanju provedenom tijekom školske godine 2016./2017. Tada je provedena online provjera znanja posebno prilagođena za programski jezik Python. Kao alat vrednovanja odabran je sustav za udaljeno učenje Loomen (Learning Managment System Loomen) jer svi učenici RH posjeduju jedinstvene pristupne podatke za taj sustav. Nadalje, provedeno je i pilot istraživanje tijekom školske godine 2017./2018. također uz pomoć sustava za udaljeno učenje Loomen.

Pozitivna iskustva i povratne informacije vezane uz primjenu sustava za udaljeno učenje Loomen, utjecala su na odluku da se za provedbu glavnog istraživanja također koristi taj sustav. Korištenjem jedinstvenih pristupnih podataka za sustav Loomen kojeg posjeduje svaki učenik RH, sačuvana je vjerodostojnost podataka sudionika istraživanja.

U prvoj fazi istraživanja provedeno je pilot istraživanje u trajanju od dva tjedna (ožujak 2018). U prvom tjednu pilot istraživanja provedena je ulazna anketa za učenike. U drugom tjednu primijenjen je alat vrednovanja te prikupljeni kvalitetni podatci u obliku bilješki učitelja vezano uz ponašanje i aktivnosti učenika tijekom provedbe alata vrednovanja.

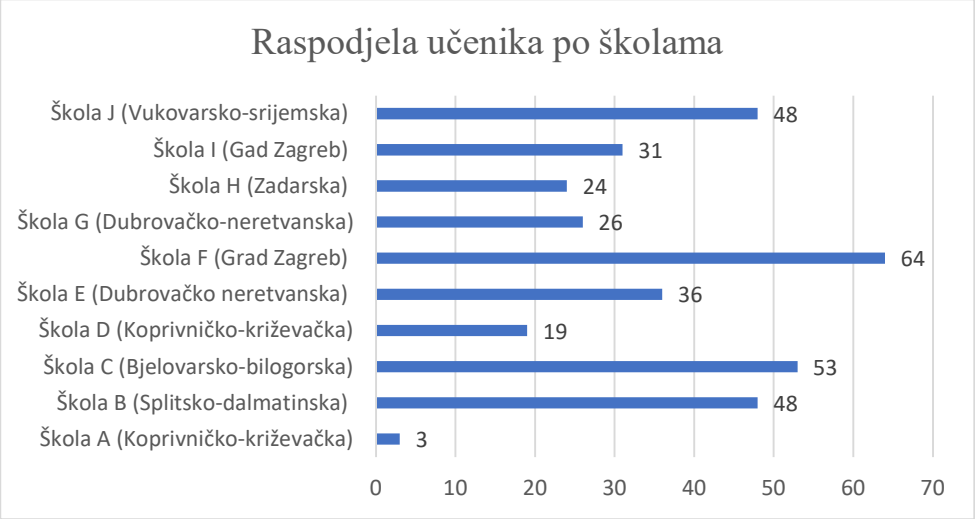
U drugoj fazi istraživanja pristupilo se analizi prikupljenih podataka iz pilot istraživanja te bilješki učitelja. Napravljene su dodatne izmjene alata vrednovanja.

Tijekom sljedeća dva mjeseca (travanj/svibanj 2018.) provedeno je glavno istraživanje na odabranom (prigodnom) uzorku učenika u trajanju od dva tjedna: u prvom tjednu provedena je prijava učenika u kolegij Računalno razmišljanje, posebno kreiranim za potrebe ovog istraživanja. Također, u prvom tjednu provedena je i ulazna anketa za učenike. Tijekom prvog

tjedan učitelji su imali pristup kolegiju i alatu vrednovanja. U tom periodu učitelji su proučili alat i provjerili primjerenost alata za njihove učenike kako bi se u potpunosti ispitala valjanost zahtjeva „in situ“ istraživanja. U drugom tjednu istraživanja primijenjen je alat vrednovanja te izlazna anketa za učitelje.

4.5. Uzorak

Istraživanje je provedeno među učenicima šestih razreda osnovne škole. U istraživanju je sudjelovalo 407 učenika osnovnih škola iz osam županija Republike Hrvatske (deset osnovnih škola). Nakon pregleda učeničkih odgovora isključeni su odgovori onih učenika koji su samo pokrenuli vrednovanje i odmah ga završili ili su proveli manje od deset minuta za njegovo rješavanje. Svi navedeni rezultati u ovom istraživanju odnose se na uzorak od 352 učenika (Slika 22). U istraživanje su uključeni učenici onih učitelja koji su prethodno pristali sudjelovati u samom istraživanju. Učenici su pristupali vrednovanju prijavom u sustav Loomen te uključivanjem u posebni kolegij e-učenja o računalnom razmišljanju. Ovakvim odabirom ispitanika stvorio se prigodni uzorak.

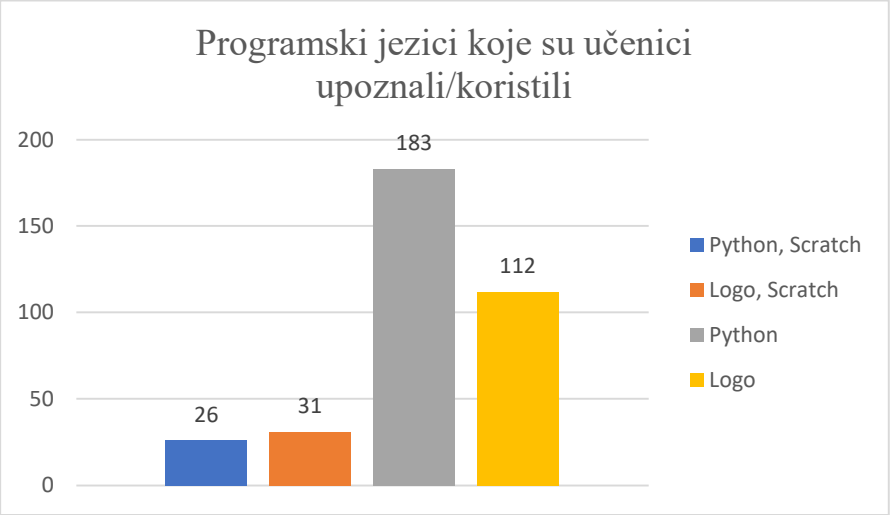


Slika 22: Raspodjela učenika po školama i županijama

U istraživanju je sudjelovalo 159 djevojčica (45%) i 193 dječaka (55%) (Slika 23).

Slika 23: Raspodjela učenika po spolu

Svi učenici koji su sudjelovali u istraživanju imali su neko ranije iskustvo programiranja. Svoja dotadašnja iskustva programiranja ostvarili su putem programskih jezika/okružja Python, Logo i Scratch ili u nekoj njihovoj kombinaciji. Sljedeća slika prikazuje raspodjelu učenika prema programskim jezicima ili okruženjima koje su dotad upoznali (Slika 24).



Slika 24: Raspodjela učenika po programskim jezicima koje su upoznali

4.6. Mjerni instrumenti i analiza podataka

U ovom istraživanju korišteno je više mjernih instrumenata:

- ulazna anketa za učenike
- izlazna anketa učitelja praktičara
- promatranja učitelja pri rješavanju testa (listić za bilješke učitelja)
- alat vrednovanja koncepata računalnog razmišljanja (test)
- dodatni test za vrednovanje usvojenosti računalnog razmišljanja (test natjecanja Dabar 2017).

Prijavom u sustav putem AAI@EduHr identiteta te sudjelovanjem učenika iz različitih dijelova RH osigurala se vanjska valjanost istraživanja. Ulaznom anketom prikupili su se neki osnovni podatci o učeniku kao što su spol, akademski uspjeh, sudjelovanje na natjecanju Dabar, ocjena iz Matematike (srednja ocjena predmeta), programski jezik kojeg je učenik upoznao tijekom nastavnog procesa (ili na neki drugi način). Svi navedeni podatci poslužili su za provedbu dodatnih analiza s naglaskom na moguće prediktore uspješnih programera početnika. Ulaznu anketu učenici su ispunili nakon prijave u sustav za udaljeno učenje Loomen u sklopu posebnog kolegija Računalno razmišljanje, a prije rješavanja samog testa računalnog razmišljanja.

Izlaznom anketom učitelja praktičara pokušao se pronaći odgovor na pitanje odgovaraju li čestice testa nepristrano i sveobuhvatno sadržaju koji se vrednuje, ali i prikupiti stručne osvrte na usklađenost te povezanost pojedinih pitanja te koncepata koji se vrednuju. Time se nastojala provjeriti sadržajna valjanost samog alata vrednovanja.

Pilot istraživanje imalo je za cilj istražiti obilježja čestica alata vrednovanja s obzirom na njihovu jasnoću, nedvosmislenost, preciznost izražavanja te prilagođenost uzrastu kojem je namijenjen. U pilot istraživanju sudjelovala su 43 učenika iz tri različite škole (Dubrovačko-neretvanska županija (1), Karlovačka županija (2)).

Pouzdanost alata vrednovanja bila je 0.63 (Cronbach alpha). Svi su zadaci bili odgovarajuće težine (0.19-0.73). Nije bilo pretjerano teških ili pretjerano laganih zadataka. Diskriminatorni indeks je za većinu zadataka bio izvrstan (0,27-0.63) dok je za dva zadataka rezultat bio iznimno loš (0,06; 0,08), odnosno neprihvatljiv. U skladu s dobivenim rezultatima korigirala su se dva zadataka (Zadatak 6, zadatak 10). Na osnovu povratnih informacija učitelja, putem usmenog razgovora ili pisanog komentara putem e-pošte, pojedini grafički prikazi su unaprijeđeni (kontrast boja, jasnoća slike), nacrtani su dozvoljeni smjerovi kretanja lika u svakom zadatku, dodatno su objašnjeni pojmovi 'idi gore', 'idi desno' te značenje prepreke i ruba labirinta za

razumijevanje kretanja glavnog lika. Učenici nisu tražili dodatno vrijeme za rješavanje zadataka pa se početno postavljena duljina rješavanja zadataka od jednog školskog sata (45 minuta) nije mijenjala.

Kvalitativni podatci prikupljeni su promatranjem i intervjuiranjem učitelja od strane istraživača, a odnosili su se na ponašanje i aktivnosti učenika tijekom rješavanja testa (jesu li tražili pomoć pri razumijevanju pitanja, jesu li tražili pomoć prilikom obilježavanja ili unošenja svojih odgovora, je li bilo pitanja vezano uz jasnoću grafičkih prikaza i sl.). Svoja zapažanja učitelji su bilježili u posebno kreiranom listiću.

Za analizu kvantitativnih podataka koristio se programski alat SPSS 24.0 za statističku obradu podataka s primjenom odgovarajućih metoda za određivanje postojanja povezanosti među podatcima. Provjeravala se pouzdanost ulaznih podataka koeficijentom pouzdanosti (npr. Cronbach alpha), provedena je faktorska analiza za provjeru odnosa među konstruktima te nastojala ojačati pouzdanost provedbom alata vrednovanja u istom periodu školske godine, s naglaskom da vrednovanje neće rezultirati ocjenom za učenika. Pouzdanost alata vrednovanja osnažila se uvođenjem dodatnog ocjenjivača koji je vrednovao rješenja učenika prema zadanom okviru.

5. REZULTATI I RASPRAVA

5.1. Osnovna statistička obilježja alata vrednovanja

Osnovna obilježja opisanog modela vrednovanja računalnog razmišljanja početno su testirana provedbom preliminarnog istraživanja s mjernim instrumentom prilagođenim za programski jezik Python, koje je provedeno tijekom školske godine 2016./2017. Osnovni cilj preliminarnog istraživanja bio je vrednovati razinu usvojenosti ishoda učenja povezanih s vještinom programiranja u programskom jeziku Python. Alat vrednovanja proveden je nakon dvanaest tjedana nastave programiranja s učenicima šestih razreda, te nakon četrnaest tjedana nastave programiranja s učenicima sedmih razreda. Preliminarno istraživanje provedeno je na uzorku od petnaest učenika šestih razreda (osam djevojčica, sedam dječaka) te deset učenika sedmih razreda (tri djevojčice, sedam dječaka). Vrednovanje je organizirano u obliku online provjere znanja putem sustava za udaljeno učenje Loomen. Online provjera sastojala se od osam zadataka (jedan zadatak uparivanja, četiri zadatka višestrukog odgovora, tri esejska zadatka otvorenog tipa) s ograničenom duljinom trajanja od četrdeset pet minuta (trajanje jednog školskog sata).

U povratnim informacijama nakon provedbe vrednovanja učenici su iskazali veliko zadovoljstvo primjenom online provjere znanja umjesto standardne provjere papir-olovka iako im je to bio prvi susret s online oblikom vrednovanja. Statistička obrada rezultata provedena je programom SPSS Statistic verzija 24.0. Pozitivni i obećavajući rezultati preliminarnog istraživanja potaknuli su na stvaranje novog modela vrednovanja, neovisnog o programskom jeziku ili okružju. Također, ponudili su održiv model vrednovanja računalnog razmišljanja temeljen na dokazu (Bubica i Boljat, 2018).

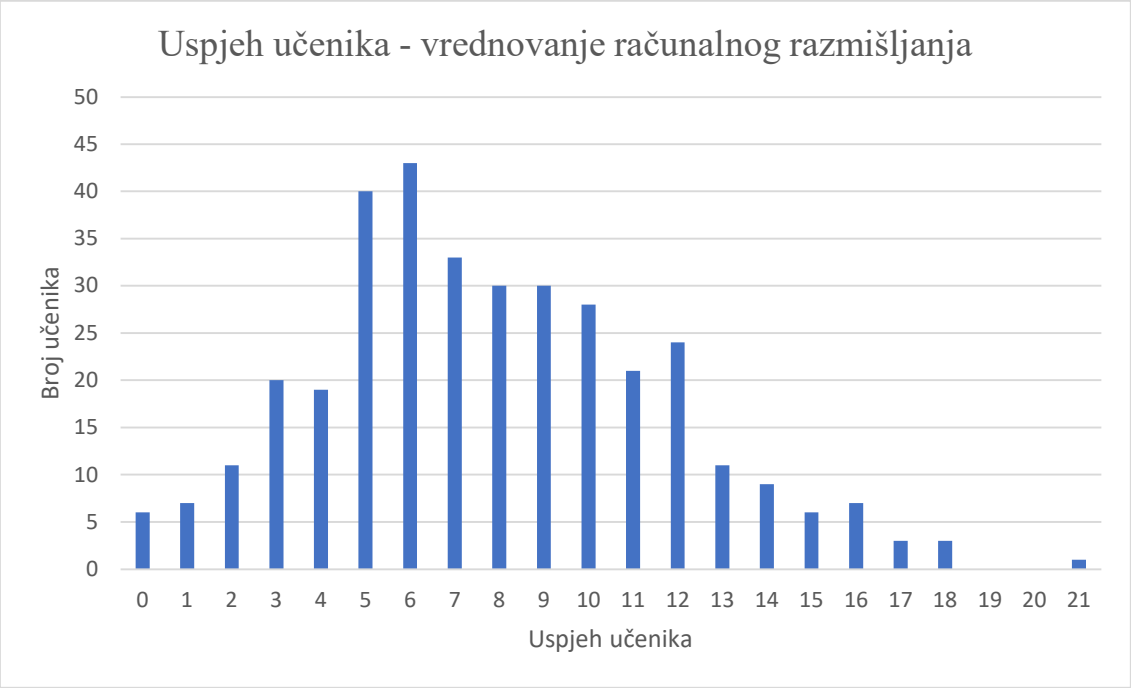
Pilot istraživanje provedeno je tijekom ožujka 2018. godine na uzorku od 42 učenika šestih i sedmih razreda triju osnovnih škola (Dubrovnik, Karlovac; 26 dječaka, 16 djevojčica). Vrednovanje je provedeno kao online provjera znanja sačinjana od deset pitanja (pitanja kratkog odgovora (2), pitanja višestrukog odgovora (5), esejska pitanja (3)). Provjera je realizirana putem sustava za udaljeno učenja Loomen s obzirom da svaki učenik osnovnih i srednjih škola RH posjeduje jedinstvene pristupne podatke za taj sustav.

Također, tijekom pilot istraživanja prikupljeni su i neki kvantitativni te kvalitativni podatci kroz povratne informacije učitelja o problemima koji su se pojavili tijekom provedbe samog vrednovanja. S obzirom na statističke parametre provedenog vrednovanja (Cronbach Aplha koeficijent unutarnje pouzdanosti 0.6743; indeks diskriminativnosti 8/10 zadataka >0.35; 2/10 zadataka ~ 0.1) napravljene su neke izmjene u alatu vrednovanja u cilju korigiranja

nezadovoljavajućih rezultata vezano uz pojedine zadatke te kreirana završna verzija alata vrednovanja.

Osnovno istraživanje provedeno je tijekom mjeseca ožujak/travanj 2018. godine na uzorku od 407 učenika iz deset različitih škola smještenih u osam gradova i sedam županija. Kao i u pilot istraživanju, za provedbu vrednovanja odabran je sustav za udaljeno učenje Loomen jer omogućava jedinstveni pristup svakom učeniku u sustav. Prvom analizom prikupljenih podataka, 27 učenika je isključeno iz daljnjih razmatranja jer nisu predali svoja rješenja. Također, daljnjih 28 učenika je isključeno iz razmatranja jer su svoje vrednovanje završili za manje od deset minuta što nije bilo dovoljno niti za prvo čitanje većine zadataka. Ti učenici uglavnom nisu predali rješenja pojedinih zadataka. Sve daljnje rasprave o uzorku učenika koji su sudjelovali u istraživanju odnose se na preostala 352 učenika (193 dječaka, 159 djevojčica).

Sljedeća slika (Slika 25) i tablica (Tablica 16) prikazuju osnovna deskriptivna obilježja alata vrednovanja. Rezultati nisu zadovoljili kriterij normalnosti.



Slika 25: Uspjeh učenika - vrednovanje računalnog razmišljanja

Srednja vrijednost na alatu vrednovanja (Mean) bila je 7.6857 što je značajno manje od polovice mogućeg rezultata (10.5) pa se može reći da je vrednovanje bilo teško za odabranu skupinu učenika. Centralni rezultat (Median) bio je 7.17 što znači da je čak 50% učenika postiglo najviše 7.17 bodova, odnosno da je čak 50% učenika ispravno riješilo samo 34.17% zadataka iz alata vrednovanja. To nije zadovoljavajući rezultat jer bi u standardnoj nastavnoj praksi vjerojatno takav rezultat značio da je barem 50% učenika postiglo nezadovoljavajući rezultat na provjeri znanja (uz pretpostavku da se barem 40% ispravno riješenih zadataka smatra prolaznim rezultatom na vrednovanju znanja). Pozitivno je što se na testu postigao najveći (1 učenik) i najmanji (6 učenika) mogući broj bodova, čime je zadovoljen zahtjev za maksimalnim rasponom rezultata.

Tablica 16: Deskriptivna obilježja alata vrednovanja računalnog razmišljanja

	N	Min	Max	Mean	Median	Std. Deviation	Variance
Uspjeh na vrednovanju računalnog razmišljanja	352	0,00	21,00	7,6857	7,17	3,82726	14,648
Valid N (listwise)	352						
Skewness	0.388						
Skewness (Std. Error)	0,130						
Kurtosis	-0.014						
Kurtosis (Std. Error)	0,256						

Analiza podataka vezano uz indeks težine te diskriminativnosti čestica alata vrednovanja prikazana je sljedećom tablicom (Tablica 17). Prema Cohen, Manion and Morison (Cohen i sur., 2000) sedam zadataka pokazalo je prihvatljivu indeks težine (33-67), jedan zadatak je bio vjerojatno prelagan (28.12) i tri zadatka su vjerojatno bila jako teška (85.72 – 93.90).

Tablica 17: Statistički podatci provjere - indeks težine i diskriminativnosti zadataka

Čestica alata	Indeks težine	Indeks diskriminativnosti	Čestica alata	Indeks težine	Indeks diskriminativnosti
1	59.42	0.32	6	57.10	0.29
2	28.12*	0.29	7	89.74**	0.41
3	44.03	0.38	8	51.19	0.36
4	59.42	0.11	9	85.72**	0.45
5	46.68	0.27	10	93.90**	0.16

* - vjerojatno vrlo lagan zadatak ** vjerojatno vrlo težak zadatak

Promatranjem prikazanih rezultata sa stanovišta strukture alata vrednovanja te praksi i vještina koncepata računalnog razmišljanja uključenih u vrednovanje, može se zaključiti da su upravo

zadaci analize te razumijevanja postojećeg algoritma, nadalje zadaci predviđanja ponašanja zadanog algoritma, zadaci nadogradnje postojećeg te stvaranja potpuno novog modula algoritamskog rješenja predstavljali učenicima najveći izazov (zadatak 7, 9, 10).

Vezano uz indeks diskriminativnosti, četiri su se zadatka pokazala izvrsna (>0.35), četiri zadatka dobra (0.25-0.34) dok je jedan zadatak bio na prihvatljivoj razini (0.16 – 0.24). S obzirom da je jedan zadatak pokazao loš diskriminatorni indeks (<0.15), u nekom budućem uređivanju alata vrednovanja trebalo bi ga razmatrati kao zadatak koji bi trebalo isključiti iz vrednovanja ili ga značajno unaprijediti. Očekivano se pokazalo da su upravo zadaci koji su bili najzahtjevniji za učenike (stvaranje vlastitih algoritamskih rješenja, nadogradnja ili izmjena postojećih rješenja), pokazali veliku snagu diskriminativnosti, odnosno veliku mogućnost razlučivanja boljih od lošijih učenika.

Razmatranje indeksa težine čestica alata vrednovanja važno je razmatrati i s obzirom na redoslijed čestica. Uobičajeno je slagati čestice od najlakših prema najtežim. Iz rezultata je vidljivo da se za promatrani alat vrednovanja težina čestica povećava prema kraju alata vrednovanja. Ipak, očigledno je težina prve čestice alata pretjerana (59.42) dok je sljedeća čestica prilično lagana (28.12). S obzirom da su čestice alata vrednovanja sadržajno povezane te rješenjima slijede jedna drugu, nije moguće izvršiti zamjenu navedenih čestica kako bi se psihološki olakšao početak rješavanja. U tom slučaju moguće rješenje moglo bi biti dodavanje lakših čestica na početak alata ili izmjena postojeće prve čestice vrednovanja na način da ima niži indeks težine od trenutne.

Jedan od najvažnijih statističkih parametara pri analizi alata vrednovanja svakako je pouzdanost. Pouzdanost je statistička i psihometrijska mjera kojom se izražava ukupna dosljednost alata vrednovanja. Tako se za alat koji ima visoku razinu pouzdanosti, može reći da će pri ponovnoj primjeni alata vrednovanja u istim uvjetima proizvesti slične rezultate (Cohen i sur., 2000). Alat vrednovanja predstavljen u ovom istraživanju pokazao je zadovoljavajući stupanj unutarnje pouzdanosti (Cronbach alpha $\alpha = 0.630$). Općenito je prihvaćeno da se vrijednosti koeficijenta pouzdanosti Cronbach aplha iznad 0.7 smatraju prihvatljivim (Cohen i sur., 2007; Nunnally, 1978; Pallant, 2005; Pallant, 2011).

Također, vrijednost koeficijenta pouzdanosti (Cronbach alpha) ovisi i o broju čestica uključenih u promatrani alat vrednovanja: niska vrijednost koeficijenta, na primjer 0.5, nije neuobičajena za alate vrednovanja koji imaju do deset čestica kao što je alat iz ovog istraživanja. U takvim je situacijama uobičajeno analizirati ispravljene korelacije čestica alata vrednovanja (engl.

corrected correlations among test items). Analiza rezultata pokazala je da se za dvije čestice alata (zadatak 4, r = 0,104; zadatak 10, r = 0,150) može zaključiti da one ne mjere isto što i ostatak instrumenta jer njihove vrijednosti korelacija odstupaju od optimalnih vrijednosti među (ostalim) česticama (Briggs i Cheek, 1986).

Nadalje, ako bi se te stavke uklonile iz alata vrednovanja, vrijednost koeficijenta pouzdanosti (Cronbach alpha) ne bi se uopće promijenila (zadatak 10) ili bi se malo povećala (zadatak 4). Budući da je jedna od ovih čestica (zadatak 4) pokazala izuzetno lošu vrijednost indeksa diskriminativnosti (0,1052), njezino uklanjanje ili značajno poboljšanje trebalo bi ozbiljno razmotriti. Također, trebalo bi razmotriti blago povećanje broja čestica u alatu vrednovanja, kao na primjer s česticama koje se bave konceptom dekompozicije, jer će veći broj čestica osnažiti pouzdanost ispitivanja.

Jedan od načina povećanja pouzdanosti vrednovanja svakako je i triangulacija ocjenjivača. Triangulacijom se nastoji pridonijeti objektivnosti i interpretativnoj valjanosti istraživačkih objašnjenja. Kako bi se povećala pouzdanost alata vrednovanja, uveden je dodatni ocjenjivač za esejske zadatke koji se nisu ocjenjivali automatski. Dodatni ocjenjivač, iskusni učitelj s velikim iskustvom u području podučavanja i vrednovanja, samostalno je ocjenjivao esejske zadatke (zadatak 3, zadatak 7, zadatak 9) prema predstavljenom modelu dokaza i mjerenja. Dodatni ocjenjivač pri tom procesu nije imao pristup ocjenama prvog ocjenjivača. Usporedba ocjena prvog i dodatnog ocjenjivača pokazala je vrlo visok stupanj podudaranja u sva tri zadatka (zadatak 3: 96%; zadatak 7: 97%; zadatak 9: 97%).

5.2. Sposobnost alata vrednovanja za vrednovanje koncepata računalnog razmišljanja apstrakcije, algoritamskog razmišljanja i dekompozicije

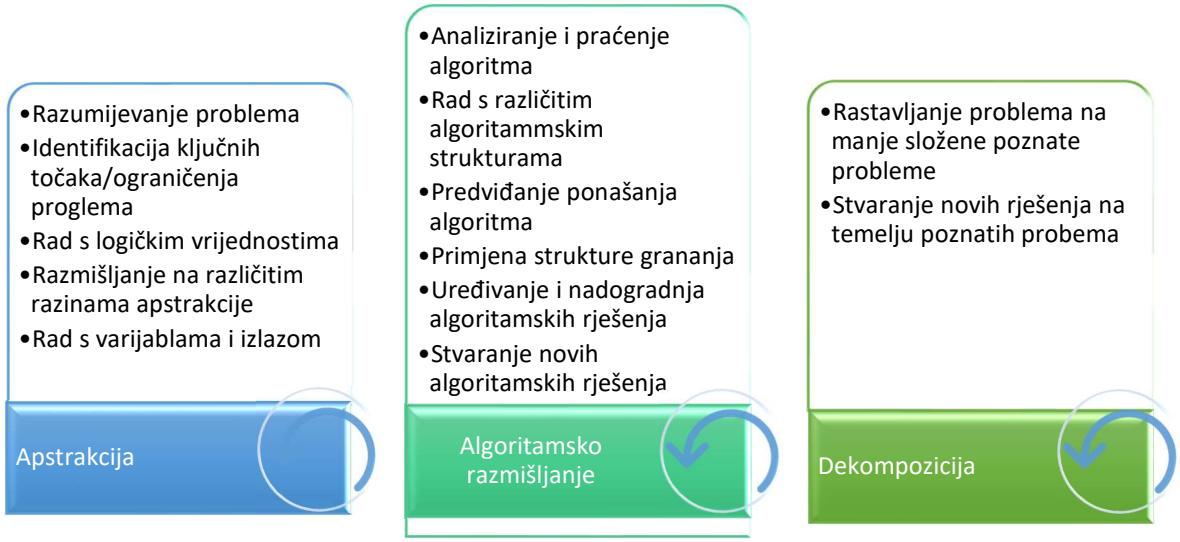
Pri istraživanju kvalitete i snage odabranog alata vrednovanja jedno od ključnih pitanja je pitanje valjanosti. Alat vrednovanja (instrument) toliko je valjan koliko on mjeri ono za što je namijenjen. Cilj ovog dijela istraživanja je provjeriti koliko čestice alata vrednovanja zaista mjere koncepte računalnog razmišljanja apstrakciju, algoritamsko razmišljanje te dekompoziciju.

U tu svrhu izrađen je model vrednovanja računalnog razmišljanja temeljen na dokazu usklađen s hrvatskim kurikulom predmeta Informatika. Stvoreni alat vrednovanja usklađen je s obrazovnim ishodima novog predmetnog kurikula za učenike šestog razreda osnovne škole. Jedno od ključnih pitanja istraživanja bilo je odrediti koliko je predloženi okvir vrednovanja primjeren za vrednovanje odabranih koncepata računalnog razmišljanja.

5.2.1. Opis istraživanja

S ciljem da se utvrdi jesu li čestice alata vrednovanja primjereno odabrane, odnosno odgovaraju li čestice vrednovanja željenim sadržajima: konceptima apstrakcije, algoritamskog razmišljanja i dekompozicije, čestice alata vrednovanja analizirane su s obzirom na primijenjene koncepte računalnog razmišljanja. Uspostavljena je poveznica između kreiranih čestica alata vrednovanja te računalnih praksi i vještina računalnog razmišljanja koje se promatraju u ovom istraživanju.

Sljedeća slika prikazuje popis praksi i vještina računalnog razmišljanja za koncepte apstrakcija, algoritamskog razmišljanja idekompozicije, koje će se vrednovati predloženim modelom, a koje su u skladu s ishodima predmetnog kurikula (Slika 26).



Slika 26: Opis promatranih koncepata računalnog razmišljanja

U opisu strukture alata vrednovanja, svaka je čestica alata opisana praksom i vještinom pojedinog koncepta računalnog razmišljanja na koji se odnosi alat vrednovanja (Tablica 9 Poglavlje 4.3.2.). Takvim prikazom istaknuta je veza između promatranih koncepata računalnog razmišljanja te odabranih čestica alata vrednovanja.

U nastojanju da se otkrije postoji li odgovarajuća povezanost pojedinih čestica alata vrednovanja, istražen je stupanj njihove međusobne unutarnje povezanosti. Cilj je bio utvrditi mogu li se čestice alata grupirati na neki način s obzirom na njihov međusobni odnos, odnosno na temelju nekog zajedničkog pojma koji se krije u pozadini. U tu svrhu primijenjena je faktorska analiza, skup matematičko-statističkih postupaka koji omogućuju da se za manifestne varijable, među kojima postoji povezanost, utvrdi manji broj latentnih varijabli koje objašnjavaju tu povezanost.

Glavne karakteristike istraživanja, broj ispitanika (N = 352) koji su sudjelovali u istraživanju te broj čestica primijenjenog alata vrednovanja (10), ispunili su osnovne preduvjete za primjenu faktorske analize (35:1) (Hair i sur., 2014). Faktorska analiza primijenjena je kako bi se otkrile moguće veze između čestica (Cohen i sur., 2000) koristeći Kaiser-Meyer-Olkin mjeru parametra adekvatnosti uzorka (0.756>0.5) i Bartlettov test sferičnosti ($p \leq 0.001$). Rezultati su pokazali da su zadaci dovoljno povezani (determinant = .406> 0,00001, $p \leq 0,001$), ali ne pretjerano međuovisni (unutarnja povezanost< 0,8) (Tablica 18).

Tablica 18: Faktorska analiza - matrica korelacija, KMO i Bartletov test

Correlation Matrix[a]

		P: 2	P: 3	P: 4	P: 5	P: 6	P: 7	P: 8	P: 9	P: 10	P : 1
Correlation	P: 2 /	1,000	,177	,090	,122	,146	,180	,198	,202	,113	,159
	P: 3 /	,177	1,000	,026	,183	,200	,191	,290	,226	,065	,282
	P: 4 /	,090	,026	1,000	,100	,032	,104	-,023	,069	,059	,013
	P: 5 /	,122	,183	,100	1,000	,137	,104	,132	,187	,031	,131
	P: 6	,146	,200	,032	,137	1,000	,183	,130	,184	,124	,108
	P: 7 /	,180	,191	,104	,104	,183	1,000	,283	,452	,146	,199
	P: 8 /	,198	,290	-,023	,132	,130	,283	1,000	,326	,058	,141
	P: 9 /	,202	,226	,069	,187	,184	,452	,326	1,000	,089	,178
	P: 10	,113	,065	,059	,031	,124	,146	,058	,089	1,000	,014
	p1korigirano	,159	,282	,013	,131	,108	,199	,141	,178	,014	1,000
Sig. (1-tailed)	P: 2		,000	,047	,011	,003	,000	,000	,000	,018	,001
	P: 3	,000		,317	,000	,000	,000	,000	,000	,114	,000
	P: 4/	,047	,317		,030	,278	,026	,333	,100	,134	,403
	P: 5	,011	,000	,030		,005	,026	,007	,000	,285	,007
	P: 6	,003	,000	,278	,005		,000	,007	,000	,010	,022
	P: 7	,000	,000	,026	,026	,000		,000	,000	,003	,000
	P: 8	,000	,000	,333	,007	,007	,000		,000	,141	,004
	P: 9	,000	,000	,100	,000	,000	,000	,000		,049	,000
	P: 10	,018	,114	,134	,285	,010	,003	,141	,049		,398
	P : 1	,001	,000	,403	,007	,022	,000	,004	,000	,398	

a. Determinant = ,406

KMO and Bartlett's Test

Kaiser-Meyer-Olkin Measure of Sampling Adequacy.		,756
Bartlett's Test of Sphericity	Approx. Chi-Square	311,202
	Df	45
	Sig.	,000

Vrijednosti za osam čestica uglavnom zadovoljavaju kriterij dobre povezanosti (to je ~ 0,2- ~ 0,8, p <0,05). Dvije čestice (zadatak 4, zadatak 10) pokazale su veću povezanost s drugim česticama nego su to pokazale ostale čestice, ali ipak na nedovoljno značajnoj razini (p> 0,05). Također treba napomenuti da su te iste čestice (zadatak 4, zadatak 10) pokazale najniži indeks diskriminativnosti (<0,16).

Procedure analize glavnih komponenti te Oblinin rotacija s Kaiserovom normalizacijom istaknule su tri komponente, što nas navodi na razmišljanje da bi se čestice mogle povezati u tri faktora. Grafički prikaz plohe (Screen plot) potvrdio je takve rezultate (Slika 27).

Slika 27: Screen plot faktorske analize

Budući da su sve apsolutne vrijednosti izvan glavne dijagonale korelacijske matrice bile manje od 0,32, primijenjene su pravokutne rotacije, pri čemu je Varimax rotacija odabrana kao ona koja nudi najjednostavniju interpretaciju (najmanji broj složenih varijabli). Ovim načinom grupiranja objašnjeno je 45,738% ukupne varijance (Tablica 19).

Tablica 19: Faktorska analiza - tablica varijanci

Total Variance Explained

Component	Initial Eigenvalues			Extraction Sums of Squared Loadings			Rotation Sums of Squared Loadings[a]
	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %	Total
1	2,455	24,551	24,551	2,455	24,551	24,551	2,282
2	1,097	10,969	35,520	1,097	10,969	35,520	1,444
3	1,022	10,218	45,738	1,022	10,218	45,738	1,159

Extraction Method: Principal Component Analysis.

a. When components are correlated, sums of squared loadings cannot be added to obtain a total variance.

5.2.2. Analiza podataka

Kako bi se istražila mogućnost/sposobnost predloženog alata da izmjeri usvajanje koncepata računalnog razmišljanja apstrakcije, algoritamskog razmišljanja i dekompozicije, istražena je povezanost pojedinog zadatka sa svakim od faktora naznačenim rezultatima faktorske analize. Dobiveni rezultati u skladu su s predloženom strukturom alata za vrednovanje (Tablica 9 Poglavlje 4.3.2.) kojim se razlikuju temeljna i sekundarna znanja i prakse računalnog razmišljanja.

Rezultati faktorske analize istaknuli su postojanje triju faktora s kojima su povezane čestice alata vrednovanja (Tablica 20).

Tablica 20: Faktorska analiza - grupiranje čestica u faktore

Rotated Component Matrix[a]

	Component		
	1	2	3
Zadatak 7	,726		
Zadatak 9	,681		
Zadatak 8	,555		
Zadatak 10	,521		
Zadatak 6	,408		
Zadatak 2	,403		
Zadatak 1		,623	
Zadatak 3		,607	
Zadatak 5		,521	
Zadatak 4		-	,828

Extraction Method: Principal Component Analysis.
Rotation Method: Varimax with Kaiser Normalization.
a. Rotation converged in 6 iterations.

Analizirajući povezanost čestica s pojedinim faktorom napravljena su grupiranja čestica prema njihovim zajedničkim obilježjima. Temeljne vještine i prakse računalnog razmišljanja povezane s istaknutim faktorima faktorske analize, grupirani su prema faktorskim opterećenjima u tri razine (jača, umjerena, slabija). U ovisnosti o veličini uzorka, faktorska

opterećenja (engl. Factor loadings) jednaka ili veća od 0,3 smatraju se značajnim (Hair i sur., 2014) (Tablica 21).

Tablica 21: Grupiranje zadataka alata vrednovanja u skladu s rezultatima faktorske analize

Povezanost s faktorom (faktorsko opterećenje)	Faktor 1 (algoritamsko razmišljanje)	Faktor 2 (apstrakcija)	Negrupirana čestica*
Jača povezanost s faktorom (>0.62)	• Uređivanje i nadogradnja algoritamskog rješenja (Algoritamsko razmišljanje, zadatak 7) • Stvaranje novih rješenja na temelju poznatih problema (Algoritamsko rješenje, dekompozicija, zadatak 9) • Razumijevanje problema (Apstrakcija, zadatak 7)	• Razumijevanje problema (Apstrakcija, zadatak 1) • Rad s logičkim vrijednostima (Apstrakcija, zadatak 3)	• Rad sa strukturom grananja (Algoritamsko razmišljanje, zadatak 4)
Umjerena povezanost s faktorom (0.52<…<0.62)	• Rastavljanje problema na manje složene već poznate probleme (Dekompozicija, zadatak 8) • Rad s varijablama i izlazom, (Apstrakcija, zadatak 10)	• Rad sa strukturom grananja (Algoritamsko razmišljanje, zadatak 5)	
Slabija povezanost s faktorom (0.3<…<0.52)	• Predviđanje ponašanja programa (Algoritamsko razmišljanje, zadatak 6) • Identifikacija ključnih obilježja/ograničenja (Apstrakcija, zadatak 2)		

* negrupirani zadaci zbog nedovoljnog broja zadatka povezanih s istaknutim faktorom

Prvi faktor faktorske analize istaknuo je čestice čija se povezanost temelji na praksi i vještinama algoritamskog razmišljanja (uređivanje i nadograđivanje algoritamskih rješenja, stvaranje novih rješenja na temelju poznatog problema, predviđanje ponašanja programa). Relativno manje snažnu, ali još uvijek značajnu povezanost s prvim faktorom, pokazale su čestice u kojima je naglasak na vještinama koje su povezane s apstrakcijom (rad s varijablama i izlazom, razumijevanje problema, prepoznavanje ključnih značajki problema).

Povezanost s drugim faktorom pokazale su čestice kojima se provjeravaju vještine i znanja povezana s apstrakcijom (razumijevanje problema, rad s logikom). Manju, ali ipak značajnu povezanost s drugim faktorom, pokazale su i čestice u kojima je naglasak na vještinama rada s algoritamskom strukturom grananja (koncept algoritamsko razmišljanje).

Treći faktor pokazao je snažnu povezanost sa samo jednom česticom (zadatak 4), u kojoj se naglašavaju vještine rada s algoritamskim strukturama grananja (jednostavnim i složenim). Ovo je iznenađujući rezultat jer se moglo očekivati da će ove vještine iskazati snažnu povezanost s prvim faktorom, u kojem prevladavaju vještine i znanja vezana uz koncept algoritamskog razmišljanja. Budući da je ovaj zadatak (zadatak 4) već pokazao izuzetno loš indeks diskriminativnosti, upitno je koliko su rezultati utjecali na neočekivano povezivanje istaknute vještine (rad sa strukturama grananja) s trećim faktorom, umjesto s prvim faktorom koji se uglavnom odnosi na koncepte algoritamskog razmišljanja. Iako se treći koncept, koji smo razmatrali u ovom alatu vrednovanja - koncept dekompozicije, nije istaknuo kao pojedinačni faktor u ovoj analizi, svejedno ga možemo snažno povezati s konceptom algoritamskog razmišljanja, kao što ga možemo povezati i s konceptom apstrakcije.

Jedan od razloga zbog kojeg se, suprotno očekivanom, koncept dekompozicije nije istaknuo kao pojedinačni faktor ove analize, možda se može objasniti činjenicom da je razmatran u samo dvije od ukupno deset čestica alata vrednovanja. Drugi razlog neočekivanog rezultata možda je uzrokovan osnovnom pretpostavkom uspješnog rješavanja zadataka vrednovanja koji se odnose na koncept dekompozicije (zadatak 8, zadatak 9) koja traži od učenika posjedovanje dovoljno razvijene vještine algoritamskog mišljenja (stvaranje novih rješenja na temelju poznatih problema, uređivanje i nadograđivanje algoritamskih rješenja), kao i posjedovanje vještina i znanja povezanih s konceptom apstrakcija (razumijevanje problema, razmišljanje na različitim razinama apstrakcije, rad s varijablama).

S obzirom na navedene rezultate možemo zaključiti da su pojmovi algoritamskog razmišljanja i apstrakcije u ovom vrednovanju jasno razlučeni i prepoznati faktorskom analizom. Koncept dekompozicije u alatu vrednovanja predstavljen je samo s dvije čestice što se pokazalo nedovoljnim za faktorsku analizu koja zahtijeva najmanje tri čestice. Ipak, ne može se reći da pojedine čestice alata vrednovanja nisu uspjele adresirati koncept dekompozicije (Poglavlje 5.6.2., str.119). Nadogradnja alata odgovarajućim česticama koje se odnose na koncept dekompozicije, svakako bi pozitivno utjecala da se i koncept dekompozicije izdvoji kao zasebni faktor faktorskom analizom rezultata vrednovanja.

5.3. Utjecaj ranijeg učenja programskog jezika na uspjeh pri vrednovanju računalnog razmišljanja

5.3.1. Opis istraživanja

Jedan od ključnih ciljeva ovog istraživanja bio je istražiti i utvrditi postoji li ovisnost alata vrednovanja o programskim jezicima te okruženjima za programiranje koji se najčešće koriste u RH. S obzirom da je u nastavni proces predmeta Informatika u RH uključeno nekoliko različitih programskih jezika i okružja za programiranje, veoma je važno da predloženi model pokaže kompatibilnost s programskim jezicima i okružjima koji su u svakodnevnoj nastavnoj praksi.

Iz tog razloga, prilikom stvaranja čestica vrednovanja jedna od osnovnih pretpostavki bila je da tekst i slike, koje se koriste u česticama alata, moraju biti odgovarajuće kvalitete i konteksta kako bi se izbjegla svaka ovisnost o specifičnoj sintaksi i strukturi nekog programskog jezika. Istovremeno, svaka je čestica alata vrednovanja morala biti dovoljno jasna i intuitivna kako ne bi odvraćala pažnju učenika od rješavanja problema.

Osnovne značajke ovog modela vrednovanja prvotno su testirane uvodnim istraživanjem, uz primjenu mjernog instrumenta prilagođenog za programski jezik Python. Obećavajući rezultati početnog vrednovanja potaknuli su stvaranje novog modela vrednovanja neovisnog o programskom alatu ili okružju koji je temeljen na ECD pristupu (Bubica i Boljat, 2018).

U sklopu ovog dijela istraživanja djelomično će se i istražiti valjanost početne hipoteze (H1) koja previđa postojanje korelacije između uspjeha kojeg postižu učenici šestih razreda osnovnih škola pri vrednovanju koncepata računalnog razmišljanja i programiranja.

5.3.2. Analiza podataka

Analiza podataka izvršena je s ciljem ispitivanja (ne)osjetljivosti predloženog alata za vrednovanje koncepata računalnog razmišljanja. Analizirani su rezultati učenika s obzirom na njihovo ranije poznavanje pojedinog programskog jezika ili okružja za programiranje. Podatci učenika (uspjeh) nisu udovoljili kriteriju normalnosti (Kolmogorov – Smirnov, kosost = 0,388, kurtoza = -0,041, $p \leq 0,001$) pa su se u daljnjoj analizi koristile neparametrijske metode. S obzirom da su neki učenici svoj prvi susret s programiranjem ostvarili kroz grafičko okruženje za programiranje Scratch te su dalje nastavili svoj rad u programskom jezikom Pythonu ili Logo, istraženo je postoji li razlika među njihovim rezultatima.

Ispitanici su razvrstani u četiri skupine ovisno o programskom jeziku ili kombinaciji programskih jezika koje su koristili tijekom usvajanja koncepata programiranja (Tablica 22).

Tablica 22: Grupiranje učenika prema programskim jezicima koje poznaju

Grupa učenika	Programski jezici	Broj učenika
grupa_1	Prvo Scratch pa Python	26
grupa_2	Prvo Scratch pa Logo	31
grupa_3	Python	183
grupa_4	Logo	112

Grupe su formirane prema inicijalnim podatcima o učenicima, odnosno podatcima iz ulaznih anketa u kojima su učenici naveli sve programske jezike i okružja s kojima su se susreli u nastavnom procesu. Kruskal-Wallisova metoda pokazala je da nije bilo značajne razlike između rezultata učenika u ovisnosti o korištenom programskom jeziku ili kombinaciji pojedinog jezika i okružja ($\chi 2$ (2) = 2.458, p = 0.483, sa srednjim rangom 176,42 za grupu_1, 203,21 za grupu_2 , 171,35 za grupu_3 i 175,14 za grupu_4) (Tablica 23-24).

Tablica 23: Kruskal Wallis test – srednji rangovi – razlike s obzirom na grupe programskih jezika

Ranks

Varijabla	Grupa	N	Mean Rank
Uspjeh na vrednovanju računalnog razmišljanja	Python, Scratch	26	176,42
	Logo, Scratch	31	203,21
	Python	183	175,14
	Logo	112	171,35
	Total	352	

Tablica 24:

Tablica 24: Kruskal Wallis test – statistika - razlike s obzirom na grupe programskih jezika

Test Statistics[a,b]

	Uspjeh na vrednovanju računalnog razmišljanja
Chi-Square	2,458
Df	3
Asymp. Sig.	,483

a. Kruskal Wallis Test
b. Grouping Variable: jezik

Daljnjim grupiranjem istražilo se postoji li utjecaj pojedinačnog programskog jezika ili okružja na uspjeh učenika pri vrednovanju računalnog razmišljanja (Tablice 25-30).

Tablica 25: Kruskal Wallis test – srednji rangovi razlike s obzirom na programski jezik Python i ostale jezike

Ranks

	pythonDANE	N	Mean Rank	Sum of Ranks
Uspjeh na vrednovanju računalnog razmišljanja	ostali jezici	143	178,26	25491,00
	Python	209	175,30	36637,00
	Total	352		

Tablica 26: Kruskal Wallis test – statistika - razlike s obzirom na programski jezik Python i ostale jezike

Test Statistics[a]

	Uspjeh na vrednovanju računalnog razmišljanja
Mann-Whitney U	14692,000
Wilcoxon W	36637,000
Z	-,268
Asymp. Sig. (2-tailed)	,788

a. Grouping Variable: pythonDANE

Tablica 27: Kruskal Wallis test – srednji rangovi - razlike s obzirom na programski jezik Logo i ostale jezike

Ranks

	Logo DANE	N	Mean Rank	Sum of Ranks
Uspjeh na vrednovanju računalnog razmišljanja	ostali jezici	209	175,30	36637,00
	Logo	143	178,26	25491,00
	Total	352		

Tablica 28: Kruskal Wallis test – statistika - razlike s obzirom na programski jezik Logo i ostale jezike

Test Statistics[a]

	Uspjeh na vrednovanju računalnog razmišljanja
Mann-Whitney U	14692,000
Wilcoxon W	36637,000
Z	-,268
Asymp. Sig. (2-tailed)	,788

a. Grouping Variable: logoDANE

Tablica 29: : Kruskal Wallis test – srednji rangovi - razlike s obzirom na programski jezik Scratch i ostale jezike

Ranks

	scratchDANE	N	Mean Rank	Sum of Ranks
Uspjeh na vrednovanju računalnog razmišljanja	Ostali jezici	295	173,70	51241,50
	Scratch	57	190,99	10886,50
	Total	352		

Tablica 30: Kruskal Wallis test – statistika - razlike s obzirom na programski jezik Scratch i ostale jezike

Test Statistics[a]

	Uspjeh na vrednovanju računalnog razmišljanja
Mann-Whitney U	7581,500
Wilcoxon W	51241,500
Z	-1,175
Asymp. Sig. (2-tailed)	,240

a. Grouping Variable: scratchDANE

Rezultati su pokazali da nema statistički značajne razlike između grupa učenika koje su učile programiranje putem okružja Scratch (Mann–Whitney 7581.500, p=0.240) te programskih jezika Logo (Mann–Whitney 14692, p=0.788) i Python (Mann–Whitney 14692, p=0.788).

Iako su rezultati ranijih istraživanja koja razmatraju prediktore uspjeha u programiranju, prvi programski jezik prepoznali kao jedan od značajnih faktora pri postizanju uspjeha u radu programera početnika, neosjetljivost mjernog instrumenta na prvi programski jezik smatra se pozitivnim obilježjem ovog pristupa vrednovanja. Ovim se rezultatom ispunilo početno očekivanje da uspjeh kojeg učenici postižu na ovom alatu vrednovanja, neće biti pod utjecajem programskog jezika ili okružja kojeg su učenici koristili prilikom usvajanja koncepata računalnog razmišljanja. Takav rezultat govori u prilog prikladnosti alata vrednovanja za primjenu u svakodnevnim nastavnim situacijama.

Primjena različitih programskih jezika i okružja programiranja u procesu učenja i podučavanja u školama koje su sudjelovale u ovom istraživanju te nedostupnost informacija o stvarnim provjerama programiranja koje su se primjenjivale tijekom školske godine, onemogućila je ispitivanje valjanosti hipoteze (H1) istraživanjem povezanosti rezultata dvaju vrednovanja. Valjanost hipoteze ispitat će se istraživanjem povezanosti sadržaja koji su uključeni u navedena vrednovanja o čemu će više biti riječi u sljedećim poglavljima.

5.4. Utjecaj duljine podučavanja te odabranih sadržaja programiranja tijekom školske godine na uspjeh pri vrednovanju računalnog razmišljanja

5.4.1. Opis istraživanja

Za određivanje kvalitete razvijenog alata vrednovanja računalnog razmišljanja korisno je istražiti koje su koncepte i prakse računalnog razmišljanja i programiranja učili učenici koji su sudjelovali u istraživanju te koliko su ih dugo učili. Nakon provedenog vrednovanja sa svojim učenicima, učitelji su imali otprilike dva tjedna kako bi analizirali alat vrednovanja te ispunili izlaznu anketu za učitelje (PRILOG 3). U sklopu ankete, između ostalog, označili su one koncepte računalnog razmišljanja i programiranja koje su do tada obradili sa svojim učenicima. Od devet uključenih učitelja u istraživanje, jedan učitelj (Škola A i D) nije ispunio upitnik.

5.4.2. Analiza podataka

Popis koncepata i praksi računalnog razmišljanja i programiranja koje su učenici do tada učili tijekom svog nastavnog rada, prikazan je sljedećom tablicom (Tablica 31).

Tablica 31: Popis koncepata računalnog razmišljanja i programiranja obrađenih tijekom nastavnog procesa po školama

Koncepti/praksa	Škola B	Škola C	Škola E	Škola F	Škola G	Škola H	Škola I	Škola J
Pojam algoritma	+	+	+		+	+		+
Rad s izlaznim vrijednostima (npr, print...)	+	+	+		+	+	+	+
Naredbe za crtanje (kornjačina grafika)		+	+	+	+		+	+
Rad s ulaznim vrijednostima (npr. naredba input)	+	+	+		+	+		
Izrada i uređenje blok dijagrama nekog algoritma		+	+		+	+		
Jednostavno grananje (npr, naredba if … then, if ….)	+	+	+		+	+	+	+
Složeno grananje (npr. naredba if...then...else..., if..else....)	+	+	+		+	+		
Ugniježđeno grananje (npr. naredba if....elif....)		+	+		+			

Petlje s određenim brojem ponavljanja (npr. naredba for...., foreach...)		+	+	+	+	+	+	
Petlje s uvjetnim ponavljanjem (npr. naredba while...)		+				+		
Potprogrami				+			+	
Programski jezik	P	P	P	L	P	P	L	L
Sati učenja	12	26	24	14	24	25	24	14
Broj učenika	48	53	36	64	26	24	31	48

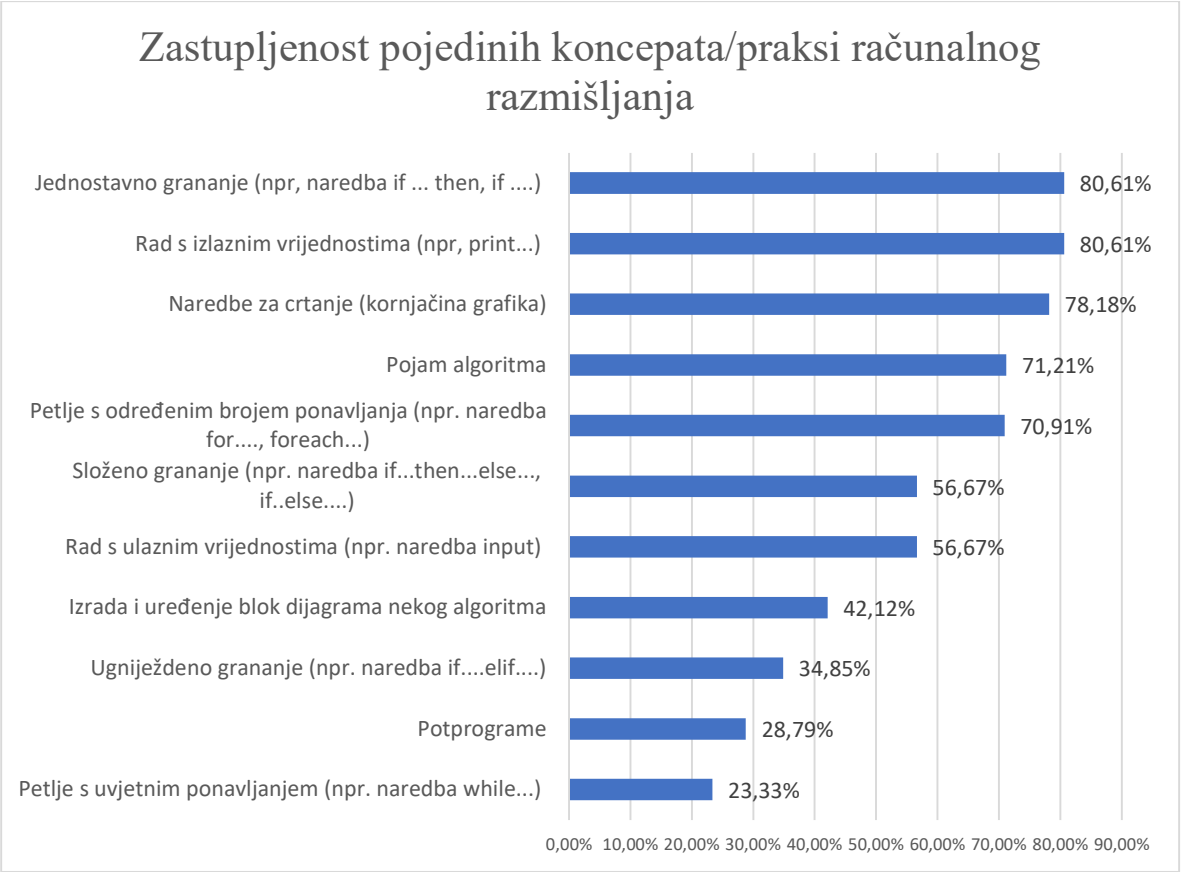
* P-Python, L-Logo,

** Škola A(3) i D(19) (nisu dostavljeni podatci)

U tabličnom prikazu uključeni su i podatci o broju sati unutar Godišnjeg nastavnog plana i programa tijekom kojih je učitelj obrađivao odabrane sadržaje računalnog razmišljanja i programiranja. Također, naveden je i programski jezik koje se pri tome koristio te ukupan broj učenika pojedine škole.

S obzirom da su učenici u procesu podučavanja upoznali različite koncepte računalnog razmišljanja i programiranja, istražilo se postoji li razlika među rezultatima učenika s obzirom na koncepte koje su učili. Iz priložene tablice vidljivo je da se niti jedan od koncepta ili praksi računalnog razmišljanja i programiranja nije obrađivao u svim promatranim školama. Najveći broj učenika upoznao se s konceptom jednostavnog grananja i radom s izlaznim vrijednostima (80,61%), naredbama crtanja u kornjačinoj grafici (78,18%), petljama s određenim brojem ponavljanja (70,91%) te pojmom algoritma (71,21%).

Svaki drugi učenik upoznao je koncept složenog grananja te rad s ulaznim vrijednostima (56,67%). Barem svaki treći učenik izrađivao je i uređivao blok dijagram nekog algoritma (42,12%) te upoznao pojam ugniježdenog grananja (34,85%). Najmanji broj učenika, otprilike svaki četvrti učenik koji je sudjelovao u ovom istraživanju, upoznao je pojam potprograma (28,79%) te petlje s uvjetnim ponavljanjem (23,33%) (Slika 28).

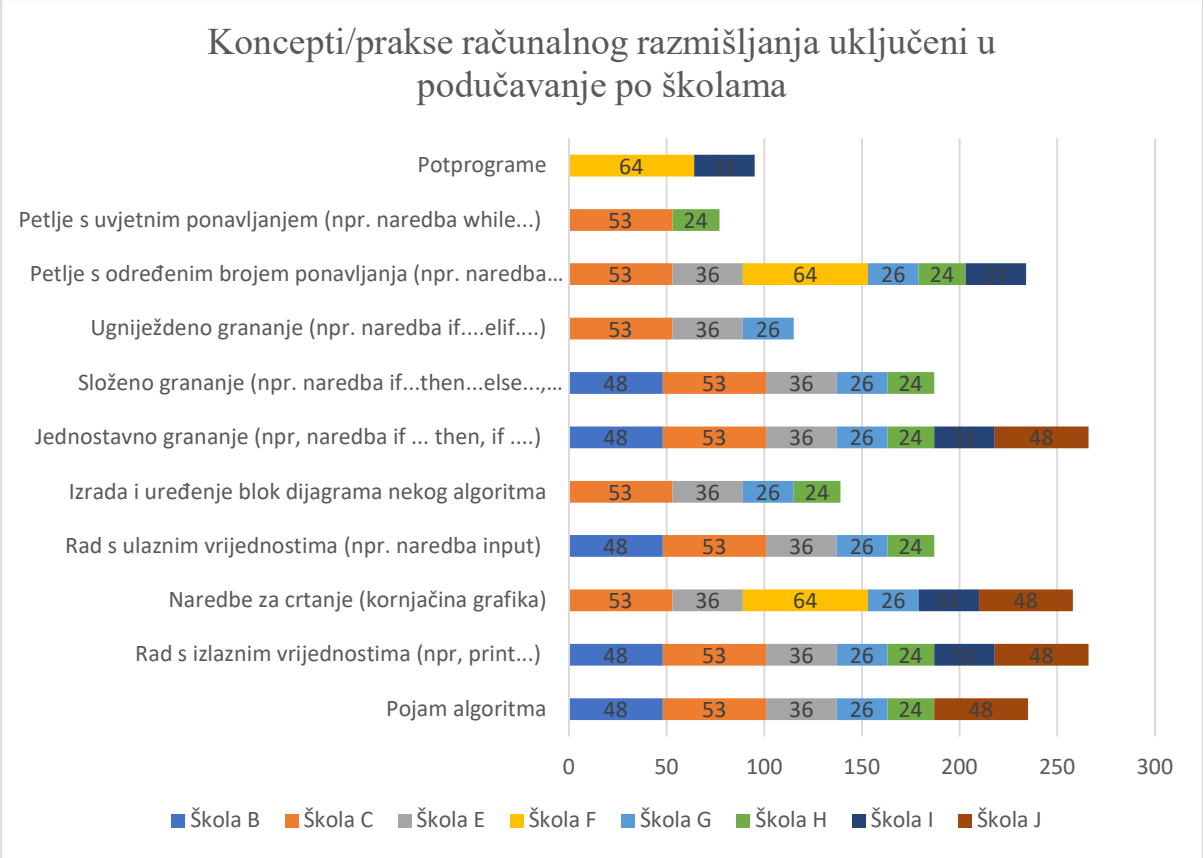


Slika 28: Zastupljenost pojedinih koncepata/praksi računalnog razmišljanja kod učenika

Zanimljivo je bilo analizirati postoji li povezanost između programskog jezika u kojem su učenici radili te koncepata i praksi računalnog razmišljanja koje su tijekom nastave usvajali. Prema navedenim rezultatima očigledno je da se koncept potprograma obrađivao samo u školama koje su odabrale rad u programskom jeziku Logo dok se istovremeno u tim školama najmanje obrađivao pojam algoritma. Rad s ulaznim vrijednostima obrađivali su svi, a izradu i uređenje blok dijagrama nekog algoritma samo neki učenici, no u oba slučaja radilo se isključivo u programskom jeziku Python.

Iz navedenih rezultata može se zaključiti da odabir programskog jezika za rad s konceptima računalnog razmišljanja i programiranja ipak utječe na odabir koncepata i praksi koji se pri time usvajaju. U školama u kojima se radi u programskom jeziku Logo, naglasak je na potprogramima, petljama s određenim brojem ponavljanje te naredbama za crtanje u kornjačinoj grafici. U školama u kojima se radi u programskom jeziku Python, uglavnom se radi o većem broju uključenih koncepata i praksi računalnog razmišljanja i programiranja.

Sljedeća slika prikazuje popis koncepata računalnog razmišljanja koji su bili uključeni u proces podučavanja, raspodijeljeni po školama te s istaknutim brojem učenika koji su bili uključeni u proces podučavanja (Slika 29).



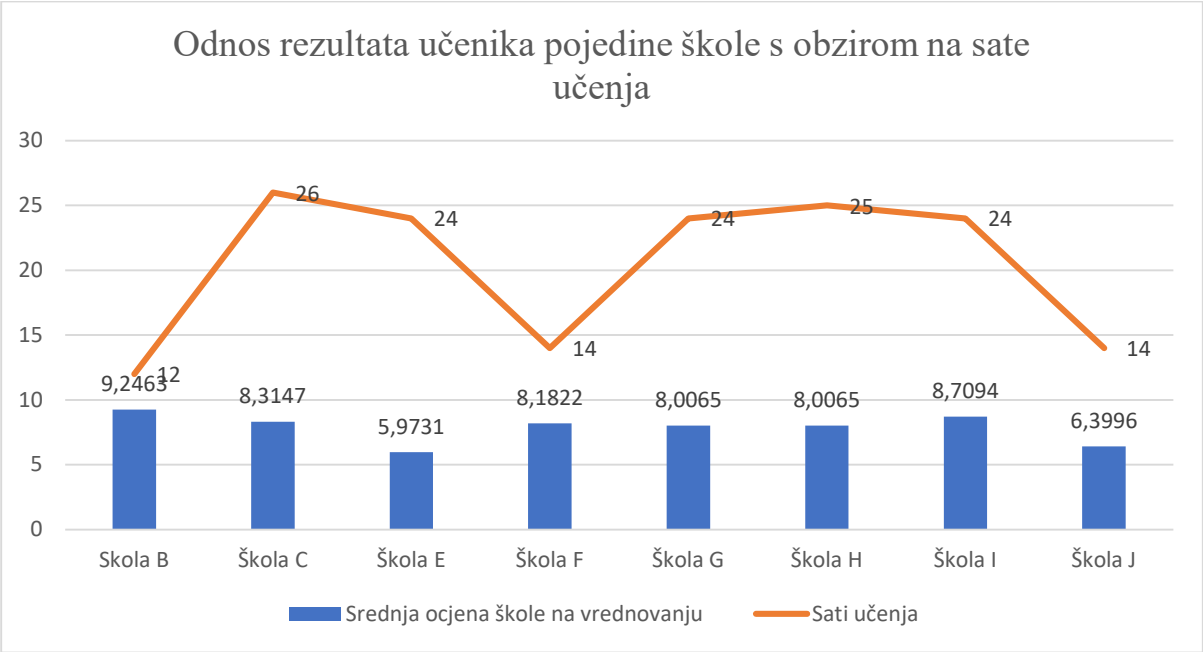
Slika 29: popis koncepata računalnog razmišljanja koji su bili uključeni u proces podučavanja raspodijeljeni po školama te s istaknutim brojem učenika

Učenici su u različitim školama sadržaje programiranja učili u vremenskim periodima različite duljine. Nužno je bilo istražiti je li duljina podučavanja (broj_sati) utjecala na uspjeh učenika pojedine škole na vrednovanju.

Rezultati su pokazali da količina vremena koje je učitelj posvetio konceptima računalnog razmišljanja i programiranja nije bila povezana s rezultatima učenika na ovom vrednovanju (Pearson r=-0.72, p=0.191>0.05). Negativna vrijednost koeficijenta korelacije ukazuje da se promatrane vrijednosti, sati učenja te uspjeh vrednovanja računalnog razmišljanja, razvijaju u suprotnim smjerovima: kada jedna vrijednost raste, druga pada i obrnuto. To bi značilo da veći broj sati proveden u podučavanju programiranja može negativno djelovati na uspjeh

vrednovanja računalnog razmišljanja. S obzirom da rezultat nije statistički značajan, može se samo reći da ne postoji linearna korelacija promatranih vrijednosti na promatranom uzorku.

Sljedeći grafikon prikazuje odnos srednjeg uspjeha učenika pojedine škole s obzirom na broj sati koji je u procesu podučavanja utrošen na koncepte računalnog razmišljanja, odnosno programiranja u nekom programskom jeziku (Slika 30). Iako je prirodno očekivati da će dulje vrijeme zadržavanja na odabranim konceptima rezultirati boljim uspjehom učenika na vrednovanju, rezultati pokazuju da to ipak nije tako. Najbolji srednji uspjeh na vrednovanju postigli su učenici škole koja je potrošila samo dvanaest sati (12) na podučavanje takvog sadržaja. Drugi najbolji rezultat postigli su učenici škole koja je odabrane sadržaje izučavala dvadeset i četiri sata (24). Istovremeno, među najlošijim rezultatima nalaze se dvije škole koje su potrošile značajno različit broj sati na poučavanje odabranih koncepata računalnog razmišljanja i programiranja (Škola E – 24 sata, Škola J – 14 sati).



Slika 30: Odnos rezultata učenika pojedine škole s obzirom na sate učenja

Povezanost utjecaja duljine učenja koncepata računalnog razmišljanja s uspjehom na vrednovanju istražena je i linearnom regresijom (R^2=0.005, F(1,328)=1,721, p=0.191>0.05). Rezultati su pokazali da ne postoji statistički značajna povezanost između dvaju promatranih varijabli (sati učenja, uspjeh na vrednovanju računalnog razmišljanja).

Iz prikazanih rezultata moglo bi se zaključiti da je u vrijeme provođenja istraživanja u nastavnoj praksi, prema aktualnom Nastavnom planu i programu, stavljen puno veći naglasak na same

vještine programiranja i rad u odabranom programskom jeziku/okružju nego na koncepte računalnog razmišljanja kao što su npr. apstrakcija, dekompozicija. Gledajući s takvog aspekta, dobiveni rezultati koji se odnose na sate učenja, mogu se smatrati očekivanim.

Uspoređujući rezultate prikazane u poglavljima u kojima su predstavljeni rezultati istraživanja povezanosti uspjeha na ovom vrednovanju, sa sadržajima programiranja koji se podučavaju, programskim jezicima ili okružjima te primjerenosti alata za vrednovanje koncepata apstrakcije, algoritamskog razmišljanja i dekompozicije, može se izvući djelomični zaključak vezano uz valjanost hipoteze H1 (*Postoji korelacija između uspjeha kojeg postižu učenici šestih razreda osnovnih škola pri vrednovanju koncepata računalnog razmišljanja i programiranja).* S obzirom da nije bilo moguće izravno usporediti rezultate koje bi učenici postigli na alatu vrednovanja i odgovarajućoj provjeri iz programiranja, napravljena je usporedba sadržaja programiranja koje su učenici usvojili tijekom procesa podučavanja. Sadržaji koje su učenici učili značajno su ovisili o programskom jeziku ili okružju koje se koristilo pri podučavanju, no u svakom slučaju radilo se o sadržajima koji su sastavni dio koncepata računalnog razmišljanja uključenih u alat vrednovanja iz ovog istraživanja. Sposobnost algoritamskog razmišljanja, rad s apstrakcijom te dekompozicija predstavljaju preduvjet stvaranja nekog računalnog rješenja, neovisno o programskom jeziku i okružju u kojem se radi. Možemo reći da postoji snažna veza između sadržaja koji su uključeni u alat iz ovog vrednovanja te sadržaja koji bi trebali biti uključeni u provjeru iz programiranja usklađenu s ishodima predmetnog kurikula šestog razreda.

5.5. Usporedba rezultata pri vrednovanju računalnog razmišljanja s rezultatima postignutima na zadacima Dabar (Bebras) izazova

Za svaki alat vrednovanja iznimno je važno provjeriti mjeri li alat upravo one sadržaje koji se i očekuju, odnosno vrijedi li sadržajna valjanost mjernog instrumenta. U tu svrhu odabran je drugi poznati mjerni instrument koji se koristi za vrednovanje računalnog razmišljanja te se provjerilo postoji li povezanost odabranih mjernih instrumenata.

Za potrebe ovog istraživanja kao dodatni mjerni instrument odabran je test natjecanja Dabar. Dabar (engl. Bebras) je međunarodno natjecanje u kojem je do sada sudjelovalo pedesetak država, a koje ima za cilj promicati računalno razmišljanje i informatiku kod učenika svih dobnih skupina (Bebras Challenge, 2019). Svaki zadatak natjecanja Dabar napisan je u obliku zagonetke ili priče koja u pozadini krije neki temeljni koncept računalnog razmišljanja ili informatike općenito.

U RH natjecanje Dabar provodi se počevši od 2016. godine u obliku online natjecanja putem sustava za udaljeno učenje Loomen. Učenike u natjecanje uključuju njihove škole i učitelji, a sudjelovanje je omogućeno isključivo putem jedinstvenih AAI identiteta svakog učenika (Suradnici u učenju, 2020).

5.5.1. Opis istraživanja

Za potrebe istraživanja sadržajne valjanosti razvijenog alata vrednovanja analizirani su koncepti i sadržaji računalnog razmišljanja koji su uključeni u oba vrednovanja. Tablica 9 (Poglavlje 4.3.2.) prikazuje strukturu razvijenog alata vrednovanja za potrebe ovog istraživanja, a Tablica 32 strukturu vrednovanja natjecanja Dabar 2017. Svaki zadatak natjecanja Dabar kreiran je od strane informatičkog stručnjaka te višestruko vrednovan od strane drugih informatičkih stručnjaka tijekom godišnjeg okupljanja udruga koje provode to natjecanje (Bebras Challenge, 2019; Suradnici u učenju, 2020).

U natjecanju Dabar mogu se koristiti samo odabrani zadaci koji su dobili pozitivne povratne informacije od strane informatičkih stručnjaka, pri čemu svaka država ima mogućnosti vlastitog odabira zadataka koje će uvrstiti u godišnje natjecanje. Svaki je zadatak opisan svojom računalnom povezanošću pri čemu se ističe koncept računalnog razmišljanja odnosno informatički sadržaj koji je uključen u zadatak (Tablica 32).

Tablica 32: Struktura testa vrednovanja natjecanja Dabar 2017

Čestice alata vrednovanja (oznaka)	Vrsta pitanja	Računalna povezanost i ključni pojmovi
DAB1	Pitanje višestrukog odgovora	Binarni brojevni sustav Apstrakcija
DAB2	Pitanje kratkog odgovora	Apstrakcija Algoritamsko razmišljanje: algoritamska struktura grananja
DAB3	Pitanje višestrukog odgovora	Algoritamsko razmišljanje: praćenje koda, algoritamska struktura slijeda, predviđanje izvršavanja programa
DAB4	Pitanje uparivanja odgovora	Algoritamska struktura grananja, algoritamska struktura ponavljanja, praćenje i predviđanje izvršavanja programa Apstrakcija (modeliranje ulaznih podataka – grafički prikaz);
DAB5	Pitanje višestrukog odgovora	Algoritamsko razmišljanje: praćenje koda, algoritamska struktura slijeda, grananja i ponavljanja, algoritam
DAB6	Pitanje višestrukog odgovora	Algoritamsko razmišljanje: praćenje koda, struktura grananja, slijeda Apstrakcija: Razumijevanje problema
DAB7	Pitanje višestrukog odgovora	Apstrakcija: logičko razmišljanje, rad s logičkim vrijednostima Algoritamsko razmišljanje: algoritamska struktura grananja, paralelizam
DAB8	Pitanje kratkog odgovora	Algoritamsko razmišljanje: algoritamska struktura ponavljanja
DAB9	Pitanje višestrukog odgovora	Kriptografija
DAB10	Pitanje kratkog odgovora	Algoritamsko razmišljanje: algoritamska struktura grananja; Apstrakcija: rad s logičkim vrijednostima
DAB11	Pitanje višestrukog odgovora	Optimizacija, traženje minimalne udaljenosti
DAB12	Pitanje višestrukog odgovora	Apstrakcija: logičko zaključivanje – rad s logičkim operatorima (Booleova algebra) Algoritamskog razmišljanje: algoritamska struktura grananja
DAB13	Pitanje uparivanja odgovora	Apstrakcija: logičko zaključivanje, teorija skupova
DAB14	Pitanje višestrukog odgovora	Paralelno programiranje, simulacija
DAB15	Pitanje kratkog odgovora	Enkripcija (kriptografija)

Daljnjom analizom uspoređeni su sadržaji uključeni u oba vrednovanja te istražena njihova povezanost s obrazovnim ishodima predmetnog kurikula. Iz prikazane sadržajne strukture vrednovanja natjecanja Dabar može se uočiti da uključeni koncepti računalnog razmišljanja te informatički sadržaji općenito nisu u potpunosti usklađeni s obrazovnim ishodima kurikula predmeta Informatika za šesti razred osnovne škole (binarni brojevni sustav, kriptografija, optimizacija, traženje minimalne vrijednosti).

Zbog navedenog bilo je nužno prepoznati i odabrati čestice vrednovanja koje imaju temelj u obrazovnim ishodima kurikula predmeta Informatika za šesti razred te istražiti povezanost takvog podskupa čestica s razvijenim alatom vrednovanja u ovom istraživanju.

S obzirom da je dio učenika koji je sudjelovao u ovom istraživanju sudjelovao i u natjecanju Dabar, analizirana je i povezanost dvaju mjernih instrumenata na tom prilagođenom uzorku učenika (N=49). Uspoređena su statistička obilježja oba vrednovanja i na njihovom cjelovitom uzroku.

5.5.2. Analiza podataka

Sljedeća tablica daje usporedni prikaz čestica promatranih alata vrednovanja s obzirom na praksu i vještine računalnog razmišljanja uključenih u vrednovanje. Odabrane su sve prakse i vještine računalnog razmišljanja koje su uključene u alat vrednovanja razvijen u ovom istraživanju te su čestice oba vrednovanja razvrstane u tablicu prema odgovarajućoj praksi i vještini računalnog razmišlja na koju se čestica pojedinog vrednovanja odnosi (Tablica 33).

Tablica 33: Usporedba praksi i vještina pojedinih koncepata računalnog razmišljanja uključenih u alat vrednovanja te natjecanje Dabar

Koncepti računalnog razmišljanja	Praksa i vještine odabranog koncepta	Čestice predloženog alata CT vrednovanja	Čestice natjecanja Dabar
Algoritamsko razmišljanje	**Praćenje koda**	**CT1, CT5, CT6, CT7, CT10**	**DAB3, DAB4, DAB5, DAB6**
	Predviđanje ponašanja programa	**CT1, CT2, CT6, CT8, CT10**	**DAB4, DAB5**
	Algoritamska struktura slijeda	**CT1**	**DAB5, DAB6**

	Algoritamska struktura grananja	**CT1, CT5,**	**DAB2, DAB3, CT3, DAB4, DAB5, DAB6, DAB7, DAB10, DAB12**
	Algoritamska struktura ponavljanja	**CT1**	**DAB4, DAB8**
	Uređivanje i nadogradnja algoritamskog rješenja	**CT7**	**-**
	Stvaranje algoritamskog rješenja	**CT9**	**-**
Apstrakcija	**Razumijevanje problema**	**CT1, CT2, CT4, CT7**	**DAB6**
	Uporaba varijabli	**CT7, CT9, CT10**	**DAB1, DAB2, DAB4**
	Rad s logičkim vrijednostima	**CT3**	**DAB7, DAB10, DAB12**
	Rad s različitim razinama apstrakcije	**CT4, CT9**	**-**
Dekompozicija	**Rastavljanje problema na manje složene**	**CT8**	**-**

Uvidom u statističke podatke vrednovanja zadataka natjecanja Dabar vidi se da se čak deset zadataka može smatrati preteškim zadacima (indeks težine >0.7) dok izuzetno lagani zadaci nisu uključeni u vrednovanje (Tablica 34). Četiri zadatka imaju dobar indeks diskriminativnosti (0.25 – 0.34), četiri zadatka prihvatljiv indeks diskriminativnosti (0.16 – 0.24) dok čak sedam zadataka ima loš indeks diskriminativnosti (<0.16).

Tablica 34: Statistička obilježja mjernog instrumenta natjecanja Dabar na cjelovitom uzorku (N=1892)

Čestica alata	Indeks težine	Indeks diskriminativnosti	Čestica alata	Indeks težine	Indeks diskriminativnosti
1	35,50	31,83	9	54,14	20,42
2	35,65	29,11	10	80,58**	17,33
3	77,80**	3,73	11	72,75**	30,89
4	71,49**	12,12	12	78,61**	15,37
5	88,29**	1,33	13	97,23**	12,53
6	64,13	19,53	14	69,07	30,4
7	73,56**	3,98	15	89,40**	27,78
8	73,08**	15,17			

** - vjerojatno pretežak zadatak, * - vjerojatno prelagan zadatak

S obzirom na povezanost praksi i vještina odabranih koncepata računalnog razmišljanja (apstrakcija, algoritamsko razmišljanje, dekompozicija) za daljnju analizu podataka odabran je podskup pitanja natjecanja Dabar koja su bila najviše usklađena sa zadacima vrednovanja ovog istraživanja. To su zadaci sa sljedećim oznakama: DAB1, DAB2, DAB3, DAB4, DAB5, DAB6, DAB10, DAB12 (PRILOG 5). Zadaci s oznakom DAB7, DAB8, DAB9, DAB11, DAB13, DAB14 i DAB15 odnose se na sadržaje kao što su kriptografija, binarni brojevni sustav, optimizacija, simulacija i paralelizam, koji nisu usklađeni s ishodima predmetnog kurikula za odabrani uzrast učenika.

Usporedbom statističkih podataka na cjelovitom uzorku oba vrednovanja vidi se da pouzdanost vrednovanja zadacima tipa Dabar nema zadovoljavajuću pouzdanost (α=0,48) dok je vrednovanje računalnog razmišljanja alatom vrednovanja iz ovog istraživanja postiglo zadovoljavajuću razinu pouzdanosti (α=0,63). Razlika u pouzdanosti dvaju instrumenata još je veća na odabranom uzorku učenika koji su sudjelovali u oba vrednovanja (N=49) (Tablica 35).

Tablica 35: Osnovna statistička obilježja natjecanja Dabar i razvijenog CT vrednovanja za odabrane uzorke sudionika

	Natjecanje Dabar	CT vrednovanje	Rezultati učenika koji su sudjelovali u oba vrednovanja	
			Dabar	CT
Veličina uzorka	N=1892	N=358	N=49	N=49
Pouzdanost (Cronbach alpha)	α=0,48	α=0,63	α=0,325	α=0,722
			Korelacija (Spearman's ρ) 0,495 (p=0.01)	

Analizom prikazanih podataka i rezultata može se reći da je razvijeni alat vrednovanja računalnog razmišljanja ovog istraživanja pokazao bolju usklađenost s nacionalnim

dokumentima te bolje statističke karakteristike (koeficijent pouzdanosti, međusobno povezivanje čestica alata vrednovanja, indeks diskriminativnosti) od vrednovanja zadacima natjecanja Dabar. Ipak, za pouzdaniji zaključak vezano uz zadatke natjecanja Dabar trebalo bi provesti istraživanje na većem i reprezentativnijem uzorku učenika.

5.6. Formativno vrednovanje mentalnih modela vezanih uz koncepte apstrakcije, dekompozicije te algoritamskog razmišljanja vrednovanjem računalnog razmišljanja

5.6.1. Opis istraživanja

Primjena razvijenog modela vrednovanja do sada je predstavljena s naglaskom na sumativni oblik vrednovanja, odnosno vrednovanje naučenog kod učenika. Nakon takve primjene alata vrednovanja učenik i učitelj dobit će povratnu informaciju kojom će se iskazati uspjeh učenika pri vrednovanju računalnog razmišljanja u obliku broja postignutih bodova, odnosno odgovarajućom ocjenom. Takav pristup vrednovanju svakako je poželjan na kraju određene nastavne cjeline, odnosno nekog obrazovnog ciklusa.

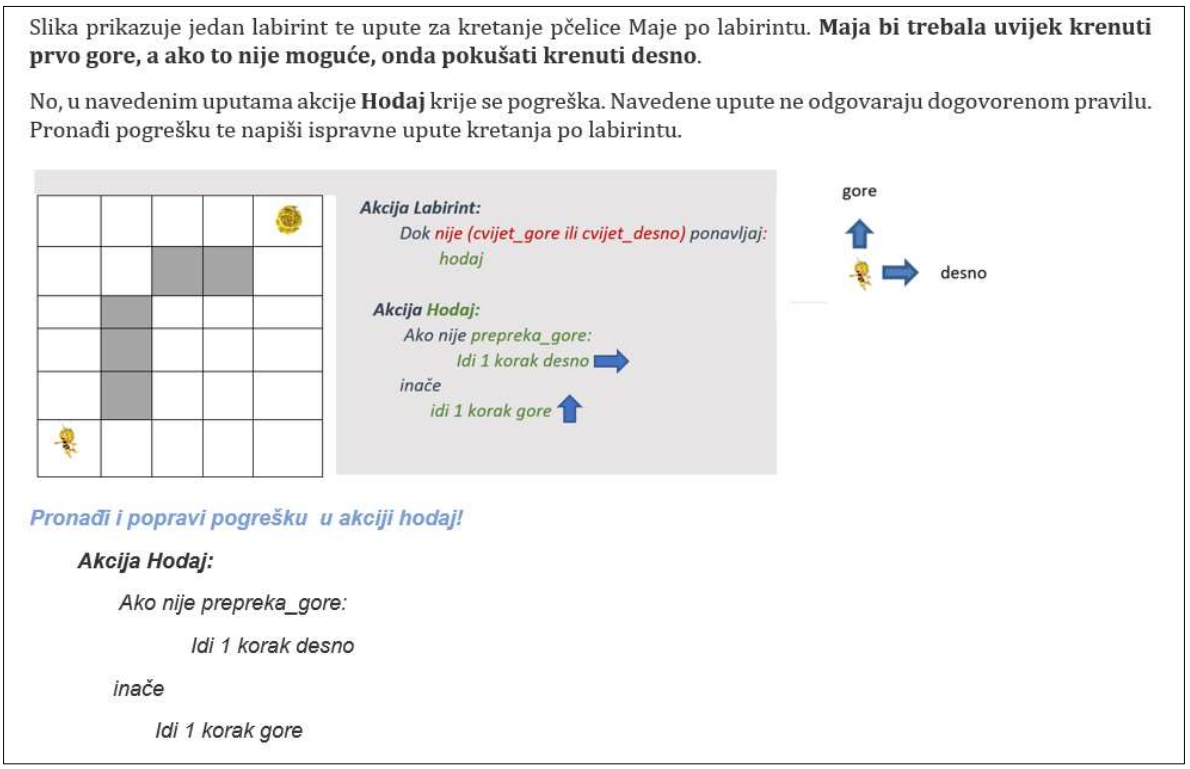
Model vrednovanja koji analizira dokaze znanja promatranih koncepata kod učenika, može ponuditi i detaljniju informaciju o samom napretku učenika tijekom usvajanja nastavnog sadržaja. Primjena alata vrednovanja za analizu usvojenih mentalnih modela promatranih koncepata kod učenika može poslužiti kao alat formativnog vrednovanja kojim se prepoznaju valjani mentalni modeli, ali i ističu poteškoće na koje učenici nailaze pri usvajanju nastavnog gradiva. Takva informacija može biti od ključne važnosti za planiranje daljnjih nastavnih aktivnosti te eventualno korigiranje kurikularnog kruga.

U sklopu okvira vrednovanja definirani su dokazi znanja te model vrednovanja svakog dokaza za zadatke otvorenog tipa (Poglavlje 4.3.3). Svaki dokaz znanja opisan je odgovarajućim mentalnim modelom te uočenim problemom (ako dokaz predstavlja pogrešno ili djelomično rješenje zadatka). Nakon analize prepoznatih mentalnih modela prikazan je jedan način formativnog vrednovanja praksi i koncepata računalnog razmišljanja. Formativno vrednovanje prikazano je i u obliku holističke rubrike za one računalne prakse i vještine odabranih koncepata za koje je, na osnovu prikupljenih podataka, bilo moguće promatrati tri razine usvojenosti.

5.6.2. Analiza podataka

Temelj za izradu formativnog vrednovanja bili su popisi praksi i vještina računalnog razmišljanja kojima se vrednuju koncepti apstrakcije, algoritamskog razmišljanja i dekompozicije (Slika 26 Poglavlje 5.2.1.) te dokazi znanja koje smo definirali razvijenim okvirom vrednovanja (Poglavlje 4.3.3). Takav pristup prikazan je na primjeru odabranih zadataka te zadatcima otvorenog tipa u alatu vrednovanja (zadatak 3, zadatak 7, zadatak 9) u kojima se od učenika očekuju analiza i nadogradnja algoritamskog rješenja te samostalno kreiranje dijela algoritamskog rješenja za zadatke u kojima se vrednuje koncept dekompozicije

ili prepoznaje neka miskoncepcija. Analizom učeničkih odgovora prepoznaju se njihovi usvojeni modeli promatranih koncepata. Tablica 36 prikazuje analizu mentalnih modela i uočenih problema kod rješenja za Zadatak 3 (Slika 31).



Slika 31: Prikaz zadatka vrednovanja - Zadatak 3

U prikazanom se zadatku od učenika tražilo da prouči zadani problem, analizira zadano algoritamsko rješenje te ga korigira u skladu sa zahtjevima problema. Uz tekst zadatka učeniku je ponuđeno pogrešno algoritamsko rješenje koje je potrebno analizirati te ispraviti na odgovarajući način.

Čak 38.07% učenika nije uopće pokušalo riješiti zadatak (ostavili su ponuđeno rješenje netaknuto, izbrisali su ponuđeno rješenje) ili su dopisali neke naredbe na kraju rješenja (npr. idi jedan korak naprijed) koje ukazuju da učenik nije uopće razumio što se postiže ponuđenim algoritamskim rješenjem. Takav pristup ukazuje na potpuno nerazumijevanje zadanog problema, ali i nesposobnost analize i praćenja koda. Manji broj učenika prepoznao je pogrešku, ali je nije znao ispraviti (3.13%). Takvi učenici očigledno imaju problem s nerazumijevanjem zadanog problema, ali i radom s algoritamskom strukturom grananja.

Učenici su uglavnom pokušali izmijeniti jednu od naredbi u grani DA ili u grani NE naredbe grananja ili su nešto izmijenili u uvjetu naredbe grananja, no ta izmjena nije imala nikakve poveznice sa zadanim problemom. Neki učenici takav bi postupak opisali komentarom '*Znam*

da nešto trebam izmijeniti u uvjetu ili naredbama u granama DA/NE, ali nemam pojma što.'. Dakle oni prepoznaju mjesto u ponuđenom rješenju koje bi trebali izmijeniti, ali još uvijek nemaju dovoljno razumijevanje problema niti vještina algoritamskog razmišljanja da bi to primijenili. Nadalje, 5.68% učenika djelomično je prepoznalo pogrešku i pokušalo je izmijeniti.

Tablica 36: Mentalni modeli prepoznati analizom učeničkih rješenja - Zadatak 3

Mentalni model	Uočeni problem	Broj učenika	
Učenik ne zna što treba napraviti ili uopće se ne uočava pogrešku.	Nerazumijevanje problema Algoritamsko razmišljanje (nesposobnost analize i praćenja koda)	134	38,07%
Učenik je prepoznao pogrešku, ali je ne zna ispraviti.	Algoritamsko razmišljanje (rad s logikom, struktura grananja) Nerazumijevanje problema	11	3,13%
Učenik prepoznaje pogrešku i djelomično je uspješno ispravlja.	Algoritamsko razmišljanje (praćenje i analiza koda) Djelomično nerazumijevanje problema	20	5,68%
Učenik uspješno koristi logiku i brzo korigira pogrešku.	-	129	36,65%
Učenik ne mijenja logički uvjet već uspješno ispravlja kretanja u obje grane naredbe if.	Rad s logičkim vrijednostima	57	16,19%

U ovoj analizi posebno su zanimljivi učenici koji su ispravno riješili zadatak. S obzirom da se u zadatku nije navelo na koji način treba riješiti zadatak, svako ispravno rješenje bodovano je s maksimalnim brojem bodova (2). Uspješno rješenje ponudilo je 52,84% učenika. Moglo bi se zaključiti da su svi ti učenici usvojili sve što se od njih očekivalo i da ne postoje problemi u njihovom razumijevanju koji bi se mogli korigirati. No, ako se pri vrednovanju ne stavlja naglasak samo na broj bodova kao mjerilo znanja već i na način rješavanja zadatka, mogu se dobiti kvalitetnije informacije o mentalnim modelima koje su usvojili učenici te otkriti moguće probleme koji učenicima mogu stvarati poteškoće u nekim drugim, više zahtjevnim zadacima ili pri usvajanju novog nastavnog gradiva.

Najbolje rješenje koje se od učenika očekivalo bilo je napraviti jednu izmjenu ponuđenog rješenja i to izmjenom logičkog uvjeta (*ako nije prepreka_gore → ako je prepreka_gore*). Takvo rješenje ponudilo je 36,65% učenika i za njih možemo reći da se uspješno koriste logikom (rad s logičkim vrijednostima, apstrakcija) te da imaju takve vještine algoritamskog

razmišljanja (praćenje koda, rad s algoritamskom strukturom grananja) koje im omogućuju brzo i ispravno korigiranje rješenja.

Za razliku od njih, 16,19% učenika koji su također ponudili ispravno rješenje problema, nije izvršilo izmjenu uvjeta naredbe grananja, već su na odgovarajući način korigirali naredbe u granama DA/NE naredbe grananja. Takav pristup rješavanju problema može ukazati na problem u radu s logičkim vrijednostima. Također, takav pristup zahtijeva dvije izmjene postojećeg rješenja umjesto jedne. Među učenicima koji su djelomično točno riješili zadatak nalazi se veći broj onih koji su krenuli ovim pristupom (izmjena naredbi u granama DA/NE), ispravno su izmijenili prvu naredbu (grana DA), ali su vjerojatno zaboravili napraviti odgovarajuću izmjenu i druge naredbe (grana NE).

Među ovim učenicima treba spomenuti i dvoje učenika koji su prilično izmijenili ponuđeno računalno rješenje na način da su jednu složenu naredbu grananja (*ako .. tada.. inače*) zamijenili s dvije jednostavne naredbe grananja (*ako…tada…*). Iako su njihove izmjene ponudile točno rješenje problema, one ukazuju na problem u radu sa složenom algoritamskom strukturom grananja te logičkim vrijednostima.

Ovaj zadatak pokazao je umjeren indeks težine (44.03), ali i veliku snagu razlikovanja učenika koji su jako dobro usvojili promatrane koncepte od onih koji pokazuju ozbiljne poteškoće kako u razumijevanju problema tako i s vještinama algoritamskog razmišljanja (indeks diskriminativnosti 0.38).

Analiza usvojenih koncepata apstrakcije kao što su rad s različitim razinama apstrakcije (razumijevanje problema) te rad s varijablama može se promatrati kroz zadatke vrednovanja koji koriste brojače, varijable kojima se prati promjena zadane vrijednosti u algoritamskom rješenju.

Sljedeći zadatak primjer je takvog vrednovanja (Slika 32). Zadatak je tražio od učenika da analizira problem te prati kretanje glavnog lika izmjenom zadane vrijednosti **broj_koraka**. Nakon analize problema i ponuđenog algoritmskog rješenja učenik mora prepoznati mjesto u rješenju koje bi trebalo izmijeniti/nadograditi (sposobnost razumijevanje problema).

Pčelica Maja odlučila je pratiti broj koraka koje napravi na svom putu prema labirintu te bilježiti ih u oznaci **broj_koraka** (na slici iznad labirinta). Na početku svakog kretanja kroz labirint vrijednost **broj_koraka** Maja uvijek postavlja na 0

Nadopuni Majine upute kretanja kroz labirint tako da vrijednost **broj_koraka** prikazuje uvijek ukupan broj napravljenih koraka. Dakle, na odgovarajuće mjesto u akciji **Hodaj** potrebno je napisati tvoju verziju upute kojom se vrijednost **broj_koraka** poveća za 1 svaki put kad se napravi jedan korak.

```
Akcija Labirint:
    Postavi broj_koraka na 0
    Dok nije cvijet_gore ponavljaj:
        hodaj
Akcija Hodaj:
    Dok nije prepreka_gore ponavljaj:
        Idi 1 korak gore
    Ako nije prepreka _desno:
        Idi 1 korak desno
    inače:
        Ako nije prepreka_lijevo:
            Idi 1 korak lijevo
```

Slika 32: Prikaz zadatka vrednovanja - Zadatak 7

Nadalje, učenik prepoznaje varijablu kao objekt kojim se može pratiti izmjena promatrane vrijednosti (kretanje glavnog lika po labirintu) te odgovarajućim vještinama algoritamskog

razmišljanja (praćenje i analiza koda, naredba pridruživanja) uvodi varijablu u algoritamsko rješenje.

Tablica (Tablica 37) prikazuje načine rješavanja koje su učenici demonstrirali. Zadatak je očigledno učenicima bio jako težak jer čak 80.7% učenika nije napravilo nikakvu izmjenu u zadatku ili je samo napisalo par naredbi/izraza u kojima se prikazuje neko povećavanje vrijednosti, ali pri tome nisu koristili varijablu.

Manji broj učenika upotrijebio je naredbu kojom se povećava brojač za jedan, no naredbu nisu smjestili na odgovarajuće mjesto nego npr. u akciju Labirint. Takav postupak ukazuje da učenik prepoznaje potrebu za uporabom varijable za praćenje kretanja glavnog lika (rad s varijablama), ali još uvijek nema dovoljno razvijene vještine algoritamskog razmišljanja (praćenje i analize koda) kako bi uspješno napravio nadogradnju ponuđenog rješenja.

Ovaj se zadatak pokazao jako teškim za učenike (indeks težine 89.74), no kao i u prethodno razmatranom zadatku može se reći da jasno razlikuje učenike koji su jako dobro usvojili promatrane koncepte od onih koji pokazuju ozbiljne poteškoće kako u razumijevanju problema tako i s vještinama algoritamskog razmišljanja (indeks diskriminativnosti 0.41).

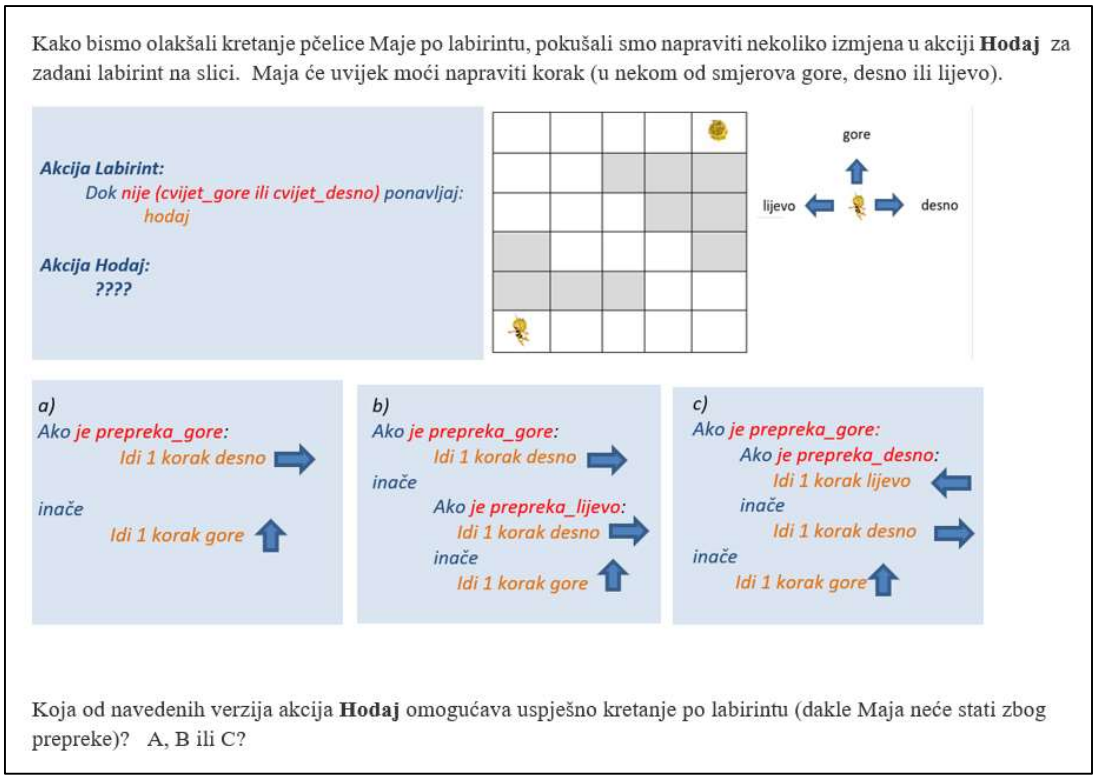
Tablica 37: Mentalni modeli prepoznati analizom učeničkih rješenja - Zadatak 7

Mentalni model	Uočeni problem	Broj učenika	
Učenik ne zna ni gdje ni kako napraviti nadogradnju programa.	Nerazumijevanje problema. Algoritamsko razmišljanje.	284	80.7%
Učenik ne prepoznaje varijable kao vrijednosti kojima se može pratiti kretanje Maje kroz labirint, Razumije da treba napraviti neku izmjenu uz samo kretanje.	Rad s varijablama (apstrakcija). Djelomično nerazumijevanje problema.		
Učenik prepoznaje potrebu za uporabom varijable za praćenje kretanja Maje po labirintu, no ne prepoznaje odgovarajuće mjesto u programu za takvu promjenu.	Algoritamsko razmišljanje (problem s praćenjem koda). Djelomično nerazumijevanje problema.	40	11.4%
Učenik prepoznaje potrebu za uporabom varijable te je uspješno primjenjuje na nekim od očekivanih mjesta.	Djelomično nerazumijevanje problema. Algoritamsko razmišljanje (praćenje koda).	11	3.1%
Učenik uspješno primjenjuje varijable u zadanom problemu.	-	17	4.8%

Koncept algoritamske strukture grananja te rad s logičkim vrijednostima od velike su važnosti pri razmatranju usvojenosti koncepata algoritamskog razmišljanja i apstrakcije. Sljedeća slika

prikazuje zadatak u kojem se očekuje analiza i razumijevanje algoritamskog rješenja te rad s logičkim vrijednostima i naredbom grananja (Slika 33). Ovo je primjer zadatka umjerene težine (46.68) i dobre sposobnosti razdvajanja učenika prema njihovoj uspješnosti rješavanja zadatka (0.27).

Za rješavanje zadatka učenik ima na raspolaganju tri moguće nadogradnje postojećeg algoritamskog rješenja. Poznavanjem rada algoritamske strukture grananja (jednostavne, složene i ugniježdene) te rada s logičkim vrijednostima učenik će doći do ispravnog rješenja odabirom jednog od ponuđenih nadogradnji.



Slika 33: Prikaz zadatka vrednovanja - Zadatak 5

Analizom odgovora učenika prepoznati su mentalni modeli i prikazani sljedećom tablicom (Tablica 38). S obzirom da se radi o umjereno teškom zadatku, čak su 54.8% učenika prepoznala točnu nadogradnju algoritamskog rješenja. Kod 45.2% učenika pojavile su se poteškoće u razumijevanju problema ili rada s logičkim vrijednostima.

Za učenike koji su odabrali prvi prijedlog nadogradnje rješenja (odgovor A, 9.38%) može se reći da imaju samo djelomično razumijevanje problema s obzirom da nisu prepoznali da njihovo rješenje ne omogućava kretanje glavnog lika u lijevo, a to je jedan od zahtjeva zadatka. Do

takve je pogreške moglo doći i zbog problema u radu s logičkim vrijednostima i složenijim strukturama grananja pa učenik nije mogao pravilno pratiti i analizirati izvršavanje predložene nadogradnje.

Tablica 38: Mentalni modeli prepoznati analizom učeničkih rješenja - Zadatak 5

Prijedlog nadogradnje rješenja	Mentalni model	Uočeni problem	Broj učenika	
	Odabrana izmjena ne omogućava cjelovito rješavanje problema (kretanje, u smjeru lijevo, desno ili gore). Razumijevanje samo osnovne strukture grananja.	Djelomično razumijevanje problema. Problem u radu sa složenijim strukturama grananja i logičkim vrijednostima.	33	9.38%
	Odabrana izmjena ne provjerava je li moguće kretanje u smjeru desno. Postoji razumijevanje složene strukture grananja, još nedovoljno razumijevanje rada s logičkim vrijednostima.	Djelomično razumijevanje problema. Problem u radu s logičkim vrijednostima i/ili problem s vještinama praćenja koda.	118	33.52%
	Učenik pokazao razumijevanje algoritamske strukture grananja te rada s logičkim vrijednostima.	-	193	54,8%
		Nisu upisali svoj odgovor	8	2.27%

Nešto bolje razumijevanje problema pokazali su učenici koji su odabrali drugi prijedlog nadogradnje (odgovor B, 33.52%). Iako su ovi učenici prepoznali problem koji se krije u prvoj predloženoj nadogradnji (odgovora A), još uvijek ne možemo reći da u potpunosti razumiju što

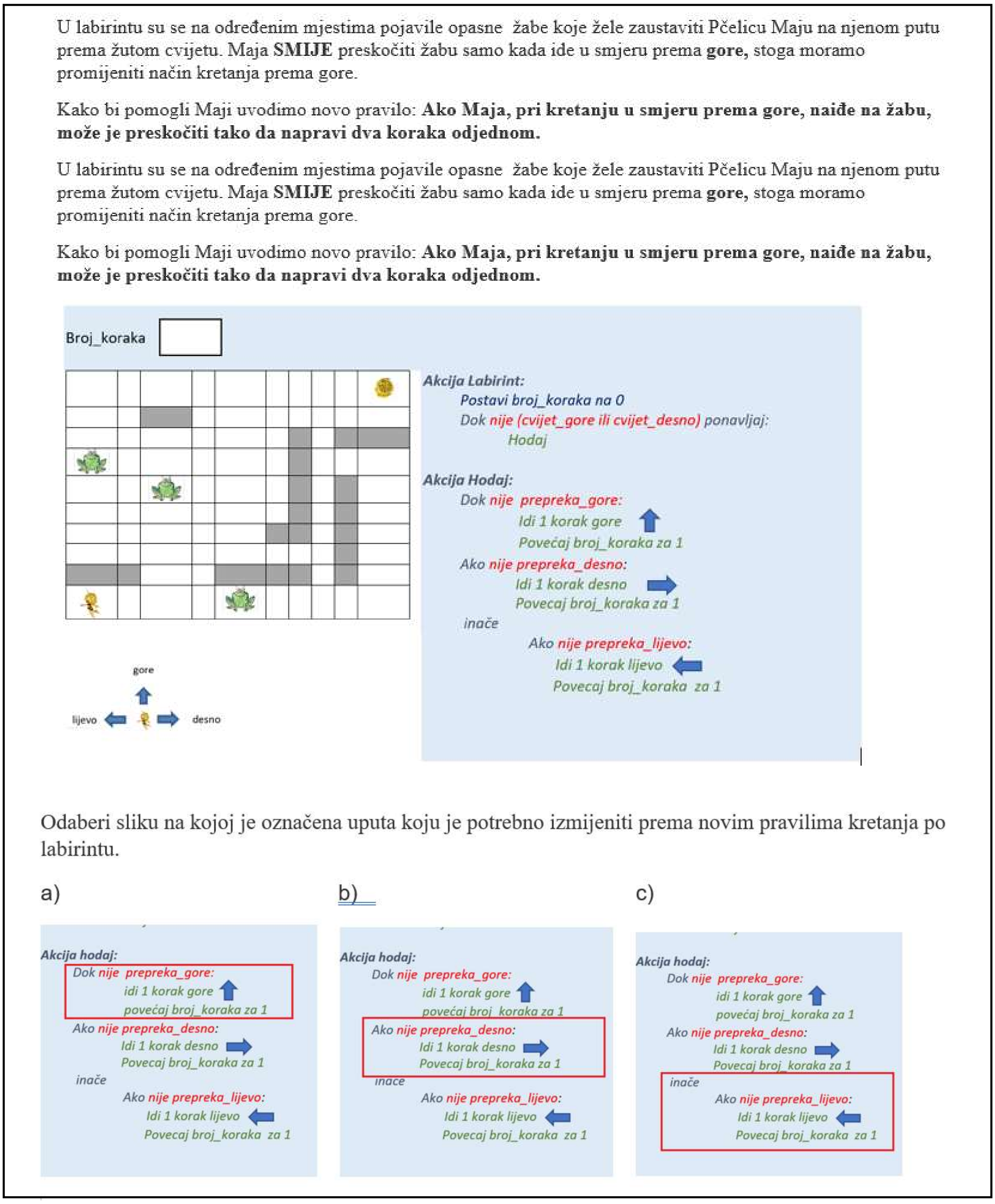
se u zadatku traži i na koji to način trebaju postići. Ovoj skupini učenika najveći problem predstavljala je analiza logičkih vrijednosti koje se koriste u složenim oblicima naredbe grananja.

Zanimljivo je analizirati kako su učenici reagirali na zadatak koji je imao naglasak na konceptu dekompozicije, rastavljanju složenih problema na manje složene (Slika 34). Kao što je već istaknuto u prethodnim poglavljima, koncept dekompozicije nije se dovoljno istaknuo kao zaseban faktor faktorske analizom rezultata. Razlog tome zasigurno je činjenica da je koncept dekompozicije bio predstavljen samo s dvije čestice u alatu vrednovanja što je nedovoljno za faktorsku analizu. Ipak, moguće je razmatrati neka obilježja koncepta dekompozicije. Prema statističkim obilježjima, ovo nije bio jako težak zadatak za učenike (zadatak 8, indeks težine 51.19), a istovremeno je zadatak napravio jako dobru diskriminativnost (0.36). Od učenika se očekivala primjena vještina praćenja i analize ponuđenog rješenja kako bi u rješenju prepoznali poznate uzorke. U ovom zadatku to je bio skup naredbi kojima se opisuje i prati kretanje po labirintu (*idi jedan korak gore; povećaj broj_koraka za 1).*

Dobro razumijevanje problema pomaže učeniku prepoznavanje sličnosti, ali i razlika među uzorcima kojima se opisuje kretanje po labirintu u ponuđenim rješenjima. Čak 49,15% učenika uspješno je prepoznalo točno mjesto, odnosno odgovarajući uzorak kretanja u ponuđenom algoritamskom rješenju koje je potrebno izmijeniti kako bi se nadogradio algoritam prema navedenim zahtjevima (rješenje A). Ta je skupina učenika uspjela prepoznati manji, jednostavniji dio početnog problema, tj. napraviti dekompoziciju problema. S obzirom da je ponuđeno algoritamsko rješenje u ovom zadatku puno složenije od rješenja koja su se nudila u početnim zadacima, možemo reći da je neočekivano velik broj učenika, u odnosu na rezultate prethodnih zadataka, uspješno riješio zadatak. Moguće je da se takav rezultat može objasniti činjenicom da su se učenici već dovoljno dobro upoznali s kontekstom zadatka koji se provlači kroz cijeli alat vrednovanja te ponuđenim algoritamskim rješenjima koja su uglavnom nadogradnja rješenja iz prethodnih zadataka. To govori u prilog činjenici da zadržavanje konteksta kroz cijeli alat vrednovanja može olakšati proces rješavanja na način da se učenici ne zamaraju previše kreiranjem problemskog prostora i više se bave trenutnim problemom u zadatku.

Među onima koji nisu uspješno riješili zadatak, 29,26% učenika odabralo je drugo rješenje (B), 15.65% treće rješenje (C), dok 5.97% učenika nije ponudilo svoj odgovor. Zajedničko im je da

nisu uspješno analizirali sličnosti i razlike među naredbama koje opisuju kretanja u rješenjima, a moguće je da se nisu dobro snašli u predviđanju ponašanja samog algoritma.


U labirintu su se na određenim mjestima pojavile opasne žabe koje žele zaustaviti Pčelicu Maju na njenom putu prema žutom cvijetu. Maja **SMIJE** preskočiti žabu samo kada ide u smjeru prema **gore,** stoga moramo promijeniti način kretanja prema gore.

Kako bi pomogli Maji uvodimo novo pravilo: **Ako Maja, pri kretanju u smjeru prema gore, naiđe na žabu, može je preskočiti tako da napravi dva koraka odjednom.**

U labirintu su se na određenim mjestima pojavile opasne žabe koje žele zaustaviti Pčelicu Maju na njenom putu prema žutom cvijetu. Maja **SMIJE** preskočiti žabu samo kada ide u smjeru prema **gore,** stoga moramo promijeniti način kretanja prema gore.

Kako bi pomogli Maji uvodimo novo pravilo: **Ako Maja, pri kretanju u smjeru prema gore, naiđe na žabu, može je preskočiti tako da napravi dva koraka odjednom.**

Odaberi sliku na kojoj je označena uputa koju je potrebno izmijeniti prema novim pravilima kretanja po labirintu.

a) b) c)

Slika 34: Prikaz zadatka vrednovanja - Zadatak 8

Nakon provedenog vrednovanja, analizom prikupljenih podataka i prepoznatih mentalnih modela grupiraju se prepoznate poteškoće, odnosno usvojenost promatranih koncepata kroz tri različite razine usvojenosti. Takva analiza može se prikazati u obliku formativne rubrike za vrednovanje tzv. holističke rubrike (Tablice 39-40). Predložene rubrike mogu se koristiti kao kvalitetna povratna informacija za učenike kako bi razumjeli koje poteškoće postoje u njihovom razumijevanju nastavnog gradiva.

Rubrike mogu biti vodilja učitelju kako bi pripremio dodatni materijal te prilagodio svoj nastavni rad u skladu s prikupljenim rezultatima formativnog vrednovanja. Za koncept apstrakcija analizirani su podatci za računalnu praksu i vještine: razine apstrakcije (razumijevanje problema), rad s varijablama te rad s logičkim vrijednostima (Tablica 39).

Tablica 39: Prijedlog rubrike za formativno vrednovanje na temelju prepoznatih mentalnih modela kod učenika za koncepte povezane s apstrakcijom

		Treba doraditi	Djelomično uspješno	Uspješno
Apstrakcija	Razine apstrakcije (razumijevanje problema)	Učenik nema saznanja /ideju gdje bi trebao izmijeniti/nadograditi program, odnosno ne uočava pogrešku od koje treba započeti izmjenu/nadogradnju.	Učenik prepoznaje potrebu za izmjenom programa na odgovarajućem mjestu i djelomično ga uspješno popravlja.	Učenik uočava pogrešku te nadograđuje program na odgovarajući način.
	Rad s varijablama	Učenik ne prepoznaje varijable kao vrijednosti kojima se mogu pratiti izmjene u programu. U programu se ne koristi varijablama ili ih koristi kao konstantne vrijednosti (njihove se vrijednosti nikada ne mijenjaju iako im je namjena da prate neke promjene (događaj).	Učenik prepoznaje potrebu za primjenom varijabli, ali ih koristi na pogrešnim mjestima ili na pogrešan način.	Učenik koristi varijable za praćenje izmjena vrijednosti koje su važne za rješavanje problema.
	Logičke vrijednosti	Učenik koristi logički izraz koji se ne odnosi na zadani problem. Učenik umjesto logičkog izraza koristi konstantne vrijednosti.	Učenik koristi logički uvjet koji se odnosi na zadani problem no uvjet sadrži sintaktičku ili logičku pogrešku npr. izrazom .A%2 umjesto A%2==0 provjera parnost, izrazom a<0	Učenik koristi logički uvjet koji u potpunosti adresira zadani problem. Učenik razlikuje aktivnosti koje je potrebno izvršiti u ovisnosti o vrijednosti uvjeta

			provjerava je li broj pozitivan … - Učenik povezuje samo istinitu vrijednost logičkog uvjeta s nekom aktivnosti koju je potrebno izvršiti (primjenjuje dvije naredbe jednostavnog grananja (if...) umjesto jedne naredbe složenog grananja (if...else…).	(istina/Laž). Uspješno povezuje vrijednost logičkog uvjeta s odgovarajućim aktivnostima primjenom odgovarajuće naredbe grananja.

Za koncept algoritamskog razmišljanje analizirani su podatci za računalnu praksu i vještinu: praćenje i analiza koda/algoritma, primjena strukture grananja. Rubrika za formativno praćenje prikazana je sljedećom tablicom (Tablica 40).

Tablica 40: Prijedlog rubrike za formativno vrednovanje na temelju prepoznatih mentalnih modela kod učenika za koncepte povezane s algoritamskim razmišljanjem.

		Treba doraditi	Djelomično uspješno	Uspješno
Algoritamsko razmišljanje	Praćenje i analiza koda/algoritma	Učenik ne uočava pogrešku ili zadanu aktivnost u programu npr. Učenik ne nudi svoje rješenje, ne radi izmjene u programu ili radi izmjene na potpuno pogrešnom mjestu u programu.	Učenik uočava pogrešku u programu, no nudi izmjenu koja ne adresira u potpunosti zadani problem, npr. vrši se više ili manje izmjena u programu nego što je to potrebno.	Učenik uspješno prepoznaje pogrešku/aktivnosti u programu te ga prema zahtjevima uspješno nadograđuje.
	Primjena strukture grananja	Učenik primjenjuje nepotpunu strukturu grananja npr. nedostaje uvjet ili nedostaje pojedina naredba koja se izvršava u ovisnosti o logičkom uvjetu.	Učenik primjenjuje strukturu grananja, no odabire krive aktivnosti koje se izvršavaju u ovisnosti o logičkom uvjetu. Učenik primjenjuje uvjetnu petlju umjesto strukture grananja.	Učenik primjenjuje odgovarajuću strukturu grananja (jednostavnu ili složenu) u ovisnosti o zadanom problemu. Odabire odgovarajuće aktivnosti koje se izvršavaju u ovisnosti o logičkom uvjetu.

Prikazani rezultati govore u prilog sposobnosti razvijenog alata vrednovanja temeljenog na dokazima da prepozna mentalne modele odabranih koncepata apstrakcije i algoritamskog razmišljanja te djelomično i koncepta dekompozicije.

Ovakav pristup vrednovanju svakako je dugotrajan i zahtjevan za učitelje. Ipak, omogućava stvaranje kvalitetnog okvira vrednovanja koji razmatra model učenika, zadatka, dokaza te mjerenja. Nadalje, omogućava kvalitetnu povratnu informaciju učiteljima i učenicima o razini usvojenosti promatranih koncepata, ali i o poteškoćama koje učenicima otežavaju napredak u učenju. Stvaranje takvih modela može poslužiti kao predložak ostalim učiteljima u kreiranju jednako kvalitetnih alata vrednovanja različitog konteksta.

5.7. Utjecaj ostalih faktora (spol, opći akademski uspjeh, uspjeh iz matematike, škola/grad) na uspjeh pri vrednovanju računalnog razmišljanja

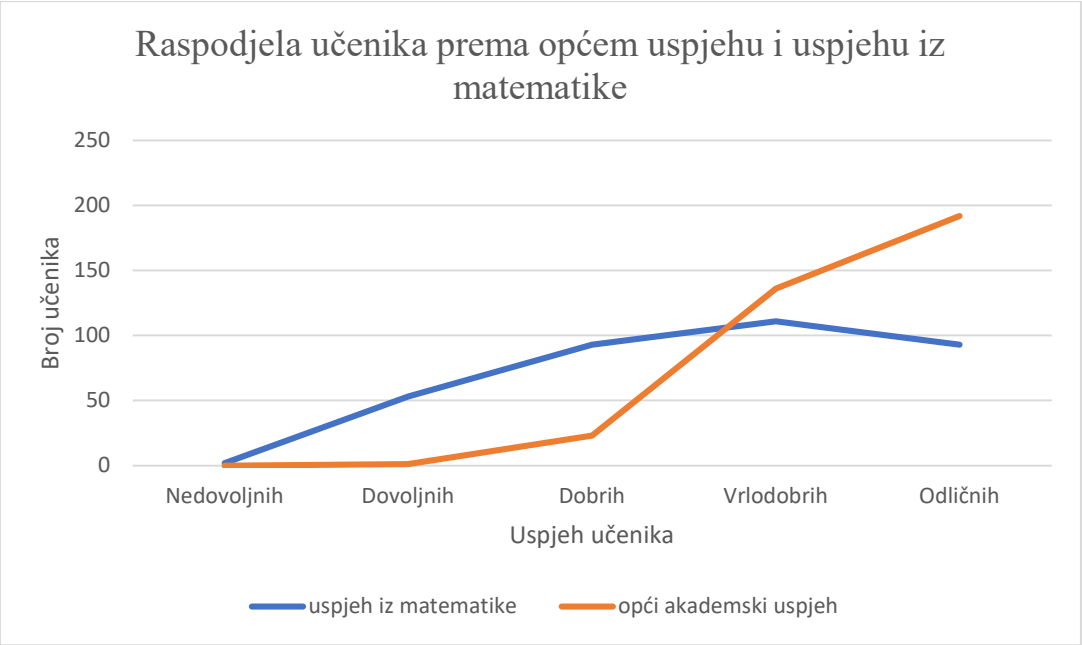
5.7.1. Opis istraživanja

Ranija istraživanja već su pokazala da postoje faktori koji su značajno povezani s uspjehom u vještinama programiranja i algoritamskog razmišljanja (Bubica i Boljat, 2014). U ovom istraživanju istražio se utjecaj faktora spola, uspjeha iz matematike, općeg akademskog uspjeha te škole učenika na uspjeh pri vrednovanju računalnog razmišljanja. Podatci učenika koji su sudjelovali u istraživanju vezano uz njihov opći uspjeh te uspjeh iz matematike ne ispunjavaju kriterij normalnosti pa se u daljnjim analizama s tim podatcima koriste neparametrijske tehnike. Tablica (Tablica 41) prikazuje srednju vrijednost oba uspjeha te raspodjelu ocjena za svaku promatranu varijablu. Zanimljiv je podatak da je u istraživanju sudjelovalo 57,9% vrlo dobrih ili odličnih učenika iz matematike. Čak je 54,5% učenika završilo prethodni razred (5. razred) s odličnim uspjehom, a 38,6% učenika s vrlo dobrim uspjehom, odnosno čak 93,1% učenika završilo je prethodni razred s vrlo dobrim ili odličnim uspjehom.

Tablica 41: Raspodjela učenika s obzirom na njihov opći uspjeh te uspjeh iz matematike

	Srednja vrijednosti	Nedovoljnih	Dovoljnih	Dobrih	Vrlo dobrih	Odličnih
Uspjeh iz matematike (zaključna ocjena)	3,6818	2 (0.6%)	53 (15,1%)	93 (25,4%)	111 (31,5%)	93 (26,4%)
Opći akademski uspjeh (srednja ocjena)	4,4084	0 (0%)	1 (0.3%)	23 (6,5%)	136 (38,6%)	192 (54,5%)

Grafički prikaz raspodjele podataka obje varijable (uspjeh iz matematike, opći uspjeh) prikazuje razliku u trendu (Slika 35). Iako za varijablu uspjeh iz matematike imamo manji broj odličnih učenika nego vrlo dobrih učenika, možemo reći da krivulja značajno odstupa od krivulje normalno raspodijeljenih podataka. Kod varijable općeg uspjeha krivulja je značajno uzlaznog trenda, što također vidljivo odstupa od krivulje normalno raspodijeljenih podataka.



Slika 35: Raspodjela učenika prema općem uspjehu i uspjehu iz matematike

U istraživanju je sudjelovalo 352 učenika iz deset različitih škola raspoređenih u osam gradova i sedam županija (Tablica 42). Svi učenici iz iste škole imali su istog učitelja. Svaku školu određuje i programski jezik/okružje koje je učitelj odabrao za podučavanje sadržaja iz programiranja, kao i duljina podučavanja navedenih sadržaja (broj sati) te iskustvo učitelja (godine staža).

Tablica 42: Raspodjela učenika po školama

Škola (1-10)	Županija (1-6)	Programski jezik	Duljina poučavanja programiranja (broj_sati)	Iskustvo učitelja (godine staža)
Škola A	Koprivničko-križevačka	-	*	*
Škola B	Splitsko-dalmatinska	Python	12/70	13
Škola C	Bjelovarsko - bilogorska	Python	26/70	11
Škola D	Koprivničko-križevačka	-	*	*
Škola E	Dubrovačko-neretvanska	Python	24/70	8
Škola F	Grad Zagreb	Logo	14/70	20
Škola G	Dubrovačko - neretvanska	Python	24/70	24
Škola H	Zadarska	Python	25/70	25
Škola I	Grad Zagreb	Logo	24/70	23
Škola J	Vukovarsko-srijemska	Python	14/70	19

* Škole A,D nisu dostavile potrebne podatke

Odgovarajućom statističkom metodom istražilo se jesu li navedene varijable utjecale na uspjeh učenika pojedine škole. U ovom dijelu istraživanja ispitala se i valjanost hipoteze da postoji korelacija između uspjeha kojeg postižu učenici šestih razreda osnovnih škola pri vrednovanju koncepata računalnog razmišljanja i uspjeha iz matematike.

5.7.2. Analiza podataka

Podatci učenika koji se odnose na njihova opća akademska i matematička postignuća pokazali su da ovaj uzorak ne zadovoljava uvjet normalne raspodjele pa je za istraživanje povezanosti podataka primijenjen neparametrijski test (jedan uzorak Kolmogorov – Smirnov test p = .001). Istraživanjem povezanosti varijabli općeg uspjeha i uspjeha iz matematike može se reći da će učenici s visokim općim akademskim postignućima vjerojatno imati visoka matematička postignuća (Spearmanov ρ= 0,803, p≤0,001). Takvi rezultati u skladu su s relevantnom literaturom (Bubica i Boljat, 2014; Wilson i Shrock, 2001). Faktori kao što su uspjeh iz matematike te opći akademski uspjeh pokazali su srednje snažnu pozitivnu korelaciju s uspjehom na vrednovanju računalnog razmišljanja iz ovog istraživanja (Spearmanov ρ= 0,425, ρ= 0,429, p≤0,001). Dakle, učenici koji imaju boji uspjeh iz matematike te bolji opći akademski uspjeh, vjerojatno će postići i bolji uspjeh na ovom vrednovanju računalnog razmišljanja. Takav rezultat omogućava prihvaćanje hipoteze istraživanja (H2) koja pretpostavlja postojanje korelacije između uspjeha kojeg postižu učenici na ovom vrednovanju te njihovog uspjeha iz matematike.

U skladu s postojećim istraživanjima (Bubica i Boljat, 2014; Pillay i Jugoo, 2005), nije se pokazala veza između spola učenika i njihovog uspjeha na alatu vrednovanja računalnog razmišljanja (Spearmanov ρ = -0,051, p = 0,337> 0,05), što govori u prilog tvrdnji da bi alat mogao biti podjednako prikladan za dječake ili djevojčice.

U prethodnim poglavljima analiziralo se ovisi li uspjeh učenika na ovom vrednovanju o programskom jeziku kojeg su učenici koristili u procesu podučavanja. Rezultati su pokazali da odabir programskog jezika nema značajan utjecaj na uspjeh učenika (Poglavlje 5.7.). Također, duljina trajanja podučavanja nije pokazala značajnu povezanost s uspjehom učenika što je neočekivan rezultat (Poglavlje 5.4.). Istražilo se postoji li razlika među rezultatima učenika pojedinih škola kako bi se objasnili dobiveni rezultati. S obzirom da podatci ne udovoljavaju uvjetu normalne raspodjele, za istraživanje se koristila neparametrijska metoda. Kruskal Wallis metoda (N neovisnih uzoraka) pokazala je da postoji statistički značajna razlika u rezultatima pojedinih grupa unutar uzorka (Tablica 43-44), odnosno da postoje razlike među rezultatima

učenika s obzirom na školu koju pohađaju. Može se reći da je škola koju učenici pohađaju jedan od faktora koji značajno utječe na rezultate ovog vrednovanja. Analizom srednjih rangova ističu se dvije škole (škola B, škola I) čiji će učenici vjerojatno postići najbolji rezultat na vrednovanju računalnog razmišljanja. Nadalje, slijedi skupina od tri škole (škola F, škola C, škola H) čiji će učenici biti gotovo podjednako uspješni. Uspješnost učenika ostalih škola mogla bi se složiti u padajući niz po prema očekivanom uspjehu učenika na ovom vrednovanju: škola G, škola J, škola A/D, škola E, škola A/D.

Tablica 43: Kruskal Wallis test – srednji rangovi

	Škola	N	Mean Rank
uspjehtest	Škola B	48	226,35
	Škola I	31	202,66
	Škola F	64	187,41
	Škola C	53	185,44
	Škola H	24	183,79
	Škola G	26	168,90
	Škola J	48	154,21
	Škola A/D	19	126,50
	Škola E	36	120,47
	Škola A/D	3	71,00
	Ukupno	352	

Tablica 44: Kruskal Wallis test – statistički podatci

Test Statistics[a,b]

	uspjehtest
Chi-Square	41,054
df	9
Asymp. Sig.	,000

a. Kruskal Wallis Test

b. Grouping Variable: šifra škole

Kako objasniti navedene rezultate? Analizirajmo neka osnovna obilježja škola koja su razmatrana u ovom istraživanju. Svaku školu obilježava učitelj koji radi sa svojim učenicima.

Učitelj u svom Nastavnom planu i programu (NPiP), danas Godišnjem izvedbenom kurikulu (GIK), planira ukupan broj nastavnih sati koje će utrošiti za obrađivanje koncepata programiranja. Svaki učitelj primjenjuje odabrane metode podučavanja u skladu s osobnim preferencijama i tehničkim mogućnostima škole, odnosno informatičke učionice. Škole obilježavaju i neki drugi faktori određeni načinom organizacije nastave s obzirom na specifične uvjete rada škole, npr. rad u više smjena, nastava informatike u školi je u sklopu redovne nastave ili u suprotnoj smjeni i sl. no takvi podatci kao ni neki podatci koji se odnose na učenika (npr. očekivanja roditelja, uvjeti života…) nisu bili dostupni u ovom istraživanju.

S obzirom da su rezultati pokazali da će učenici određenih škola postići bolji rezultat od učenika nekih drugih škola, istražilo se postoje li razlike među školama s obzirom na opći uspjeh učenika, njegov uspjeh iz matematike, duljinu podučavanja sadržaja programiranja (sati učenja) te duljinu staža učitelja.

Provedena je višestruka regresijska analiza kako bi se istražila mogućnost predviđanja uspjeha (broj bodova) na vrednovanju iz ovog istraživanja navedenim varijablama (sati učenja, opći uspjeh, uspjeh iz matematike, staž učitelja). Pokazalo se da varijable sati_učenja i uspjeh_iz_matematike statistički značajno predviđaju uspjeh na vrednovanju računalnog razmišljanja iz ovog istraživanja $F(4,281)=20,743$, $p=0.000>0.001$, $R^2=0.228$) (Tablica 45-47). Treba naglasiti da je u prethodnoj analizi zasebno promatranje utjecaja varijable općeg uspjeha pokazalo statistički značajan pozitivan utjecaj na uspjeh na vrednovanju, no taj utjecaj očigledno slabi kada ga razmatramo zajedno s ostalim varijablama (uspjeh iz matematike, sati učenja).

Tablica 45: Analiza modela regresijske analize

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	,477[a]	,228	,217	3,35875

a. Predictop: (Constant), ucitelj_staz, sati_ucenja, uspjeh, uspjehmat

Tablica 46: Predviđanje uspjeha na vrednovanju – rezultati višestruke regresijske analize

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	936,043	4	234,011	20,743	,000[b]
	Residual	3170,009	281	11,281		
	Total	4106,053	285			

a. Dependent Variable: Ocjena2100

b. Predictop: (Constant), ucitelj_staz, sati_ucenja, uspjeh, uspjehmat

Tablica 47: Koeficijenti regresijske analize

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	,042	1,690		,025	,980
	uspjehmat	1,382	,276	,372	5,006	,000
	uspjeh	,673	,438	,112	1,537	,125
	sati ucenja	-,079	,036	-,118	-2,211	,028
	ucitelj staz	,069	,036	,101	1,913	,057

a. Dependent Variable: Ocjena2100

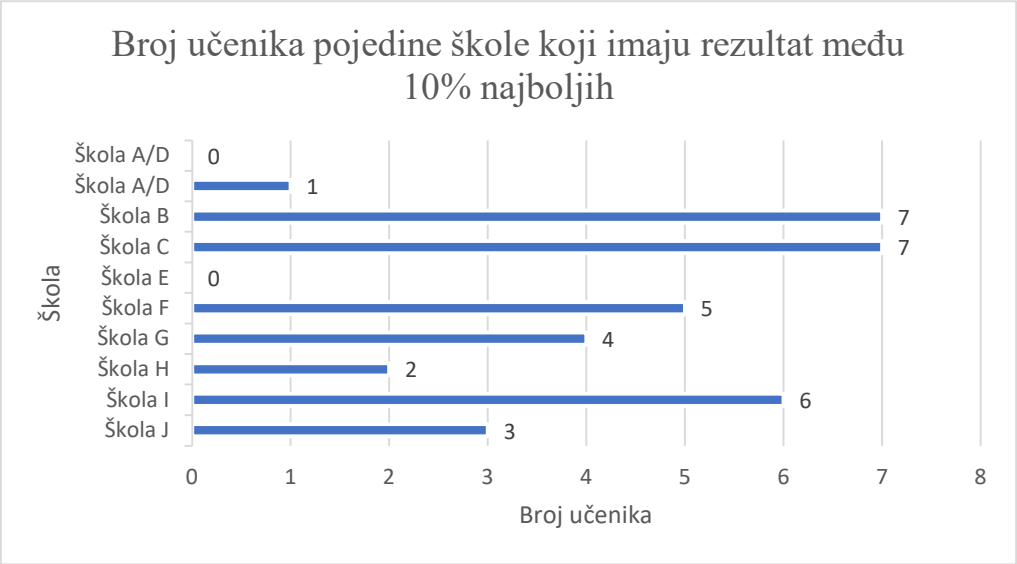
Iz priloženih tablica vidi se da je uspjeh iz matematike najsnažniji prediktor uspjeha na ovom vrednovanju (p=0,000, Beta=0,372). Također, takav rezultat potvrdio je prethodno istraženu povezanost varijabli uspjeha na ovom vrednovanju i uspjeha iz matematike. U poglavlju 5.4. istraženi su podatci učenika i škola koje se odnose na sate učenja sadržaja programiranja kao i na koncepte koji su bili uključeni u proces podučavanja. Istražilo se postoji li veza tih faktora s uspjehom učenika na ovom vrednovanju.

Rezultati su pokazali da se ne može zaključiti kako će duljina trajanja podučavanja sadržaja iz programiranja značajno utjecati na uspjeh učenika na ovom vrednovanju. Ipak, provedena višestruka regresijska analiza istaknula je varijablu sati_učenja kao jedan od značajnijih faktora, ali s negativnim predznakom (p=0.28<0.05; Beta=-0.118). Možemo reći da duljina podučavanja kao samostalan faktor nema snagu prediktora uspjeha na ovom vrednovanju,

ali zajedno s varijablom uspjeh iz matematike može značajno utjecati na rezultat vrednovanja računalnog razmišljanja.

Višestrukom regresijskom analizom istražen je i utjecaj varijable godine_staža učitelja na uspjeh učenika pri vrednovanju. Rezultati su pokazali da je utjecaj iskustva učitelja značajan, ali ipak ne statistički značajan (p=0.057>0.05, Beta=0,101). Uzimajući u obzir činjenicu da su učenici u svojoj školi imali istog učitelja koji je odgovoran za godišnji plan rada, koji odabire koncepte koji se podučavaju, koji bira programsko okružje ili alat u kojem se provodi podučavanje, nužno je istaknuti učitelja kao jednog od važnijih faktora koji utječe na činjenicu da će učenici odabranih škola postići bolji uspjeh od drugih na vrednovanju računalnog razmišljanja u ovom istraživanju.

Ipak, ovim istraživanjem nije moguće prepoznati na koji način djelovanje učitelja može utjecati na uspjeh učenika, jesu li to metode podučavanja, način komunikacije s učenicima u stvarnim i virtualnim učionicama, vrsta nastavnih materijala koje učitelj koristi u svom radu, planiranje zadataka aktivnog učenja za učenike te aktivnosti 'zabavnog učenja' (učenje kroz igru), aktualnost sadržaja podučavanja i slično. Analizom rezultata 10% najuspješnijih učenika na ovom vrednovanju vidi se da su dobiveni rezultati većim dijelom u skladu s očekivanim predviđanjem najuspješnijih učenika s obzirom na školu koju pohađaju (Slika 36).



Slika 36: Broj učenika svake škole koji su na vrednovanju postigli uspjeh unutar 10% najboljih

Vezano uz utjecaj varijable sati_učenja sadržaja programiranja na uspjeh vrednovanja računalnog razmišljanja važno je naglasiti da u vrijeme provođenja ovog istraživanja koncepti računalnog razmišljanja nisu bili obvezni dio službenog Nastavnog plana i programa predmeta Informatika, već su se pojedini koncepti i prakse usvajali s velikim naglaskom na vještine programiranja u nekom programskom jeziku ili grafičkom okružju za programiranje.

Novi kurikul predmeta Informatika koji stavlja jednak naglasak na koncepte računalnog razmišljanja kao i na koncepte programiranja uvodi se tek u sljedećoj nastavnoj godini. Navedeno može biti objašnjenje činjenice da duljina podučavanja takvih sadržaja nije značajno utjecala na uspjeh učenika na vrednovanju računalnog razmišljanja, a prema rezultatima višestruke regresijske analize u kombinaciji s nekim drugim faktorima može čak i negativno djelovati na uspjeh kojeg postižu učenici (Poglavlje 5.4.).

Analizom prikazanih rezultata predstavljenih u nekoliko prethodnih poglavlja u kojima se raspravlja o utjecaju općeg akademskog uspjeha, utjecaju uspjeha iz matematike, utjecaju prvog programskog jezika koji je učenik upoznao te utjecaju mentalnih modela koje usvajaju učenici na razvijeni alat vrednovanja, ispitat će se valjanost hipoteze H3.

Matematički i opći akademski uspjeh već su se prethodno pokazali kao prediktori uspjeha u programiranju, a rezultati ovog istraživanja pokazali su da se ti faktori, više ili manje, mogu smatrati prediktorima uspjeha i na ovom vrednovanju. U prethodnim poglavljima istaknuta je povezanost sadržaja koji se vrednuju ovim alatom vrednovanja te odgovarajućom provjerom iz programiranja. Prepoznata je povezanost dvaju vrednovanja (vrednovanje računalnog razmišljanja i odgovarajuće vrednovanje programiranja) kroz velik broj zajedničkih koncepata uključenih u proces vrednovanja.

Nadalje, postizanje uspjeha na ovom vrednovanju ukazuje na postojanje valjanih mentalnih modela koncepata apstrakcije i algoritamskog razmišljanja, odnosno koncepata koji predstavljaju važan preduvjet za stvaranje valjanih računalnih rješenja i postizanje uspjeha u programiranju. Rezultat vrednovanja računalnog razmišljanja iz ovog istraživanja pod značajnim je pozitivnim utjecajem većeg broja faktora koji su već prepoznati i kao prediktori uspjeha u rješavanju problema programiranjem. Iako relevantna istraživanja ističu prvi programski jezik kao faktor koji može predvidjeti uspjeh u programiranju, takav se rezultat nije potvrdio kod ovog vrednovanja. To se smatra pozitivnim obilježjem vrednovanja jer je postignut dovoljan odmak od sintakse programskog jezika i okružja u kojem je učenik do tada radio.

Obzirom da se izbor programskog jezika nije pokazao značajnim prediktorom uspjeha u vrednovanju računalnog razmišljanja u ovom istraživanju, zaključak vezano uz valjanost hipoteze H3 može se svesti na analizu utjecaja faktora uspjeha iz matematike, općeg akademskog uspjeha te valjanih mentalnih modela. Svi navedeni faktori prepoznati su kao prediktori uspjeha u programiranju, ali i ovom vrednovanja računalnog razmišljanja. Vrijedi da će učenik boljeg akademskog ili matematičkog uspjeha s razvijenim valjanim mentalnim modelima odabranih koncepta računalnog razmišljanja postići bolji uspjeh na ovom vrednovanju od učenika sa slabijim uspjehom ili s razvijenim nevaljalim mentalnim modelima. Isto se može reći i za odgovarajuću provjeru iz programiranja.

Predstavljeno vrednovanje temelji se na dokazima znanja kod učenika koji su usko povezani s njihovim mentalnim modelima. Omogućava se analiza stvarnog učeničkog razumijevanja odabranih koncepata te nudi kvalitetnija povratna informacija o razumijevanju teško mjerljivih koncepata kao što su apstrakcija, algoritamsko razmišljanje i dekompozicija od odgovarajuće provjere iz programiranja koja često stavlja veći naglasak na sintaksu programskog jezika (Poglavlje 5.4.1.). Sve navedeno govori u prilog valjanosti tvrdnje hipoteze H3 (*Ovaj alat vrednovanja računalnog razmišljanja bolji je prediktor uspjeha u programiranju od akademskog uspjeha, ocjene iz matematike te prvog programskog jezika).*

5.8. Povratne informacije učitelja i učenika koji se odnose na jasnoću i razumljivost mjernog instrumenta za vrednovanje računalnog razmišljanja

5.8.1. Opis istraživanja

U nastojanju da se procijeni kvaliteta alata vrednovanja, u sklopu istraživanja prikupljene su povratne informacije od učitelja i učenika koje se odnose na jasnoću i razumljivost alata za primjenu u nastavnoj praksi. Nakon što su učitelji proveli vrednovanje sa svojim učenicima, ispunili su izlaznu anketu kojom su se prikupile informacije vezano uz njihovu organizaciju rada i iskustvo. Cjelovit prikaz izlazne ankete za učitelje dostupan je u prilogu (PRILOG 3). Izlazna anketa uključivala je pitanja koja se odnose na:

- odabrani programski jezik/okružje za podučavanje sadržaja programiranja
- broj sati unutar Nastavnog plana i programa rezerviran za obradu sadržaja programiranja
- koncepte programiranja koji su obrađivani u nastavnom radu
- duljinu staža učitelja
- sudjelovanje u učeničkim informatičkim natjecanjima (Algoritmi, Logo, Osnove informatike, Razvoj softvera, Dabar)
- ocjenu primjerenosti alata vrednovanja.

Podatci koji se odnose na staž učitelja, broj sati koji su predviđeni za sadržaje programiranja, i odabrane koncepte programiranja obrađeni su u prethodnim poglavljima. U ovom poglavlju naglasak je na kvalitetnim povratnim informacijama koje su prikupili učitelji.

Tijekom provedbe vrednovanja u svakom razredu učitelji su promatrali učenike te bilježili kvalitetne informacije o njihovom ponašanju, pitanjima ili komentarima koje su eventualno postavljali. Bilješke i eventualne komentare zapisali su u posebno pripremljenom listiću (Tablica 48).

Nakon provedbe vrednovanja u svakom razredu, učitelji su ispunili listić s odgovarajućim povratnim informacijama u kojima su označili svako traženje pomoći od strane učenika, nejasnoće vezano uz tekst pitanja ili grafičke prikaze koji su pratili zadatke alata vrednovanja. U rubrici **Dodatna napomena** opisali su konkretne upite ili zahtjeve koje su učenici postavljali tijekom primjene alata, ako je takvih upita ili zahtjeva bilo.

Tablica 48: Listić za bilješke učitelja tijekom provedbe vrednovanja

Učenik	Upisati znak + u odgovarajući stupac svaki put kad učenik traži neku pomoć				Traži/potrebno dodatno vrijeme za rješavanje	Dodatna napomena
	Traži pomoć pri razumijevanju pitanja	**Traži pomoć pri unošenju odgovora**	**Traži pomoć zbog nerazumijevanja grafičkih prikaza u zadacima**	**Koristi papir za skiciranje pri rješavanju testa**		
					DA NE	
					DA NE	
					DA NE	

5.8.2. Analiza podataka

Kvalitativne podatke o ponašanju učenika tijekom provedbe alata vrednovanja putem posebnog listića za bilješke ispunilo je šest učitelja dok su tri učitelja povratne informacije opisali putem usmenog razgovora (intervjua) ili pisanog odgovora putem e-pošte.

Učitelji su izjavili da su se svi učenici tjedan prije natjecanja prijavili u sustav kako bi se testirala prijava na online sustav za udaljeno učenje te AAI korisnički podatci za prijavu učenika. Takav postupak bio je neophodan jer jedan dio učenika nije do tada koristio Loomen, sustav za udaljeno učenje. Također, tom prilikom učenici su ispunili ulaznu anketu kojom su se prikupili njihovi osobni podatci koji su se razmatrali u istraživanju (spol, uspjeh iz matematike, opći uspjeh, poznavanje programskih jezika/okružja, …).

Vrednovanje je provedeno jedan tjedan nakon što su učenici pristupili ulaznoj anketi. Iako su učenici nakon prve prijave mogli slobodno pristupiti posebno kreiranom kolegiju na sustavu za udaljeno učenje Loomen, alat vrednovanja nije im bio dostupan sve do samog početka vrednovanja, jedan tjedan kasnije. Za razliku od njih učitelji su imali pristup alatu kako bi prije provedbe vrednovanja provjerili primjerenost alata za primjenu u njihovom nastavnom radu te usklađenost sadržaja koji se vrednuju sa sadržajima koje su obrađivali s učenicima.

Povratne informacije učitelja ukazuju da su učenici potpuno samostalno rješavali zadatke vrednovanja uz minimalnu intervenciju ili pomoć učitelja. Kvalitativni podatci prikazani su u sljedećoj tablici i slici (Tablica 49, Slika 37).

Učenici su imali na raspolaganju 45 minuta za rješavanje zadataka i samo je 0.57% učenika tražilo omogućavanje dodatnog vremena za rješavanje. Učitelji nisu imali takve ovlasti u kolegiju pa im to nisu mogli omogućiti.

Nejasnoće oko samog teksta zadataka iskazalo je 7.39% učenika. Pitanja ili komentari koja su učenici postavljali su:

- 1. zadatak: *Kako se broje redci, odozgo prema dolje ili obratno?*
- 7. zadatak: Š*to u pitanju znači 'gore'* ?
- 10. zadatak: *Je li žaba prepreka? Učenik misli da se ne smatra preprekom, ali samo provjerava!!!*

Poteškoće s unosom odgovora prijavilo je 3.13% učenika, a mogu se opisati sljedećim komentarima/pitanjima:

- 4. zadatak: *„Oznake tablica a, b i c nije dobro postavljena iznad tablica koje predstavljaju odgovore.“*
- 7. zadatak: *„U pisanju koda u odgovoru učenik pita može li riječ jedan pisati kao riječ ili kao broj“*
- 9. zadatak: „*Učenica pita gdje točno mora upisivati odgovor u text boxu ili gore u kodu gdje se nalaze ?“*

Samo 1.42% učenika tražilo je pomoć zbog nerazumijevanja grafičkih prikaza u zadacima. Naveli su sljedeće poteškoće:

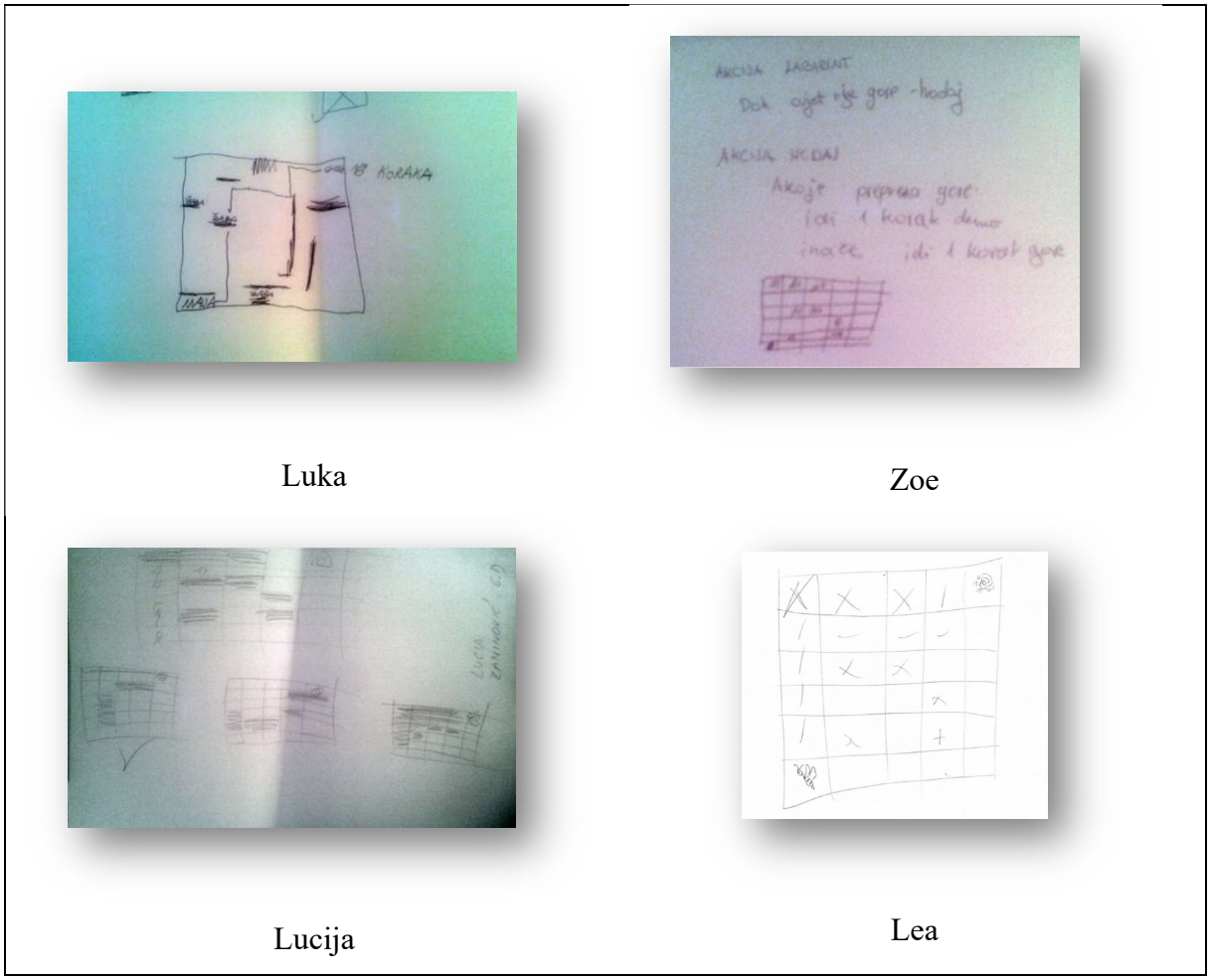
- 2. zadatak: „*Ne vidi se dobro siva boja na poljima koji predstavljaju prepreku.“*
- 4. zadatak: „*Računalo „poludilo“ – pojavili se neki crni kvadrati po ekranu. Kasnije se smirilo.“*

Tablica 49: Bilješke učitelja za vrijeme rješavanje zadataka - kvalitativni podatci o ponašanju učenika za vrijeme provedbe vrednovanja

Traži dodatno vrijeme	Traži pomoć pri razumijevanju pitanja	Traži pomoć pri unošenju odgovora	Traži pomoć zbog nerazumijevanja grafičkih prikaza u zadacima	Koristi papir za skiciranje pri rješavanju testa
2 (0.57%)	26 (7.39%)	11 (3.13%)	5 (1.42%)	19 (5.40%)

Posebno su zanimljive bilješke koje se odnose na uporabu papira za skiciranje pri rješavanju zadataka vrednovanja. Praćenje i analiza skica koje su učenici koristili može nam pomoći pri analizi njihovih rješenja, ali i pogrešaka. Svim su učenicima na početku vrednovanja podijeljeni papir i olovka za skiciranje, ako im to bude potrebno za lakše rješavanje zadataka. Od 352 učenika samo je njih devetnaest (5.40%) koristilo papir pri rješavanju.

Sljedeća slika prikazuje neke grafičke prikaze koje su učenici stvorili prilikom rješavanja zadataka (Slika 37). Nekoliko je učenika iskoristilo papir na način da su se potpisali na početku vrednovanja, ali nakon toga papir više nisu koristili. Par učenika nacrtalo je neke crteže koji nemaju poveznicu sa zadacima vrednovanja (crtež dinosaura, livade i sl.). Takve crteže možemo opisati kao crteže učenika koji su vjerojatno samo ispunili slobodno vrijeme koje im je preostalo nakon što su riješili sve zadatke koje su znali ili su se tako osjećali ugodnije za vrijeme vrednovanja. Neki od crteža koji su imali poveznicu sa zadacima vrednovanja, prikazani su na sljedećim crtežima.



Slika 37: Grafički prikazi koje su učenici stvarali prilikom rješavanja zadataka

Iako je mali broj učenika napravio neke grafičke prikaze tijekom rješavanja zadataka, zanimljivo je primijetiti da su to većinom bile djevojčice. Pri kreiranju grafičkih prikaza učenici su pokušali ponuditi vizualni prikaz ideja i problema vezanih uz zadatak. Prema dostavljenim crtežima vidi se da su učenici uglavnom pokušali skicirati labirint te način kretanja glavnog lika

Maje kroz labirint. Na taj način provjeravali su svoje vještine praćenje i analiziranje algoritamskog rješenja.

Iako je jako mali broj učenika izrazio potrebu za pomoći pri razumijevanju problema, crtanje labirinta te kreiranje kretanja glavnog lika po labirintu praćenjem zadanog algoritma moglo bi značiti da bi takav grafički prikaz učenicima mogao biti od pomoći ako bi ga dobili u obliku listića na samom početku vrednovanja. U tom uzrastu mnogi učenici neće biti sposobni sami grafički prikazati zadani problem ili rješenje, što ukazuje i na nedovoljno razvijene sposobnosti apstrakcije, odnosno prepoznavanje i prikazivanje samo neophodnih obilježja zadatka čime se postiže stvaranje jednostavnijeg prikaza zadatka.

Postojanje gotovog listića na kojem se nalazi crtež labirinta iz svakog zadatka po kojem učenici mogu slobodno skicirati kretanje glavnog lika, moguće da bi olakšalo rješavanje zadataka. Jedan od učitelja u svojim povratnim informacijama istaknuo je upravo takvu mogućnosti kao nešto što bi bilo kvalitetno unapređenje alata vrednovanja.

Također učitelji su komentirali da alat očekuje od učenika pažljivo čitanje s razumijevanjem što, prema njihovim riječima, mnogi učenici danas ne rade često. To se može vidjeti iz pitanja koje su učenici postavljali jer su vrlo često odgovori na njihova pitanja bili uključeni u sam tekst zadatka, ali oni to nisu pažljivo pročitali (npr. treba li odgovor upisati u obliku riječi ili broja).

5.9. Povratne informacije učitelja koje se odnose na primjenjivost alata vrednovanja u redovnoj nastavnoj praksi

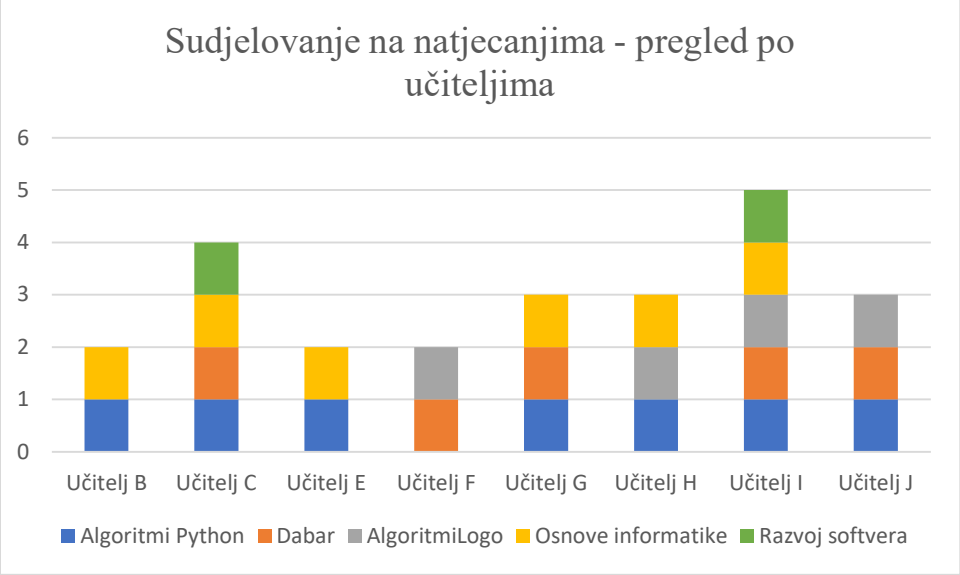
5.9.1. Opis istraživanja

U nastojanju da se prikupe završne ocjene učitelja o prikladnosti razvijenog alata vrednovanja za primjenu u redovnoj nastavnoj praksi, učitelji su u sklopu izlazne ankete o provedenom vrednovanju procijenili kvalitetu alata te iskazali svoju spremnost za primjenu alata u redovnoj nastavnoj praksi. Izlaznu anketu ispunilo je osam (8) učitelja (PRILOG 3).

U prethodnim poglavljima prikazani su podatci koji se odnose na škole, ali i učitelje informatike zaposlene u navedenim školama (primijenjeni programski jezici ili okružje, duljina poučavanja sadržaja programiranja te godine staža učitelja) (Poglavlja 5.8.).

Važno je naglasiti da, iako se radi o učiteljima koji su različitih godina podučavanja (godine staža), nadalje učiteljima koji u svom radu planiraju različitu duljinu poučavanja sadržaja programiranja te pri tome koriste različite programske jezike ili grafička okružja, svi su učitelj izuzetno aktivni i istaknuti pojedinci koji pored svojih svakodnevnih školskih obveza redovito sa svojim učenicima sudjeluju u informatičkim natjecanjima (Slika 38).

Svaki je drugi učitelj sudjelovao sa svojim učenicima u međunarodnom natjecanja računalnog razmišljanja Dabar (Bebras), svi su učitelji sudjelovali u nekom natjecanju u poznavanju algoritama pri čemu prevladava rad u programskom jeziku Python.

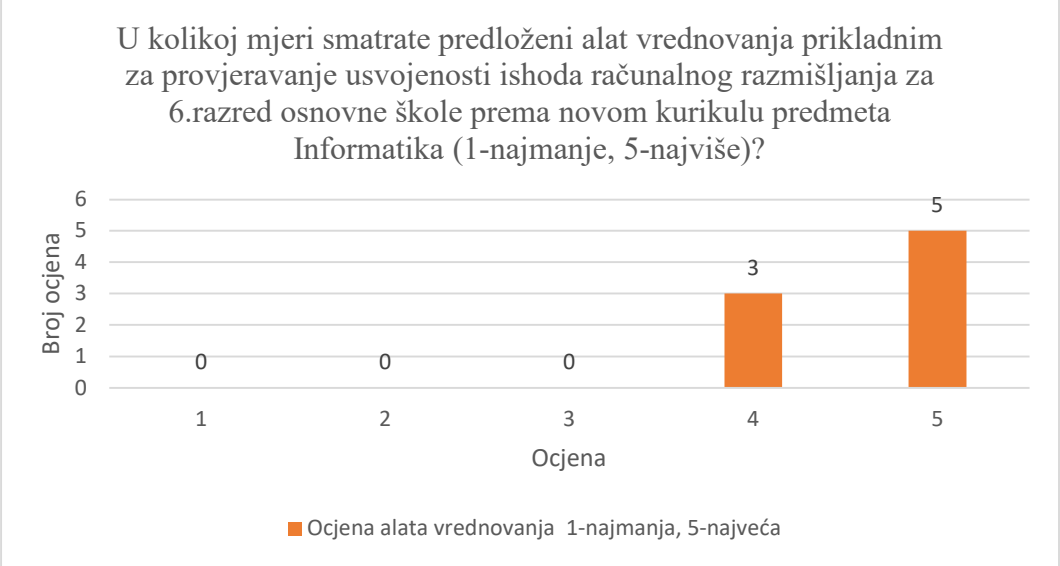


Slika 38: Sudjelovanje na natjecanjima – pregled po učiteljima

Svaki drugi učitelj sudjelovao je i u natjecanju Osnove informatike koje je najviše usklađeno s Nastavnim planom i programom (danas Godišnjim izvedbenim kurikulom). To se natjecanje odnosi na sve nastavne sadržaje koje učenici usvajaju u nastavi Informatike, a ne samo sadržaje programiranja.

5.9.2. Analiza podataka

Učitelji su ocijenili prikladnost alata vrednovanja za primjenu u redovnom nastavnoj praksi s ocjenama od jedan do pet pri čemu je ocjena jedan (1) predstavljala najmanju prikladnost, a ocjena pet (5) najveću prikladnost. Srednja ocjena alata prema učiteljima je 4,625 (Slika 39).



Slika 39: Prikladnost alata vrednovanja za primjenu u nastavnoj praksi - ocjene učitelja

Svi učitelji koji su ispunili izlaznu anketu, odgovorili su da bi primijenili alat vrednovanja sa svojim učenicima u sklopu redovnog procesa vrednovanja. Među njima pet učitelja odmah bi primijenilo alat vrednovanja u svojoj svakodnevnoj praksi dok su tri učitelja istaknula da bi cijenili mogućnosti prilagodbe/izmjene zadataka ili nekog drugog obilježja alata vrednovanja. kako bi ga uspješno primijenili sa svojim učenicima.

Iako je nekoliko učitelja u svojim povratnim informacijama, bilo usmenim, bilo pisanim putem, izrazilo bojazan da su zadaci zahtjevni za njihove učenike, niti jedan učitelj nije odlučio da ne bi koristio alat vrednovanja jer su zadatci nerazumljivi, neprikladni za vrednovanje koncepata računalnog razmišljanja, preteški ili prelagani.

6. ZAKLJUČAK

Početkom procesa reforme obrazovanja u Republici Hrvatskoj započeo je proces stvaranja novih predmetnih kurikula temeljnih na ishodima učenja uz primjenu novih nastavnih strategija i vrednovanja. Uvođenjem novog kurikula predmeta Informatika i dalje se naglašava važnost poznavanja osnovnih informatičkih pojmova poput programiranja, algoritama i struktura podataka. Potiče se razvoj računalnog razmišljanja s primarnim ciljem poticanja samostalnosti i kreativnosti pri uporabi informacijsko-komunikacije tehnologije, pri rješavanju zadataka i problema u svakodnevnom životu.

Uvođenjem računalnog razmišljanja u predmetni kurikul potiče se primjena novih nastavnih metoda i sadržaja, ali i istraživanje kvalitetnih primjera vrednovanja takvih koncepata. Znanstvena zajednica još uvijek nije ponudila dovoljno kvalitetnih preporuka i postupaka vrednovanja takvih koncepata, odnosno kvalitetnu podršku učiteljima praktičarima koji se u svom radu bave računalnim razmišljanjem.

Ovo istraživanje prikazalo je model vrednovanja računalnog razmišljanja koji je temeljen na dokazu i usklađen s novim kurikulom predmeta Informatika. Nastojeći pronaći odgovore na ključna pitanja, istražena je prikladnost predloženog modela vrednovanja za primjenu u svakodnevnoj nastavnoj praksi u šestom razredu osnovne škole Republike Hrvatske.

U razvijenom modelu vrednovanja stavljen je naglasak na koncepte apstrakcije, algoritamskog razmišljanja te dekompozicije. Rezultati su pokazali da su pojmovi algoritamskog razmišljanja i apstrakcije u ovom vrednovanju jasno prepoznati dok je koncept dekompozicije, usprkos uočenim vezama s algoritamskim razmišljanjem i apstrakcijom, bio nedovoljno predstavljen česticama alata vrednovanja kako bi se istaknuo kao zasebni faktor faktorske analize. Ipak, odabrane čestice alata pokazale su sposobnost vrednovanja koncepta dekompozicije.

Nadalje, sposobnost alata vrednovanja, razvijenog prema zadanom modelu, za vrednovanje navedenih koncepata računalnog razmišljanja potvrđena je i kroz njegovu povezanost s drugim odgovarajućim alatom vrednovanja računalnog razmišljanja, zadatcima natjecanja Dabar. Iako se na promatranom uzorku alat vrednovanja pokazao uspješnijim u vrednovanju koncepata računalnog razmišljanja od zadataka natjecanja Dabar (koeficijent pouzdanosti, usklađenost s ishodima predmetnog kurikula, međusobna povezanost čestica alata…), za pouzdaniji zaključak trebalo bi provesti istraživanje na većem uzorku.

Utjecaj programskog jezika ili okružja s kojima se učenici susreću tijekom procesa učenja i podučavanja, pokazao se nebitnim za uspjeh na ovom vrednovanju. To je od iznimne važnosti jer se time ispunio početni cilj da se ponudi model za kreiranje kvalitetnih zadataka vrednovanja koji neće staviti naglasak na sintaksu ili pravila pojedinog programskog jezika, odnosno neće biti još jedan 'alat za vrednovanje vještina programiranja'.

Jedna od boljih karakteristika razvijenog modela vrednovanja predstavlja mogućnost primjene u svrhu formativnog vrednovanja. Kreiranjem okvira vrednovanja koji se temelji na uzorcima dizajna te dokazima znanja koje učenici demonstriraju u svojim rješenjima, stvorila se mogućnost razmatranja i analize mentalnih modela ključnih koncepata kod učenika. S obzirom da se koncepti računalnog razmišljanja mogu ubrojiti u teško mjerljive koncepte, pravovremeno prepoznavanje neodrživih mentalnih modela te eventualnih pogrešaka u razumijevanju od ključne je važnosti. Učeniku se nudi brza kvalitetna povratna informacija, a učitelj u skladu s navedenom informacijom može pravovremeno prilagoditi svoj proces kurikularnog planiranja.

Iako se u ovom istraživanju učitelj istaknuo kao jedan od značajnih faktora koji doprinose razvoju računalnog razmišljanja, nije moguće odrediti koja su njegova obilježja pokazala najveći utjecaj. Svakako su specifična obilježja pojedinih učenika i škola pokazala snagu predviđanja uspjeha pa će učenici nekih škola vjerojatno biti uspješniji od drugih, kao i učenici koji imaju bolje matematičke sposobnosti. Premda je istraživanje pokazalo da prethodno poznavanje programskog jezika ili okružja neće utjecati na uspjeh pri ovom vrednovanju, analiza sadržaja vezanih uz programiranje koji su se podučavali u školama, istaknula je razlike među školama, odnosno učenicima. Može se reći da će se nastavni sadržaji birati upravo u skladu s odabranim programskim jezikom ili okružjem. S obzirom da se duljina podučavanja odabranih sadržaja nije pokazala kao značajan faktor pri postizanju uspjeha na ovom vrednovanju, pretpostavlja se da je tijekom procesa podučavanja stavljen puno veći naglasak na vještine programiranja nego na same koncepte računalnog razmišljanja. To je očekivani rezultat s obzirom da postoje značajne razlike u duljini podučavanja među školama te da tadašnji Nastavni plan i program nije isticao koncepte računalnog razmišljanja već samo programiranje.

U skladu s navedenim rezultatima te povratnim informacijama učitelja koji su izjavili da bi primijenili model vrednovanja u svom nastavnom radu, može se reći da bi ovaj model vrednovanja mogao biti prikladan za primjenu u svakodnevnoj nastavnoj praksi. Primjenom modela stvaraju se različiti zadatci, a time i alati vrednovanja, čija se obilježja mogu dalje

nadograđivati u skladu s zahtjevima modela kako bi se npr. postigao bolji stupanj pouzdanosti kao i veća kvaliteta pojedinih čestica alata (blago povećanje broja čestica, unapređenje čestica s niskim indeksom diskriminativnosti i sl.). To je ujedno i preporuka učitelja praktičara uključenih u istraživanje - ponuditi model koji omogućava razvijanje nadogradivih alata vrednovanja koje će učitelji bolje uklopiti u svoju nastavnu praksu.

Razvijenim modelom vrednovanja omogućava se vrednovanje koncepata računalnog razmišljanja neovisno o programskom alatu ili okružju. Model stavlja naglasak na koncepte računalnog razmišljanja umjesto na sintaksu programskog jezika. Iako je primjena modela vrednovanja, koji se temelji na dokazima znanja za koje učitelj pretpostavlja da će se pojaviti kod učenika, svakako dugotrajan i zahtjevan posao, model zauzvrat nudi bogatu povratnu informaciju i djelomičan uvid u stvarno znanje učenika.

Zajednica praktičara učitelja Informatike sve više ističe potrebu za ozbiljnijim istraživanjem područja vrednovanja računalnog razmišljanja te prepoznavanjem i isticanjem primjera dobre prakse. U svakodnevnoj nastavnoj praksi u kojoj se koriste različiti programski jezici i okružja, stvaranje jednog kvalitetnog modela vrednovanja temeljenog na dokazima znanja i uzorcima dizajna, oblikovanim kao online alat vrednovanja uz zadržavanje konteksta tijekom procesa vrednovanja te stvaranjem dovoljnog odmaka od usvojenih programskih jezika ili okružja, može poslužiti kao predožak za izradu kvalitetnih formativnih i sumativnih vrednovanja.

LITERATURA

Acara (2015). *Australian Curriculum - Digital Techologies.* Retrieved from ACARA: http://www.acara.edu.au/_resources/Digital_Technologies_-_Sequence_of_content.pdf.

Adelson, B., & Soloway, E. (1985). The role of domain experience in software design. *Transactions on Software Engineering 11*.

Aho, A. V. (2012). Computation and Computational Thinking. *The Computer Journal,* 55(4): 832-835.

Allert, J. (2004). Learning Style and Factors Contributing to Success in an Introductory Computer Science Course,". *Proceedings of the IEEE International Conference on Advanced Learning Technologies - ICALT'04.*

Almond, R. G., Steinberg, L. S., & Mislevy, R. J. (2002). Enhancing the Design and Delivery of Assessment Systems: A Four Process Architecture. *The Journal of Technology, Learning, and Assessment,* 1(5).

Amabile, T. (1996). Creativity and Innovation in Organizations. *Harvard Business School Background Note 396-239, January 1996.*

Ambrosio, A. P., & Martins, S. W. (2011). "What Matters Most When Teaching CS1? *ITICSE'11* pp. 27-29. Darmstadt, Germany.

Araujo, A. L., Santos, J. S., Andrade, W. L., Guerrero, D. D., & Dagienė, V. (2017). Exploring computational thinking assessment in introductory programming courses. *IEEE Frontiers in Education Conference (FIE),*Indianapolis, IN pp. 1-9.

Arlin, P. K. (1975). Cognitive development in adulthood: A fifth stage? *Developmental Psychology 11(5):* 602–606.

Armoni, M., Meerbaum-Salant, O., & Ben-Ari, M. (2015). From Scratch to "Real" Programming. *ACM Transactions on Computing Education, 14(4): 1-15; https://doi.org/10.1145/2677087*.

Astrachan, O., Hambrusch, S., Peckham, J., & Settle, A. (2009). The present and the Future of Computational Thinking. Chattanooga, Tennessee, USA, pp. 549-550. ACM 978-1-60558-183-5/09/03.

Bebras Challenge (2019). Retrieved from Countries: https://www.bebras.org/?q=countries

Bell, T., Witten, I. H., & Fellows, M. (2015). *CS Unplugged: An enrichment and extension programm for primary-aged students.* Retrieved from Cs Unplugged: http://csunplugged.org/wp-content/uploads/2015/03/CSUnplugged_OS_2015_v3.1.pdf

Ben-Ari, M. (1998). Constructivism in Computer Science Education. *iSIGSCE 98.* Atlanta, GA, USA.

Bennedsen, J., & Caspersen, M. (2005). An Investigation of Potential success Factors for an introductory model-driven programming course. *Proc. of the 1st. Intl. Computing Education Research Workshop, ICER 2005.*

Berry, D. M. (1995.). The importance of ignorance in requirements engineering. *J. Syst.Softw. 28*: 179–184.

Bienkowski, M. (2015). Making computer science a first-class object in the K-12 Next Generation Science Standards (Lightning Talk). *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* New York, NY: pp. 513. ACM https://doi.org/10.1145/2676723.2691882.

Bienkowski, M., Snow, E., Rutstein, D., & Grover, S. (2015). Assessment Design Patterns for Computational Thinking Practices in Secondary Computer Science: a first look. Menlo Park, CA: SRI Educatio.

Bienkowski, M. (2016). Deepening learning in high school computer science through practices in the NGSS (Abstract Only). SIGCSE '16: *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* pp. 694. https://doi.org/10.1145/2839509.2850557.

Bienkowski, M., Snow, E., McElhaney, K., Xu, Y., Rutstein, D., & Tate, C. (2017). *Studying Implementation of Secondary Introductory Computer Science: Preliminary Results (Abstract Only).* Proceedings of the ACM SIGCSE Technical Symposium on CompuEducation SIGCSE '17 New York, NY pp. 703-703. ACM http://dl.acm.org/citation.cfm?id=3022432.

Blackwell, A. F., Church, L., & Green, T. (2008). *The Abstract is 'an Enemy':Alternative Perspectives to Computational Thinking*. Retrieved from PPIG: www.ppig.org.

Bocconi, S., Chioccariello, A., Dettori, G. F., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education – Implications for policy and practice.* EUR 28295 EN.

Boom, K. D., Bower, M., Arguel, A., Siemon, J., & Scholkmann, A. (2018). Relationship between Computational Thinking and a Measure of Intelligence as a General Problem-Solving Ability. *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education* pp. 206-211. ACM https://doi.org/10.1145/3197091.3197104.

Bootstrap World. (2015). *Curriculum and software*. Retrieved from iz Bootstrap materials: http://www.bootstrapworld.org/materials/fall2015/index.shtml.

Bornat, R., Dehnadi, S., & Simon. (2008). Mental models, Consistency and Programming Aptitude. *ACE2008.* Wollongong, Australia.

Bransford, J. D., & Stein, B. S. (1993). *The ideal problem solver: a guide for improving thinking, learning, and creativity.* New York: WH Freeman.

Brennan, K., & Resnik, M. (2012). New frameworks for studying and assessing the development of computational thinking. *AERA.*

Briggs, S. R., & Cheek, J. M. (1986). The role of factor analysis in the development and evaluation of personality scales. *Journal of Personality,* 54: 106-148.

Brođanac, P., Bubica, N., Kralj, L., Markučić, Z., Mirković, M., Rubić, M., & Sudarević, D. (2016). *Nacionalni kurikulum nacstavnog predmeta Informatika*. Retrieved from Cjelovita kurukularna reforma - predmetni kurikulumi: http://www.kurikulum.hr/wp-content/uploads/2016/03/Informatika.pdf.

Bryson, M., Bereiter, C., Scardamalia, M., & Joram, E. (1991). Going beyond the problem as given: Problem solving in expert and novice writers. U . J. (Eds.), *Complex problem solving: Principles and mechanisms,* pp. 61–84. Lawrence Erlbaum Associates, Inc.

Bubica, N. (2018). *Vrednovanje računalnog razmišljanja*. Retrieved from Loomen: https://loomen.carnet.hr/course/view.php?id=6151.

Bubica, N., & Boljat, I. (2014). Predictors of Novices Programmers' Performance. *ICERI2014 Proceedings,* pp. 1536-1545. Seville, Spain.

Bubica, N., & Boljat, I. (2018). Assessment of Computational Thinking. *CTE2018.* Hong Kong.

Bubica, N.; Boljat, I. (2015). Programming Novices' Mental Models. *EDULEARN15 Proceedings*, pp. 5882-5891.

Byrne, P., & Lyons, G. (2001). "The Effect of Student Attributes on Success in Programming,". *ITiCSE 2001.* 6101 Canterbury, UK.

Cantwell Wilson, B., & Shrock, S. (2001). Contributing to Success in an Introductory Computer Science Course: A Study of Twelve Factors. *SIGCSE 2001.* Charlotte, NC, USA.

Carver, S. M. (1986). *Transfer of LOGO Debugging Skill: Analysis, Instruction, and Assessment.* Ph.D. Dissertation, Carnegie-Mellon.

Center for Computational Thinking, Carnogie Mellon. (2012). *Center for Computational Thinking*. Retrieved from Carnogie Mellon: https://www.cs.cmu.edu/~CompThink/

Chi, M. T., Glaser, R., & Rees, E. (1982). Expertise in problem solving. U. In R. J. Sternberg (Ed.), *Advances in the psychology of human intelligence, 1:* 7-76. Hillsdale, NJ: Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Chmura, G. A. (1998,). What Ablilities are Necessary for Success in Computer Science? . *ACM SIGCSE Bulletin 30(4): 55-58.* https://doi.org/10.1145/306286.306316.

Cohen, L., Manion, L., & Morrison, K. (2000). *Research Methods in Education, 5th Edition.* London: Routledge Falmer.

Cohen, L., Manion, L., & Morrison, K. (2007). *Reearch Methods in Education, sixth edition.* New York, USA: Routledge.

Computer Science for High School. (2016). *Computer Science for High School*. Retrieved from Computer Science for High School: https://www.cs4hs.com/.

Computing at School. (2013). *Computing in the National Curriculum.* Retrieved from Computing at School; Department for Education: http://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf.

Computing at School. (2014). *Computing in the national curriculum: a guide for secondary teachers.* Retrieved from Computing at School, Department for Education: http://www.computingatschool.org.uk/data/uploads/cas_secondary.pdf.

Computing in the Core. (2016). *Computing in the Core*. Retrieved from Code.org: https://code.org/computing-in-the-core.

Craik, K. J. (1943). *The nature of explanation.* University Press, Macmillan.

CS Education Research Group; University of Canterbury. (2015). *CS Unplugged: Computer Science without a computer*. Retrieved from http://csunplugged.org/.

Csizmadia, A., Curzon, P. P., Dorling, M., Humphreys, S., Ng, T., Selby, D. C., & Woollard, D. J. (2015). Computational Thinking - a guide for teachers. Computing At School.

CSTA (2016). *Interim CSTA K12 CS Standards: Level 3B*. Retrieved from CSTeachers.org: http://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/Docs/Standards/2016StandardsRevision/INTERIM_StandardsFINAL_07222.pdf.

Curran, J. R. (2019). *Coding and Computational Thinking - What is the evidence?,* NSW Department of Education.

Curzon, P., Dorling, M., Ng, T., Selby, C., & Woollard, J. (2014). *Developing computational thinking in the classroom: a framework.* Retrieved from Computing At School: https://pdfs.semanticscholar.org/e4f3/c24924c5a707a196b2015494c829c15618d1.pdf.

Dagiene, V., & Stupiriene, G. (2016). Informatics Concepts and Computational Thinking in K-12 Education: A Lithuanian Perspective. *Journal of Information Processing 24(4):* 732–739.

Dann, W., Cosgrove, D., Slater, D., Culyba, D., & Cooper, S. (2012). Mediated Transfer: Alice 3 to Java. *SIGCSE'12*. Raleigh, North Carolina, USA.

Davidson, J. E., & Sternberg, R. J. (2003). *The psychology of problem solving.* Cambridge University Press.

Denning, P. J. (2010). The Great Principles of Computing. *American Scientist, 98*: 369-372.

Denning, P. J. (2009). The Proffesion of IT Beyond Computational Thinking. *Communications ot the ACM, 52(6*): 28-30.

Detterman, D. K., & Sternberg, R. J. (1993). *Transfer on trial: Intelligence, cognition, and instruction.* Ablex Publishing.

Developing Computational Thinking. (2015). *Developing Computational Thinking*. Retrieved from Teaching London Computing: a Resource Hub from CAS London: https://teachinglondoncomputing.org/resources/developing-computational-thinking/.

Dierbach, C., Hochheiser, H., Collins, S., Jerome, G., Ariza, C., Kelleher, T., Kaza, S. (2011). A Model for Piloting Pathways for Computational Thinking in a General Education Curriculum. *SIGCSE'11* Dallas, Texas, USA pp. 257-262. ACM 978-1-4503-0500-6/11/03.

DIMACS (2021). *Compitational Thinking for High School Teachers*. Retrieved from What is Computational Thinking?, https://ctpdonline.org/computational-thinking/.

Djurdjevic-Pahl, A., Pahl, C., Fronza, I., & Ioini, N. E. (2017). A Pathway into Computational Thinking in Primary Schools. U *Lecture Notes in Computer Science.* Springer International.

Dorling, M., & Walker, M. (2014). *Computing Progression Pathways*. Retrieved from http://community.computingatschool.org.uk/resources/1692.

Dr.Scratch. (2019). Retreived from http://www.drscratch.org/.

European Schoolnet. (2015). *Computer programming and coding: Priorities, school curricula and initiatives across Europe.* Brussells, Belgium: European Schoolnet.

Exploring Computational Thinking. (2014). *Exploring Computational Thinking*. Retrieved from Google for Education : https://www.google.com/edu/resources/programs/exploring-computational-thinking/.

Fincher, S., Robins, A., Baker, B., Box, I., Cutts, Q., Tutty, J. (2006). Predictors of Success in a First Programming Course. *Australian Computing Education Conference (ACE),52:* 189-196.

Formative Assessment. (2014). Retrieved from The Glossary of Education Reform: https://www.edglossary.org/formative-assessment/.

Fowler, T. (2020). *How Many Computer Programming Languages Are There?* Retrieved from Careerkarma: https://careerkarma.com/blog/how-many-coding-languages-are-there/.

Frensch, P. A., & Sternberg, R. J. (1989). Expertise and intelligent thinking: When is it worse to know better? U R. J. (Ed.), *Advances in the psychology of human intelligence, 5*: 157–188; Lawrence Erlbaum Associates, Inc.

Gardner, H. (1983). *Frames of mind: The Theory of multiple intelligences.* New York, NY: Basic Books.

Garneli, V., Giannakos, M. N., & Chorianopoulos, K. (2015). Computing Education in K-12 Schools: A Review of the Literature. IEEE Global Engineering Education Conference (EDUCON). Tallinn University of Technology, Tallinn, Estonia.

Glaser, R., & Chi, M. T. (1988). Overview. In M. Chi, R. Glaser, & M. Farr (Eds.), The nature of expertise pp. 15-28. Hillsdale: Erlbaum.

Gobet, F., & Simon, H. A. (1996). Recall of rapidly presented random chess positions is a function of skill. *Psychonomic Bulletin & Review 3(2)*: 159-163.

González, M. R. (2015). Computational Thinking Test: Design Guidelines and Content Validation. Proceedings of EDULEARN15, Barcelona, Spain pp. 2436-2444. doi:10.13140/RG.2.1.4203.4329.

Grover, S., & Pea, R. (2012). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher, 42(1):38-43.* doi: 10.3102/0013189X12463051.

Grover, S. (2020). Designing an Assessment for Introductory Programming Concepts in Middle School Computer Science. *SIGCSE '20,.* Portland, OR, USA pp. 678-684: ACM 978-1-4503-6793-6/20/03.

Guzdial, M. (2008). Paving the way for the Computational Thinking. *Communications of the ACM, 51(8):* 25-27.

Hair, J. J., Black, W. C., Babin, B. J., & Anderson, R. E. (2014). *Multivariate Data Analysis, seventh edition.* England; Harlow: Pearson Education Limited.

Hendrickson, A., Ewing, M., Kaliski, P., & Huff, K. (2013). Evidence - Centered Design: Recommendations for Implementation and Practice. *Journal of Applied Testing Technology, JATT, 14, Association of Test Publishers.*

Hoover, A. K., Puttick, G., Barnes, J., Tucker-Raymond, E., Fatehi, B., Harteveld, C., & Moreno-León, J. (2016). Assessing Computational Thinking in Students' Game Designs. *HI PLAY'16 Extended Abstracts.* Austin, TX.

Hunt, E. (1975). *Artificial intelligence.* New York, NY: Academic Press.

Imberman, S. P., Sturm, D., & Azhar, M. Q. (2014). Computational Thinking: Expanding The Toolkit. *Consortium for Computing Sciences in Colleges, 29(6*): 39-46.

Interdisciplinary Computatonal Thinking. (2017). Retrieved from Teaching London Computing: https://teachinglondoncomputing.org/interdisciplinary-computational-thinking/.

ISTE, & CSTA (2011). *CSTA: Operational Definition of Computational Thinking for K12 Education.* Retrieved from http://www.etch.wlwv.k12.or.us/~whs_hyman/APCS/documents/CompThinkingFlyer.pdf

Jenkins, C. (2013). A Work in Progress Paper: Evaluating a Microworlds based Learning Approach for Developing Literacy and Computational Thinking in Cross-curricular Contexts. London, United Kingdom: *WiPSCE '15*.

Jeager, R. M, (1998). Evaluating the Psychometric Qualities of the National Board for Professional Teaching Standards' Assessments: A Methodological Accounting. *Journal of Personnel Evaluation in Education,* 12(2): 189-210.

Johnson, M. (2016). *Computational Thinking for All Students*. Retrieved from https://blog.google/topics/education/computational-thinking-for-all-students_3/

Kinnunen, P., & Simon, B. (2014). My program is OK - am I? Computing freshman's experience of doing programming assignements. *Computer Science education, 22(1):* 1-28.

Kinnunen, P., Murphy, L., McCartney, R., & L. Thomas. (2007). Through the Eyes of Instructors: A Phenomenographic Investigation of Student Success. *ICER'07.* Atlanta, Georgia, USA.

Klahr, D., & Carver, S. M. (1988). Cognitive Objectives in a LOGO Debugging Curriculum: Instruction, Learning and Transfer. *Cognitive Psihology, 20:* 362-404.

Kranch, D. A. (2011). Teaching STEM to Novices: Maximize Your Effectiveness and Minimize Your Losses [Performance]. *STEMtech conference, 2011.*

Kranch, D. A. (2012). Teaching the novice programmer: A study of instructional sequences and percetion. *Educ Inf Technol, 17*: 291.-313.

Lahtinen, E., AlaMutka, K., & HannuMatti. (2005). A Study of the Difficulties of Novice Programmers. *ITiCSE'05,* pp. 14-18. Monte de Caparica, Portugal.

Lee, M. J., & Ko, A. J. (2011). Personifying Programming Tool Feedback Improves Novice Programmers' Learning u. *ICER'11.* ProvidenceRI, USA.

Lesgold, A. (1988). Problem solving . U R. J. (Eds.), *The psychology of human thought, pp.* 188–213, Cambridge University Press.

Loomen. (2018). Hrvatska akademska i istraživačka mreža – CARNET. Retreived from: https://loomen.carnet.hr/mod/book/view.php?id=358024&chapterid=62768.

Lu, J. J., & Fletcher, G. H. (2009). Thinking About Computational Thinking. ACM SIGCSE Bulletin 41(1): 260-264. Chattanooga, TN, USA. DOI:10.1145/1539024.1508959.

Lye, S., & Koh, J. (2014). Review on teaching and learning of computational thinking through programming: what is next for K-12? *Computers in Human Behaviour, 41*: 51-61.

Ma, L. (2007). *Investigating and Impruving Novice Programmers' Mental Models of Programming Concepts.* University of Strathclyde, Department of Computer & Information Science.

Ma, L.; Ferguson, J.; Roper, M.; Ross, I.; Wood, M. (2009). Improving the Mental Models Held by Novice Programmers Using Cognitive Conflict and Jeliot Visualisations. *ITiCSE'09.* Paris, France pp. 166-170.

Marghitu, D., Brahim, B., Rawajfih, T., Weaver, Y., J., & Shanahan, J. (2010). Auburn University K-12 inclusive STEM outreach programs. *SDPS 2010 Conference.* Dallas, TX USA.

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B., Wilusz, T. (2001). *A multi-national, multi-institutional study of assessment of programming skills of first-year CS students.* SIGCSE Bulletin, 33(4): 182-196.

McGrew, K. (2009). CHC theory and the human cognitive abllities project: Standing on the shoulders of the giants of psychometric intelligence research. *Intelligence, 37(1):* 1-10.

McKinsey & Company (2007). *How the World's Best-Performing School Systems Come Out on Top.*

Meerbaum-Salant, O., Haberman, B., & Pollack, S. (2015). "Computer Science, Academia and Industry" as Pedagogical Model to Enhance Computational Thinking. *ITiCSE'15*. Vilnius, Lithuania pp. 341. ACM 978-1-4503-3440-2/15/07.

Mehrotra, G. (2011). *Role of Domain Ignorance in Software Development.* Waterloo, Ontario, Canada, University of Waterloo.

Michalko, M. (2010). *Thinkertoys: A Handbook of Creative-Thinking Techniques; 2nd edition.* Toronto: Ten Speed Press.

Miller, L., Soh, L.-K., Chiriacescu, V., Ingraham, E., Shell, D. F., & Hazley, M. P. (2014). Integrating Computational and Creative Thinking to Improve Learning and Performance in CS1. *SIGCSE*. Atlanta, Georgia, USA pp. 476-480. ACM 978-1-4503-2605-6/14/03.

Ministarstvo znanosti i obrazovanja. (2006). *Nastavni plani program za OŠ.* Retrieved from Ministarstvo znanosti i obrazovanja - Osnovno obrazovanje: https://mzo.hr/sites/default/files/dokumenti/2017/06/nastavni-plan-i-program-za-os_2006.pdf.

Ministarstvo znanosti i obrazovanja. (2018). *Kurikulum predmeta Informatika.* Retrieved from Nacionalni kurikulum: https://mzo.hr/hr/rubrike/predmetni-kurikulumi.

Ministarstvo znanosti i obrazovanja. (2019a). Kurikulum međupredmetne teme Učiti kako učiti. *Narodne novine,* 82/2019-1709. Retrieved from Ministarstvo znanosti i obrazovanja: https://narodne-novine.nn.hr/clanci/sluzbeni/2019_09_82_1709.html.

Ministarstvo znanosti i obrazovanja. (2019b). Kurikulum međupredmetne teme Uporaba informacijske i komunikacijske tehnologije. *Narodne novine,* 7/2019-150. Retrieved from: https://narodne-novine.nn.hr/clanci/sluzbeni/2019_01_7_150.html.

Ministarstvo znanosti i obrazovanja. (2019c). Pravilnik o izmjenama i dopuni pravilnika o načinima, postupcima i elementima vrednovanja učenika u osnovnim i srednjim školama. *Narodne novine* 82/2019-1709. Retrieved from: https://narodne-novine.nn.hr/clanci/sluzbeni/2019_09_82_1709.html.

Ministarstvo znanosti i obrazovanja. (2019d). Kurikulum međupredmetne teme Osobni i socijalni razvoj. *Narodne novine*, 7/2019-153. Retreived from: https://narodne-novine.nn.hr/clanci/sluzbeni/2019_01_7_153.html

Mislevy, R. J., & Haertel, G. (2006). Implications for evidence-centered design for educational assessment. *Educational Measurement: Issues and Practice,* 25: 6-20.

Mislevy, R. J., Almond, R. G., & Lukas, J. F. (2003). *A Brief Introduction to Evidence-centered Design.* NJ 08541: Research & Development Division Princeton.

Mislevy, R. J.; Riconscente, Michelle, M. (2005). *Evidence-Centered Assessment Design: Layers, Structures, and Terminology.* SRI International.

Newell, A., & Simon, H. A. (1972). *Human problem solbing.* Englewood Cliffs, NJ: Prentice-Hall.

Nolan, V. (2010). Synectics As A Creative Problem-Solving Technique. Retrieved from SynecticsWorld: https://synecticsworld.com/synectics-as-a-creative-problem-solving-technique/.

Norman, D. A. (1983). *Learning and Memory.* W H Freeman & Co.

Nunnally, J. C. (1978). *Psychometric theory (2nd ed.).* New York: McGraw-Hill.

Pallant, J. (2005). *SPSS Survival Guide: A Step by Step Guide to Data Analysis Using SPSS for Windows 3rd Edition.* New York: Open University Press.

Pallant, J. (2011). *SPSS survival manual: A step by step guide to data analysis using the SPSS program. 4th Edition.* Berkshire: Allen & Unwin.

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas.* New Year. ACM. 0-465-04627-4.

Pea, R. D. (1987). *Logo Programming and Problem Solving.[Technical Report No. 12.]*.

Pejić, P., Tuhtan-Maras, T., & Arrigoni, J. (2007). Suvremeni pristupi poticanju dječje darovitosti kreativnim radionicama. *Magistra Iaderina, 2(2)*: 133-149.

Pellegrino, J. W. (2020). Important Considerations for Assessment to Function in the Service of Education. *Educational Measurement: Issues and Practice, 39(3):* 1–5.

Pillay, N., & Jugoo, V. R. (2005). An Investigation into Student Characteristics Affecting Novice Programming Performance. *inroads – The SIGCSE Bulletin, 37(4*): 107-110.

Pioro, B. T. (2006). Introductory computer programming: gender, major, discrete mathematics and calculus. *J. Comput. Small Coll. 21(5)*: 123-129.

Porter, L., Zingaro, D., & Lister, R. (2014). Predicting Students Success using Fine Grain Clicker Data. *ICER'14.* Glasgow, Uniteg Kingdom pp. 51-58. https://doi.org/10.1145/2632320.2632354.

Raadt, M. d., Toleman, M., & Watson, R. (2004). Training Strategic Problem Solvers. *SIGSCE Bulletin, 36(2*): 48-51.

Rabkin, L. A. (2013). Empirical Analysis of Programming Language Adoption. *OOPSLA'13*,. Indianapolis, Indiana, USA pp. 29-31.

Repenning, A., & Ioannidou, A. (2008). Broadening participation through scalable game design. *ACM Special Interest Group on Computer Science Education Conference (SIGCSE 2008).* Portland, Oregon USA pp. 305-309. ACM Press https://doi.org/10.1145/1352135.1352242.

Rodrigez, B. R. (2015). *Assessing Computational Thinking in Computer Science Unplugged Activities* (Master Thesis izd.). Colorado, Scool of Mines: Colorado School of Mines. Arthur Lakes Library.

Rodriguez, B., Kennicutt, S., Rader, C., & Camp, T. (2017). Assessing Computational Thinking in CS Unplugged Activities. SIGCSE '17: Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education, Seattle, WA, USA pp. 501–506. https://doi.org/10.1145/3017680.3017779.

Rodriguez, B., Rader, C., & Camp, T. (2016). Using Student Performance to Assess CS Unplugged Activities in a Classroom Environment. ITiCSE ’16 Arequipa, Peru: pp. 95-100. DOI: http://dx.doi.org/10.1145/2899415.2899465.

Roman-Gonzales, M. (2014). Aprender a programar 'apps' como enriquecimiento curricular en alumnado de alta capacidad. *Bordon, Revista de Pedagogia,* 66(4)*:* 135-155.

Romero, M., Lepage, A., & Lille, B. (2017). Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education*, 14(42). DOI: 10.1186/s41239-017-0080-z.

Romn-Gonzlez, M., Prez-Gonzlez, J. C., & Jimnez-Fernndez, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behaviour,* 72: 678-691.

Royal Society. (2012). *Royal Society.* Retrieved from Shut down or restart: The way forward for computing in UK schools.: http://royalsociety.org/education/policy/computing-in-schools/report/.

Saqr, M., Ng, K., Oyelere, S. S., & Tedre, M. (2021). People, Ideas, Milestones: A Scientometric Study of Computational Thinking. ACM Transactions on Computing Education, 21(3):1-17. DOI: 10.1145/3445984.

Schulte, C. (2013). Reflections on the role of programming in primary and secondary computing education. *WiPSE '13: Proceedings of the 8th Workshop in Primary and Secondary Computing Education* pp. 17–24. https://doi.org/10.1145/2532748.2532754.

Schulte, C., Busjahn, T., Clear, T., Paterson, J., & Taherkhani, A. (2010). An Introduction to Program Comprehension for Computer Science Educators *in ITiCSE-WGR’10*, pp. 26-30.

Selby, C., & Woollard, J. (2013). *Computational Thinking: The Developing Definition*. Retrieved from Univerity of Southampton: http://eprints.soton.ac.uk/356481/.

Sentance, S., & Csizmadia, A. (2015). Teachers’ perspectives on successful strategies for teaching Computing in school. IFIP TC3 Working Conference 2015: A New Culture of Learning: Computing and Next Generations. Vilnius, Lithuania.

Sheard, J., Carbone, A., Markham, S., Hurst, A. J., Casey, D., & Avram, C. (2008). Performance and Progression of first year ICT students. *Proceedings of the 10th Australian Computing Education Conference - ACE 2008,* 78:119-127. Wollongong, Australia.

Siegmund, J., Kästner, C., Apel, S., Parnin, C., Bethmann, A., Leich, T., Brechmann, A. (2014). Understanding Understanding Source Code with Functional Magnetic Resonance Imaging. *ICSE '14.* Hyderabad, India. ACM 978-1-4503-2756-5/14/05.

Simon, Fincher, S., Robins, A., Baker, B., Box, I., Cutts, Q., Tutty, J. (2006). Predictors of Success in a First Programming Course. *Australian Computing Education Conference (ACE' 06),* 52: 189-196.

Somolanji, I., & Bognar, L. (2008). Kreativnost u osnovnoškolskim uvjetima. *Život i škola, 54(19)*: 87-94.

Srce. (2008). *Općenito o sustavu AAI@EduHr*. Retrieved from Autentikacijska i autorizacijska infrastruktura sustava znanosti i visokog obrazovanja u Republici Hrvatskoj: https://www.aaiedu.hr/.

SRI Education. (2020). *Principle Assessment of Computational Thinking*. Retrieved from https://pact.sri.com/.

Sternberg, R. J. (1996). *Successful intelligence.* New York, NY: Simon and Schuster.

Sternberg, R. J. (1998). Styles of Thinking and Learning. *Canadian Journal of School Psychology,* 13(2): 15-40.

Sternberg, R. J. (1999). *Uspješna inteligencija: kako praktična i kreativna inteligencija određuju uspjeh u životu.* Zagreb: BARKA.

Sternberg, R. J. (2000). The Theory of Successful Intelligence. *Gifted Education International,* 15(1): 4-21.

Sternberg, R. J. (2004). *Kognitivna psihologija.* Jastrebarsko: "Naklada Slap".

Sternberg, R. J. (2006). The nature of creativity. *Creativity Research Journal, 18 (1)*: 87–98.

Sternberg, R. J.; Lubart, Todd I. (1996). investing in creativity. *American Psychologist,* 51(7): 677-688.

Suradnici u učenju. (2020). *Dabar - međunarodno natjecanje iz informatike i računalnog razmišljanja*. Retrieved from Suradnici u učenju: http://ucitelji.hr/dabar/.

Swan, K. (1989). Programming Objects To Think With: Logo and the Teaching and Learning of Problem Solving. Annual Meeting of the American Educational Research Association. San Francisco, USA.

Tedre, M., & Denning, P. J. (2016). The Long Quest for Computational Thinking. *Proceedings of the 16th Koli Calling Conference on Computing Education Research*. Koli, Finland pp. 120-129.

Terman, L. M. (1916). *The measurement of intelligence An explanation of and a compile guide for the use of the Stanford revision and extension of the Binet-Simon intelilgence scale* . Boston, MA: Houghton-Mifflin.

The New Zeland Curriculum Online. (2014). *The New Zeland Curriculum Online: Technlogy - Achievement objectives.* Retrieved from The New Zeland Curriculum Online: http://nzcurriculum.tki.org.nz/The-New-Zealand-Curriculum/Technology/Achievement-objectives#collapsible8.

Thomas, L. (2019). *Formative Assessment*. Retrieved from George Lucas Educational Foundation: https://www.edutopia.org/article/7-smart-fast-ways-do-formative-assessment.

Thompson, T. G., & Barnes, R. E. (2009). *The Engine of Successful Educational Reform: Effective Teachers and Principals.* Denver, USA.

Touretzky, D. S., Marghitu, D., Ludi, S., Bernstein, D., & Ni, L. (2013). *Accelarating K12 Computational Thinking using Scaffolding, Staging and Abstraction.* Denver, Colorado, USA pp. 609-614. ACM 978-1-4503-1868-6/13/03.

Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes.* Cambridge, MA: Harvard University Press.

Watson, C. L., & Godwin, J. L. (2014). No Tests Required: Comparing Tradicional and Dynamic Predictors of Programming Success. *Sigcse 2014.* Atlanta, GA, USA pp. 469-474. https://doi.org/10.1145/2538862.2538930.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Tecnology,* 25(1*): 127-147; published online*.

Weisberg, R. W. (1999). Creativity and knowledge: A challenge to theories. *Handbook of creativity*, p. 226.

Werner, L., Dener, J., & Campe, S. (2012). The Fairy Assessment: Measuring Computational Thinking in Middle School. *SIGSCE'12.* Raleigh, North Carolina, USA pp. 215-220. https://doi.org/10.1145/2157136.2157200.

White, G., & Sivitanides, M. (2003). An empirical investigation of the relationship between success in mathematics and visual programming courses. *Journal of Information of Systems Education,* 14(1): 409-416.

Wilson, B. C., & Shrock, S. (2001). Contributing to Success in an Introductory Computer Science Course: A Study of Twelve Factors. *SIGCSE Bulletin,* 33(1): 184-188. https://doi.org/10.1145/366413.364581

Wing, J. M. (2006). Computational thinking. *Communication of the ACM, 49(3):* 33-35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, pp. 3717-3724. https://doi.org/10.1098/rsta.2008.0118.

Wing, J. M. (2010). Computational thinking: What and Why? Retrieved from https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf.

Winslow, L. E. (1996.). Programming Pedagogy -- A Psychological Overview. *SIGCSE Bulletin* 28(3): 17-25.

Yadav, A., Gretter, S., Jon Good, & McLean, T. (2017). Computational Thinking in Teacher Education. *Emerging Research, Practice, and Policy on Computational Thinking, Educational Communications and Technology: Issues and Innovations*, pp. 205-219.

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational Thinking for Teacher Education . *Communications of the ACM 55,* 60(4): 55-62.

Zarevski, P. (2012). *Struktura i priroda inteligencije.* Jastrebarsko: Naklada Slap.

Zhang, B. R. (2019). *The Overlooked Value of "Use" in "Use-Edit-Create"*. Retrieved from Exploring Math through Code: https://researchideas.ca/mc/httpresearchideas-camchttpresearchideas-camchttpresearchideas-camcthe-overlooked-value-of-use-in-use-edit-create/.

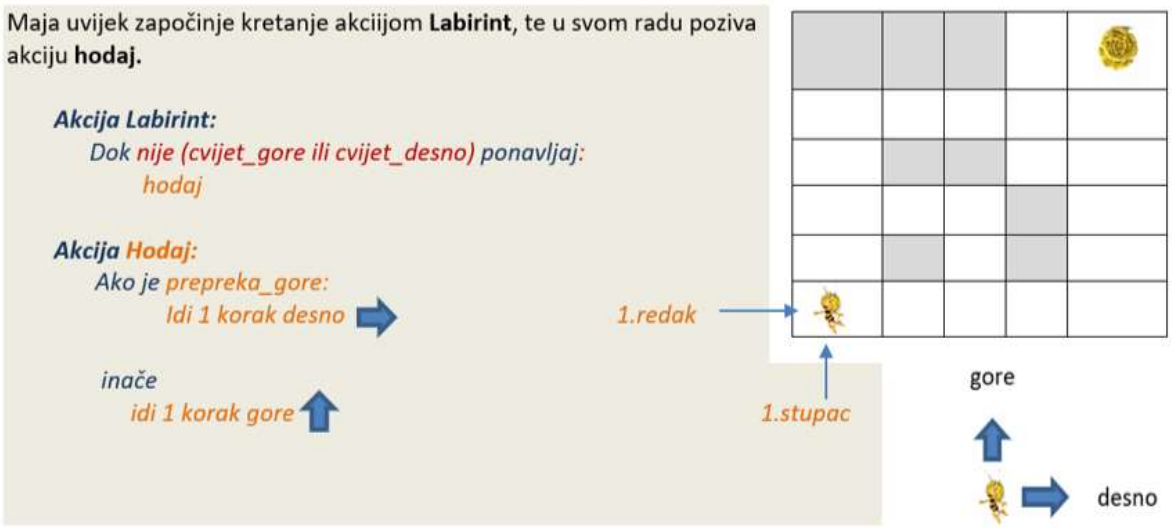
PRILOZI

PRILOG 1. Mjerni instrument za vrednovanje računalnog razmišljanja

Zadatak 1

Pčelica Maja ima zadatak pronaći svoj omiljeni žuti cvijet koji se nalazi u labirintu. Maja zna gdje se nalazi cvijet, ali ne i gdje se nalaze prepreke, sivo obojani kvadratići te rubovi labirinta. Pomogni Maji u pronalasku njenog omiljenog cvijeta. Za prolaz do cvijeta Maja će koristiti akciju **Labirint,** koja poziva akciju **Hodaj.**

Pčelica Maja uvijek počinje kretanje iz prvog stupca i prvog retka. Smjer gledanja Pčelice Maje nije bitan!



Na kojoj će se poziciji (redak, stupac) naći Pčelica Maja nakon što napravi 5 koraka u skladu s dopuštenim akcijama kretanja u labirintu?

Redak= {1:SHORTANSWER:=5} Stupac={1:SHORTANSWER:=2}

Zadatak 2:

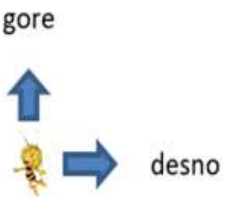
Ako pogledamo sliku labirinta, primijetit ćemo kako se može dogoditi da Maja, zbog prepreka te dozvoljenih akcija kretanja, neće biti u mogućnosti doći do svog cvijeta. **Prepreke predstavljaju siva polja te rubovi labirinta. Smjer gledanja Pčelice Maje nije bitan!**

Maja uvijek započinje kretanje akcijom **Labirint**, te u svom radu poziva akciju **Hodaj.**

Akcija Labirint:
Dok nije (cvijet_gore ili cvijet_desno) ponavljaj:
hodaj

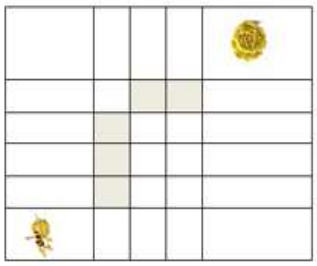
Akcija Hodaj:
Ako je prepreka_gore:
Idi 1 korak desno

inače
idi 1 korak gore

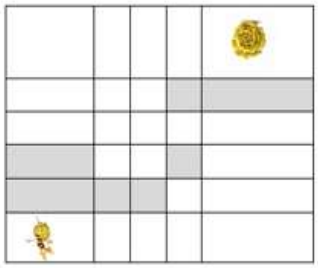


U kojem od sljedećih primjera labirinta Pčelica Maja neće biti u mogućnosti stići do svog cvijeta dopuštenim akcijama kretanja?

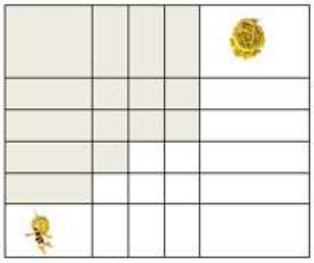
a)



b)



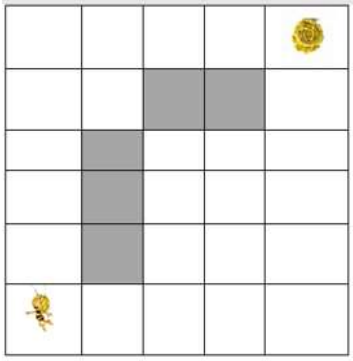
c)



Zadatak 3:

Slika prikazuje jedan labirint te upute za kretanje pčelice Maje po labirintu. **Maja bi trebala uvijek krenuti prvo gore, a ako to nije moguće, onda pokušati krenuti desno**.

No, u navedenim uputama akcije **Hodaj** krije se pogreška. Navedene upute ne odgovaraju dogovorenom pravilu. Pronađi pogrešku te napiši ispravne upute kretanja po labirintu.



Akcija Labirint:

Dok nije (cvijet_gore ili cvijet_desno) ponavljaj:

hodaj

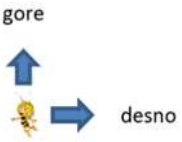
Akcija Hodaj:

Ako nije prepreka_gore:

Idi 1 korak desno

inače

idi 1 korak gore



Pronađi i popravi pogrešku u akciji hodaj!

Akcija Hodaj:

Ako nije prepreka_gore:

Idi 1 korak desno

inače

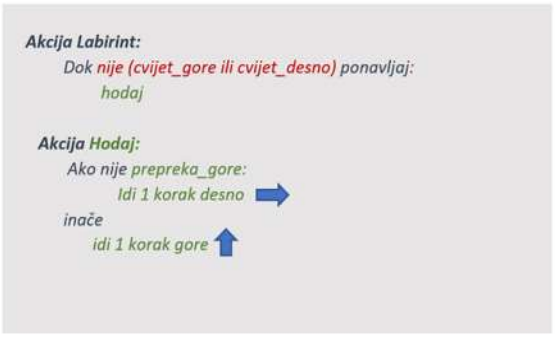
Idi 1 korak gore

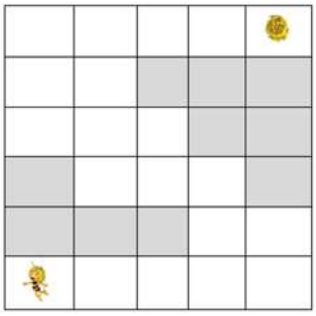
Zadatak 4:

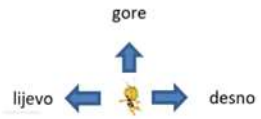
Budući da je Maja u prošlim primjerima zaglavila u labirintu, naučimo je kako se može kretati lijevo u labirintu. Umetnimo novu naredbu 

.

Novu naredbu moramo uključiti u postupak kretanja kroz labirint na način na Maja može ići lijevo **tek kada je sigurna da ne može više ići niti gore niti desno.**







Koju je od sljedećih akcija potrebno izmijeniti kako bi se Maja mogla kretati lijevo po labirintu, naravno ako je to moguće, s obzirom na prepreke? Ponovimo, prepreke su sivi kvadratići te rubovi labirinta.

a) Akciju **Labirint**
b) Akciju **Hodaj**
c) Nije potrebno vršiti nikakve izmjene
d) Napisati novu akciju koja sadrži samo tu jednu naredbu za kretanje 1 korak lijevo bez izmjene postojećih akcija Hodaj i Labirint.

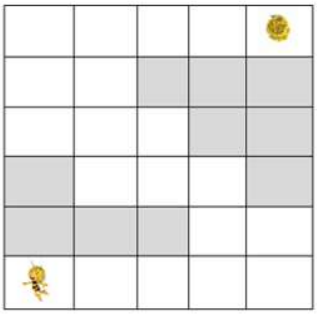
Zadatak 5:

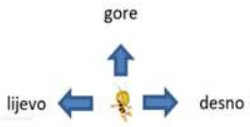
Kako bismo olakšali kretanje pčelice Maje po labirintu, pokušali smo napraviti nekoliko izmjena u akciji **Hodaj** za zadani labirint na slici. Maja će uvijek moći napraviti korak (u nekom od smjerova gore, desno ili lijevo).

Akcija Labirint:

Dok nije (cvijet_gore ili cvijet_desno) ponavljaj:
hodaj

Akcija Hodaj:
????





a)
Ako je prepreka_gore:
Idi 1 korak desno
inače
Idi 1 korak gore

b)
Ako je prepreka_gore:
Idi 1 korak desno
inače
Ako je prepreka_lijevo:
Idi 1 korak desno
inače
Idi 1 korak gore

c)
Ako je prepreka_gore:
Ako je prepreka_desno:
Idi 1 korak lijevo
inače
Idi 1 korak desno
inače
Idi 1 korak gore

Koja od navedenih verzija akcija **Hodaj** omogućava uspješno kretanje po labirintu (dakle Maja neće stati zbog prepreke)? A, B ili C?

Zadatak 6:

Ubrzali smo Majino kretanje po labirintu izmjenom akcije **Hodaj** tako da se **Maja kreće prema gore sve dok ne naiđe na prepreku, a tek onda skrene desno ili lijevo**.

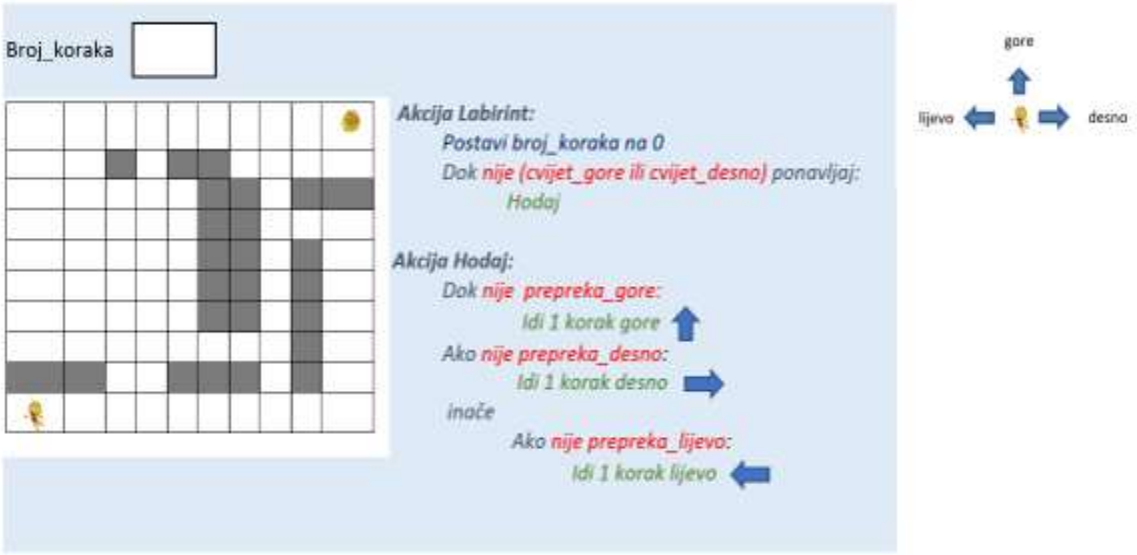
Promotri sljedeći labirint i provjeri može li Maja prema novim uputama stići do cvijeta.

Koje su od sljedećih tvrdnji istinite? (Može biti više točnih odgovora, a svaki se netočni odgovor negativno boduje (-0.5).

a) Pčelica Maja ovim će algoritmom napraviti barem tri koraka nakon što prvi put promijeni smjer.
b) Ovim algoritmom Pčelica Maja neće uspjeti doći do svog cvijeta.
c) Ovim algoritmom Pčelica Maja barem će jednom ići lijevo.
d) Pčelica Maja će napraviti barem tri koraka prije nego promijeni smjer kretanja.

Zadatak 7:

Pčelica Maja odlučila je pratiti broj koraka koje napravi na svom putu prema labirintu te bilježiti ih u oznaci **broj_koraka** (na slici iznad labirinta). Na početku svakog kretanja kroz labirint vrijednost **broj_koraka** Maja uvijek postavlja na 0



Nadopuni Majine upute kretanja kroz labirint tako da vrijednost **broj_koraka** prikazuje uvijek ukupan broj napravljenih koraka. Dakle, na odgovarajuće mjesto u akciji **Hodaj** potrebno je napisati tvoju verziju upute kojom se vrijednost **broj_koraka** poveća za 1 svaki put kad se napravi jedan korak.

Akcija Labirint:

Postavi broj_koraka na 0

Dok nije cvijet_gore ponavljaj:

hodaj

Akcija Hodaj:

Dok nije prepreka_gore ponavljaj:

Idi 1 korak gore

Ako nije prepreka _desno:

Idi 1 korak desno

inače:

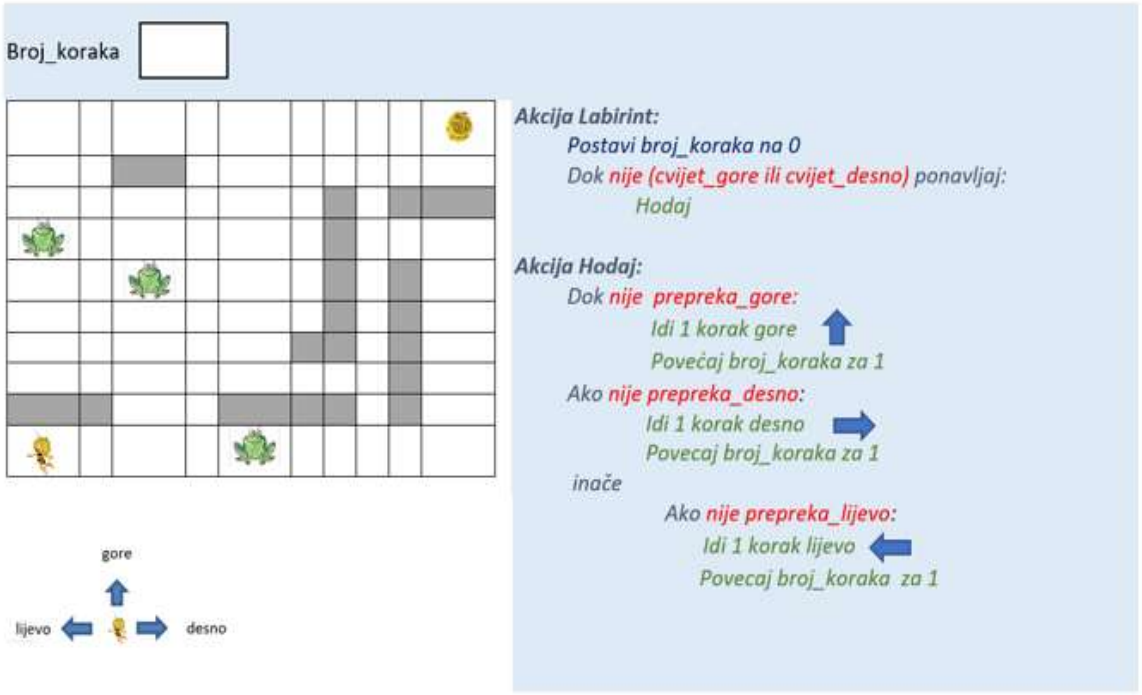
Ako nije prepreka_lijevo:

Idi 1 korak lijevo

Zadatak 8:

U labirintu su se na određenim mjestima pojavile opasne žabe koje žele zaustaviti Pčelicu Maju na njenom putu prema žutom cvijetu. Maja **SMIJE** preskočiti žabu samo kada ide u smjeru prema **gore,** stoga moramo promijeniti način kretanja prema gore.

Kako bi pomogli Maji uvodimo novo pravilo: **Ako Maja, pri kretanju u smjeru prema gore, naiđe na žabu, može je preskočiti tako da napravi dva koraka odjednom.**



Odaberi sliku na kojoj je označena uputa koju je potrebno izmijeniti prema novim pravilima kretanja po labirintu.

a)

b)

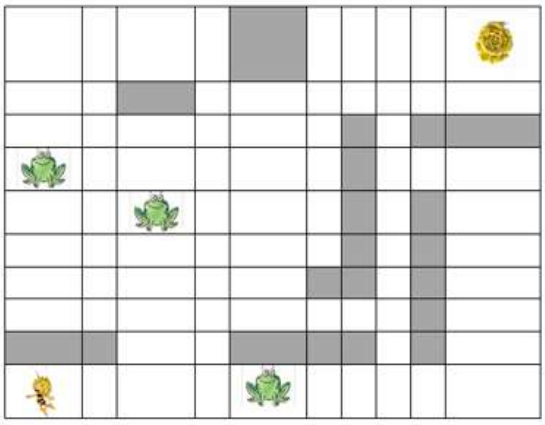
c)

Zadatak 9:

U prethodnom zadatku prepoznali smo upute koje je potrebno izmijeniti kako bi se Maja uspješno kretala po labirintu te izbjegla opasne žabe gdje je to moguće. Dakle, Maja **SMIJE** preskočiti žabu samo kada ide u smjeru prema **gore.**

Kako Maja preskače žabu? **Ako Maja, pri kretanju u smjeru prema gore, naiđe na žabu, može je preskočiti tako da napravi dva koraka odjednom.**

Broj_koraka



```
Akcija Labirint:
    Postavi broj_koraka na 0
    Dok nije ( cvijet_gore ili cvijet_desno) ponavljaj:
        Hodaj

Akcija hodaj:
    Dok nije prepreka_gore:
        Idigore_i_preskocizabu
    Ako nije prepreka_desno:
        Idi 1 korak desno
        Povecaj broj_koraka za 1
    inače
        Ako nije prepreka_lijevo:
            Idi 1 korak lijevo
            Povecaj broj_koraka za 1

Akcija Idigore_i_preskocižabu:
    ???
```

Nadopunimo upute za kretanje Pčelice Maje po labirintu tako da može izbjegavati žabe kada se kreće prema gore.

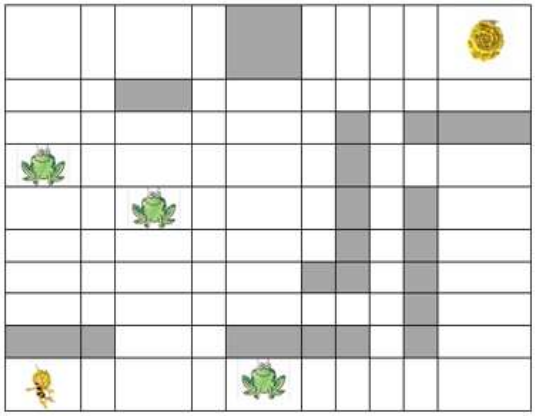
Napiši upute za kretanje u obliku nove akcije **Idigore_i_preskocizabu** kojom se omogućava da Maja ide 2 koraka gore, ako je žaba_gore, odnosno 1 koraka gore ako nije.

*Akcija **Idigore_i_preskocizabu:***

Zadatak 10:

Promotri sljedeći labirint te niz uputa.

Broj_koraka



Akcija Labirint:
Postavi broj_koraka na 0
Dok nije (cvijet_gore ili cvijet_desno) ponavljaj:
Hodaj

Akcija hodaj:
Dok nije prepreka_gore:
Idigore_i_preskocizabu
Ako nije prepreka_desno:
Idi 1 korak desno
Povecaj broj_koraka za 1
inače
Ako nije prepreka_lijevo:
Idi 1 korak lijevo
Povecaj broj_koraka za 1

Akcija Idigore_i_preskocižabu:
Ako je žaba_gore:
Idi 2 koraka gore; povećaj broj_koraka za 2
inače
Idi 1 korak gore; povećaj broj_koraka za 1

Koliko je koraka pčelica Maja napravila prije nego se zaustavila (došla je do cvijeta ili je naišla na prepreku zbog koje morala zaustaviti jer je nije znala zaobići) ? Odgovor napiši u obliku broja. {1:SHORTANSWER:=15}

PRILOG 2. Ulazna anketa za učenike

(Pitanje 1) Spol*

○ Ženski ○ Muški

(Pitanje 2) Koliko imaš godina?*

(Pitanje 3) Koji razred pohađaš?*

○ 5. razred OŠ
○ 6. razred OŠ
○ 7. razred OŠ
○ 8. razred OŠ
○ 1. razred SŠ
○ 2. razred SŠ
○ 3. razred SŠ
○ 4. razred SŠ

(Pitanje 4) Jesi li ikada programirao u nekom programskom jeziku?*

○ DA ○ NE

(Pitanje 5) programski jezik

☐ SCRATCH
☐ LOGO
☐ PYTHON
☐ KODU
☐ BASIC
☐ Neki drugi programski jezik

(Pitanje 6) Unesi svoj prosjek ocjena na kraju prošle školske godine (uspjeh) ?*

(Pitanje 7) Unesi prosjek ocjena iz predmeta Matematika (e-dnevnik)*

PRILOG 3. Izlazna anketa za učitelje

Pitanje 1. Upišite naziv škole u kojoj ste zaposleni (ako ste zaposleni u dvije škole upišite školu u kojoj provodite istraživanje): *

Obvezatno

Pitanje 2. U kojem se gradu nalazi škola?*

Pitanje 3. U kojoj se županiji nalazi vaša škola?

Pitanje 4. Koliko ste godina zaposleni kao učitelj informatike?*

Pitanje 5. Koliko se ukupno sati u vašem godišnjem planu rada odnosi na cjelinu programiranja (ove školske godine) (0 - 70)

Pitanje 6. Koji ste od navedenih programskih jezika koristili pri obradi nastavnog gradiva iz programiranja? Pitanje se odnosi na vaše učenike koji sudjeluju u ovom istraživanju o vrednovanju računalnog razmišljanja.

- [ ] Logo
- [ ] Python
- [ ] Scratch
- [ ] Kodu

- [ ] Basic
- [ ] Small Basic
- [ ] nešto drugo

Pitanje 7. Koje ste nastavne sadržaje iz programiranja do sada (od početka izborne nastave 5.razreda) obrađivali s ovim učenicima na razini dovoljnoj za rješavanje zadataka iz istraživanja?*

- [ ] Pojam algoritma
- [ ] Rad s izlaznim vrijednostima (npr, print...)
- [ ] Naredbe za crtanje (kornjačina grafika)
- [ ] Rad s ulaznim vrijednostima (npr. naredba input)
- [ ] Izrada i uređenje blok dijagrama nekog algoritma
- [ ] Jednostavno grananje (npr, naredba if ... then, if ....)
- [ ] Složeno grananje (npr. naredba if...then...else..., if..else....)
- [ ] Ugnježđeno grananje (npr. naredba if....elif....)
- [ ] Petlje s određenim brojem ponavljanja (npr. naredba for...., foreach...)
- [ ] Petlje s uvjetnim ponavljanjem (npr. naredba while...)
- [ ] Potprograme

Pitanje 8. Sudjelujete s učenicima na informatičkim natjecanjima?*

- ○ DA
- ○ NE

Pitanje 9. U kojim ste kategorijama natjecanja do sada sudjelovali sa svojim učenicima?

- [ ] Algoritmi (Python/Basic)
- [ ] Logo
- [ ] Osnove informatike
- [ ] Razvoj softvera (Smotra radova)
- [ ] Dabar

Pitanje 10. U kolikoj mjeri smatrate predloženi alat vrednovanja prikladnim za provjeravanje usvojenosti ishoda računalnog razmišljanja za 6. razred osnovne škole prema novom kurikulu predmeta Informatika (1-najmanje, 5-najviše) (1 - 5)*

Pitanje 11. Biste li primijenili navedeni alat vrednovanja računalnog razmišljanja u svom nastavnom radu?*

- ( ) Ne
- ( ) Da
- ( ) Vjerojatno, pogotovo ako bi ga mogao/mogla dodatno uređivati/mijenjati

Pitanje 12. Ako NE želite koristiti predloženi alat vrednovanja računalnog razmišljanja, odaberite koji je od razloga utjecao na vašu odluku.

- [ ] Smatram da zadaci nisu razumljivi.
- [ ] Smatram da zadaci nisu prikladni za vrednovanje koncepata računalnog razmišljanja.
- [ ] Smatram da su zadaci preteški.
- [ ] Smatram da su zadaci prelagani.
- [ ] Ništa od navedenog.

PRILOG 4. Listić za praćenje aktivnosti, pitanje te općenito ponašanja učenika tijekom rješavanja alata vrednovanja

Učenik	Upisati znak + u odgovarajući stupac svaki put kad učenik traži neku pomoć				Traži/potrebno dodatno vrijeme za rješavanje	Dodatna napomena
	Traži pomoć pri razumijevanju pitanja	**Traži pomoć pri unošenju odgovora**	**Traži pomoć zbog nerazumijevanja grafičkih prikaza u zadacima**	**Koristi papir za skiciranje pri rješavanju testa**		
Učenik1					DA NE	
Učenik2					DA NE	
Učenik3					DA NE	
Učenik4					DA NE	
Učenik5					DA NE	
Učenik6					DA NE	
Učenik7					DA NE	
Učenik8					DA NE	
Učenik9					DA NE	
Učenik10					DA NE	
Učenik11					DA NE	
Učenik12					DA NE	
Učenik13					DA NE	
Učenik14					DA NE	
Učenik15					DA NE	
Učenik16					DA NE	
Učenik17					DA NE	
Učenik18					DA NE	
Učenik19					DA NE	
Učenik20					DA NE	
Učenik21					DA NE	
Učenik22					DA NE	
Učenik23					DA NE	
Učenik24					DA NE	
Učenik25					DA NE	
Učenik26					DA NE	
Učenik27					DA NE	
Učenik28					DA NE	
Učenik29					DA NE	
Učenik30					DA NE	

Ime i prezime učitelja: _____

Naziv škole: _____

Grad, županija: _____

datum provedbe vrednovanja: _____

razred/grupa učenika: _____

potpis učitelja: _____

PRILOG 5: Odabrani zadaci natjecanja Dabar 2017 (Suradnici u učenju, 2020)

Zadatak 1:

PAKIRANJE JABUKA

Oznaka zadatka: 2017-TW-03 | **Tip pitanja: višestruki odgovor**

Ključne riječi: oktalni brojevni sustav, binarni brojevni sustav

ZADATAK

Obitelj dabrova ima voćnjak s jabukama. Tijekom sezone berbe dabrovi traže najbolji način pakiranja jabuka za dostavu na tržnicu. Ove godine odlučili su pakirati jabuke prema sljedećim pravilima:

1. Jabuke se stavljaju u vreće. Svaka vreća sadrži točno 8 jabuka. Ako ostane manje od 8 jabuka, one se ostavljaju izvan vreće.
2. Vreće se stavljaju u kutije. Svaka kutija sadrži točno 8 vreća. Ako ostane manje od 8 vreća, one se ostavljaju izvan kutija.

PITANJE

Danas su dabrovi nabrali 275 jabuka. Koliko kutija, vreća i jabuka su dabrovi spakirali?

PONUĐENI ODGOVORI

A. 

B. 

C. 

D. 

TOČAN ODGOVOR

B)

OBJAŠNJENJE

Svaka kutija sadrži ukupno 64 jabuke (8 vreća puta 8 jabuka po vreći). Dabrovi su nabrali 275 jabuka i napunili 4 kutije (275/64 = 4.296875). Te četiri kutije sadrže ukupno 64*4 = 256 jabuka. Kako svaka vreća sadrži 8 jabuka, ostatak jabuka (275-256 = 19) je stavljeno u 2 vreće izvan kutija (19/8 = 2.375). Te dvije vreće sadrže 16 jabuka što znači da 3 jabuke ostaju izvan vreća.

Provjerimo:

Izbor A ukupno sadrži 3*64 + 7*8 + 7 = 255 jabuka

Izbor C ukupno sadrži 3*64 + 5*8 + 1 = 233 jabuka

Izbor D ukupno sadrži 4*64 + 1*8 + 6 = 270 jabuka

RAČUNALNA POVEZANOST

Binarni brojevni sustav je najjednostavniji oblik računalnog koda ili informacije. Sastoji se isključivo od binarnih brojeva, odnosno slijeda nula i jedinica. S druge strane, oktalni sustav koristi osam različitih simbola, vrijednosti od 0 do 7. Ako je potrebna bilo koja vrijednost veća od 7, dodatni stupci dodaju se s lijeva. Vrijednost svakog stupca je 8 puta veća od vrijednosti stupca s njegove desne strane. Oktalni sustav omogućuje da se ista informacija prikaže koristeći samo 1/3 znamenki jer svaka oktalna znamenka predstavlja tri binarne znamenke. To je bilo idealno za neke od starih računalnih sustava zato što je na taj način bilo moguće sažeto prikazati određeni strojni kod. U zadatku, kutije predstavljaju 8^2 jabuka, preostale vreće 8^1 jabuka, a preostale jabuke 8^0 (jedinica). Prema tome, dekadski broj 275 je prikazan kao 423 u oktalnom brojevnom sustavu.

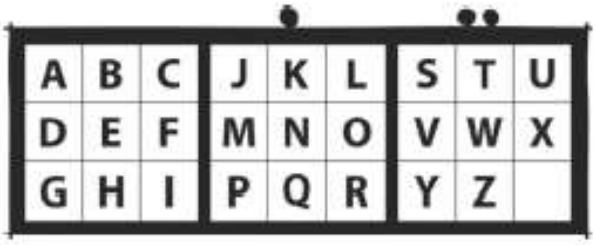
Zadatak 2:

MREŽNA ABECEDA

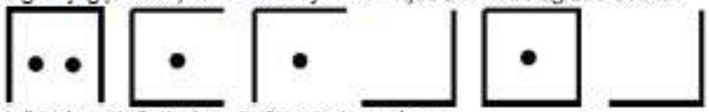
Oznaka zadatka: 2017-LT-09	Tip pitanja: Kratki odgovor
Ključne riječi: šifriranje, kodiranje, kod	

ZADATAK

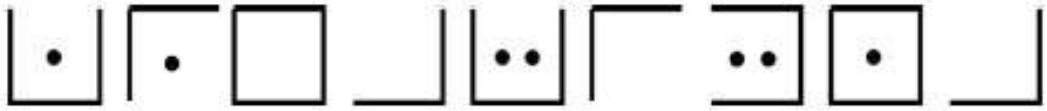
Dabrica Zorana voli kodirati riječi. U novinama je pronašla predložak za kodiranje koji je sastavljen od 3 tablice. Svaka se tablica sastoji od 3 stupca i 3 retka koji su ispunjeni slovima.



Kada je, uz pomoć gornjeg prikaza, kodirala svoje ime - riječ ZORANA izgleda ovako:



Zatim je kodirala još jednu riječ. Kod te riječi izgleda ovako:



PITANJE/IZAZOV

Koju je riječ dabrica Zorana napisala?
Napomena: u polje za odgovor napiši samo tu riječ!

TOČAN ODGOVOR

KREATIVNA

OBJAŠNJENJE

Oznaka ruba za svako slovo znači gdje se to slovo nalazi u mreži, a točke znače u kojoj se od tri mreže nalazi slovo.

RAČUNALNA POVEZANOST

Ovakav način kodiranja zove se šifra zamjenom, svako slovo zamjenjuje se određenim simbolom. Ovakva šifra je geometrijski jednostavna, i obično je svako slovo iz rešetke predstavljeno simbolom koji je dio rešetke. Bez sheme rešetke (šifra ključa) teško je dešifrirati tekst (za bolje šifre gotovo nemoguće). Zbog jednostavnosti šifriranja često se koristi u dječjim knjigama o šiframa i tajnom pisanju. U informatici enkripcija je jako važna za sigurnost.

Zadatak 3:

RIBE

Oznaka zadatka: 2017-JP-01	Tip pitanja: Višestruki odgovor
Ključne riječi: operacija, apstrakcija, programiranje	

ZADATAK

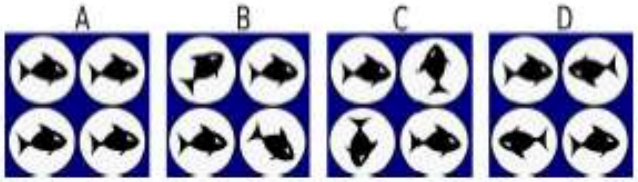
Četiri igračke ribe postavljene su na pladanj kao na slici.
Ako zakrenete bilo koju ribu za npr. 45° u smjeru kazaljke na satu, tada će se i riba koja se nalazi dijagonalno od nje zakrenuti za 45°, ali u smjeru suprotno od kazaljke na satu.

PITANJE/IZAZOV

Postupite prema uputama :
1. Ribu u gornjem lijevom kutu zakrenite za 45° u smjeru kazaljke na satu.
2. Ribu u donjem lijevom kutu zakrenite za 90° u smjeru kazaljke na satu.
3. Ribu u donjem desnom kutu zakrenite za 90° u smjeru kazaljke na satu.
4. Ribu u gornjem lijevom kutu zakrenite za 45° u smjeru kazaljke na satu.

Koji od ponuđenih odgovora prikazuje situaciju nastalu nakon gornja 4 koraka?

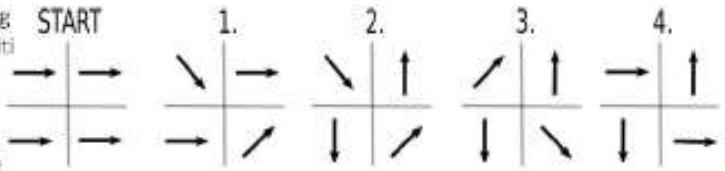
PONUĐENI ODGOVORI



TOČAN ODGOVOR

Točan odgovor je C.

OBJAŠNJENJE



Za pouzdano rješavanje ovog zadatka poželjno je koristiti što jednostavniju metodu, rješavanje napamet se ne preporučuje.
Jedan način je korištenje jednostavnih oznaka kojima pratimo promjene nakon svakog koraka. Postupak na slici dolje prikazuje jedan takav način gdje se ribe označavaju strelicama.

Drugi način pomoću kojega možemo još brže doći do rješenja jest uočavanje da samo korak 2 utječe na položaj riba u donjem lijevom i gornjem desnom kutu, pa tako učenik može pregledavanjem ponuđenih odgovora pronaći jedino rješenje koje odgovara zadanom položaju riba.

RAČUNALNA POVEZANOST

Računalni program je slijed uputa. Ručna simulacija je prvi korak u učenju programiranja.
Rješavanje ovog zadatka koristeći notaciju strelica zahtjeva mnogo apstrakcije.

- Nepotrebni detalji, kao što je vrsta igračke, su skriveni, smjer je jedino što je potrebno.
- Ključni elementi se moraju uočiti kako bi mogli znati u kojem smjeru je riba okrenuta.
- Strelica je izabrana kao najjednostavniji prikaz igračke ribe.

Zadatak 4:

SPORTOVI

Oznaka zadatka:2017-EE-01	Tip pitanja: Uparivanje odgovora
Ključne riječi: Booleove vrijednosti, formalna logika, ispunjavanje uvjeta	

ZADATAK

U sportskom centru Dabrosport nalaze se tereni za odbojku, košarku, tenis i nogomet.
Prijatelji dabrovi: Ana, Bruno, Ivan i Diana dolaze u Dabrosport i treniraju svoj najdraži sport.
Poznato je da Ana i Ivan ne koriste rekete. Odbojkaš, nogometaš i Diana imaju treninge isti dan. Nogometaš planira gledati Ivanov turnir. Bruno i nogometaš trče ujutro. Diana dijeli stan s tenisačem.

PITANJE/IZAZOV

Kojim se sportom bave ovi dabrovi? Svakom od njih pridruži odgovarajući sport.

TOČAN ODGOVOR

Točan odgovor je: Ana – nogomet, Bruno – tenis, Ivan – odbojka, Diana – košarka.

OBJAŠNJENJE

Zadatak možemo riješiti uz pomoć tablice. Čitajući rečenice zadatka u tablicu bilježimo kojim se sportom određeni dabar ne može baviti.

	Odbojka	Košarka	Tenis	Nogomet
Ana			ne	
Bruno				ne
Ivan			ne	ne
Diana	ne		ne	ne

Sada se iz tablice jasno vidi kako se Ana jedina može baviti nogometom, Bruno tenisom, a Diani je jedina opcija košarka. Preostaje Ivan, a on se može baviti samo odbojkom jer je to jedini sport koji je ostao slobodan.

RAČUNALNA POVEZANOST

Ovaj zadatak primjer je korištenja Booleovih izraza ("točno" i "netočno" vrijednosti) za pronalazak rješenja. Korištenje binarnog kodiranja informacija i njihova obrada pomoću Booleovih operacija temeljni su principi računalne znanosti.

Zadatak 5:

NATJECANJE DABROVA

Oznaka zadatka:2017-HR-04a-eng	Tip pitanja: Višestruki odgovori
Ključne riječi: Algoritmi, strukture podataka, petlje	

ZADATAK

Dabar Krešo je gledao kako se 8 dabrova u parovima natječe u utrci na 100 metara. Pobjednik utrke ide u sljedeći krug natjecanja. Dabar Krešo je promatrajući natjecanja na ploči bilježio pobjednika svake utrke. Za to je koristio kartice označene brojevima od 1 do 8 koje su predstavljale svakog dabra koji je sudjelovao u utrci.

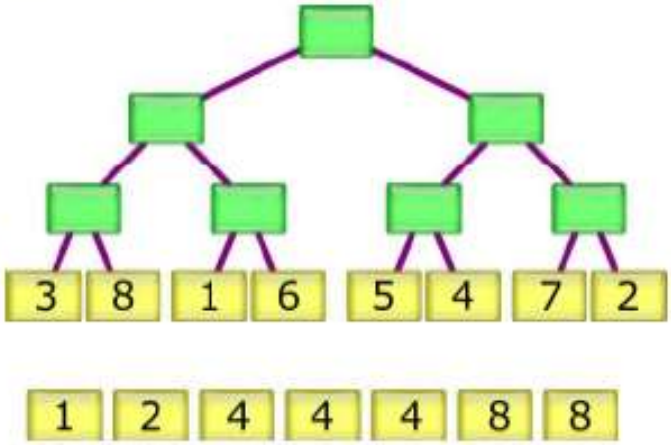
Ali onda je Krešin mlađi brat Tomo izmiješao sve kartice pobjednika utrka, osim parova prvog kruga natjecanja.

PITANJE/IZAZOV

Krešu ovo nije uzrujalo ni najmanje, shvatio je da može ponovno posložiti sve rezultate natjecanja iz pomiješanih kartica i informacija iz prvog kruga natjecanja.

Koji je pravilan redoslijed svih kartica?

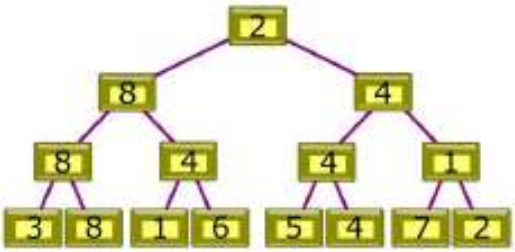
PONUĐENI

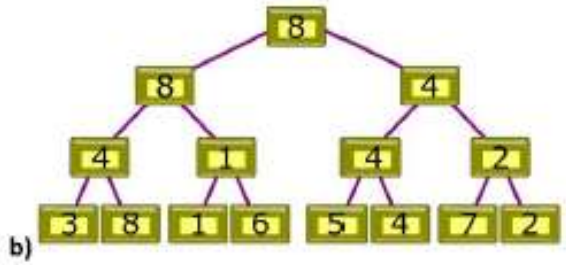


ODGOVORI

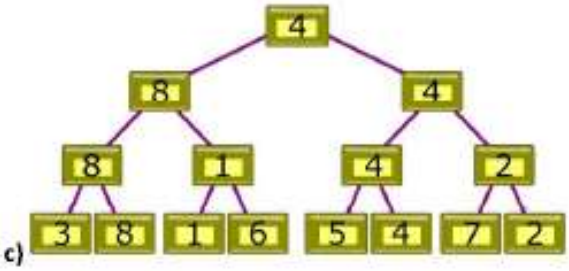
Učenici trebaju od ponuđenih odgovora izabrati točan.

a)





b)



c)

TOČAN ODGOVOR

c)

OBJAŠNJENJE

Za svaku utrku parova dabrova u prvom krugu natjecanja, broj koji odgovara dabru pobjedniku bi morao biti u izmiješanim kartama. Natjecatelji koji su izgubili, ne sudjeluju u sljedećim krugovima natjecanja. Pa kako popunjavamo polja s rezultatima, jednostavno trebamo pogledati koji od dva natjecatelja svake utrke postoji u karticama koje su preostale.

RAČUNALNA POVEZANOST

Prilikom rješavanja problema i kreiranja algoritma, važno je uočiti sve moguće uvjete koji utječu na rješenje zadatka. Uvjeti se nekad preklapaju, nekad slijede jedan drugog, a nekad jedan od uvjeta može biti ispunjen samo ako je prethodni uvjet ispunjen ili samo ako prethodni uvjet nije ispunjen. Pažljivo praćenje i zadovoljavanje potrebnih uvjeta, osnova je ispravnog logičkog rješavanja problema i izrade učinkovitog algoritamskog rješenja.

Uz provjeru zadovoljavanja uvjeta (grane grafa), ponavljanje (ponavljajuća struktura ili petlja) se često koristi pri izradi računalnih rješenja. U gornjem zadatku, uvjet provjere procedure mora biti ponovljen dok ne označimo pobjednika. Znači da broj krugova natjecanja označava broj potrebnih ponavljanja za danu akciju.

U ovom zadatku, uvjeti koje student treba uočiti su: ako je kartica s brojem ponuđena, to znači da je natjecatelj s tim brojem pobjednik. Ova se procedura ponavlja dok se sve kartice ne postave na ploču.

Zadatak 6:

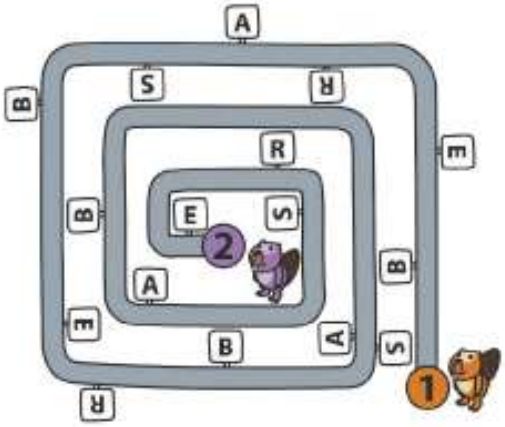
SAKUPLJANJE SLOVA

Oznaka zadatka: 2017-CY-04-eng	Tip pitanja: višestruki odabir
Ključne riječi: znakovni tip, znakovi, polje	

ZADATAK

Mario i Ivan žele prošetati po donjoj stazi. Mario želi proći stazu od točke 1 do točke 2, a Ivan proći stazom od točke 2 do točke 1.

Duž staze postavljena su slova **B, E, R, A** i **S** koje Mario i Ivan moraju skupiti. Mario i Ivan mogu pročitati samo slova postavljena duž staze s njihove lijeve strane. Kada Mario i Ivan stignu na kraj staze moraju imati skupljena slova u određenom redoslijedu.



PITANJE/IZAZOV

Koja slova imaju Mario i Ivan na kraju šetnje?

PONUĐENI ODGOVORI

A. Mario: BRSBAASE Ivan: RBSRBAE

B. Mario: BRSEBAASE Ivan: RBBSRBAE

C. Mario: BRSEBAASE Ivan: RBSRBAE

D. Mario: ‚BRSEBAAS Ivan: RBSRBE

TOČAN ODGOVOR

Točan odgovor je C.
Mario će skupiti BRSEBAASE, a Ivan ima sakupljen niz slova RBSRBAE.

RAČUNALNA POVEZANOST

Znakovni tip se općenito smatra tipom podataka koji se odnosi na polje sa znakovima spremljenim kao niz elemenata zapisanih u nekom kodu. Znakovni tip općenito može se odnositi na tip podataka i struktura za polje ili niz znakova (ili lista). U formalnom jeziku znakovni tip definira se kao konačan slijed simbola koji su odabrani iz skupa koji zovemo abeceda.

Zadatak 10:

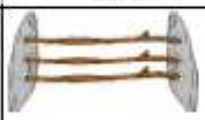
BINARNI ULAZ

Oznaka zadatka:2017-AZ-02-eng	Tip pitanja: Brojčano
Ključne riječi: binarni sustavi, kombinatorika	

ZADATAK

Dabrovi su gostoljubivi i vole se međusobno posjećivati. Kako bi njihovo međusobno posjećivanje bilo što uspješnije, na ulazu u dvorište dabrovi ostavljaju poruku gostima. Na taj će način gosti znati jesu li dabrovi doma ili ako nisu – kad će biti.

Dabrovi su smislili sljedeće 4 različite poruke:

Doma smo. Slobodno uđite!	Vratit ćemo se u podne.	Vratit ćemo se večeras.	Vratit ćemo se u ponoć.
			

Ipak, mali dabar Krešo misli da je moguće napraviti više od 4 poruke promjenom mjesta ovih grana. On zna da:

- grane moraju biti ili pričvršćene vodoravno ili ih uopće nema
- oblik i smjer u kojem su grane okrenute nije bitan.

Ali on ne zna točno koliko je poruka moguće izraditi.

PITANJE/IZAZOV

Koji je najveći broj poruka koje je moguće sastaviti, uključujući 4 originalne poruke?

TOČAN ODGOVOR

Točan odgovor je 8.

OBJAŠNJENJE

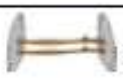
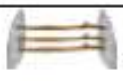
Svaka grana ima 2 moguća stanja: ili je pričvršćena ili nije.

Ukupno ima 3 grane.

To znači da je broj mogućih kombinacija 2x2x2=8.

Binarni brojevi su brojčani sustav koji predstavlja vrijednosti korištenjem dva simbola: obično su to 0 (nula) i 1 (jedan).

Sljedeće slike pokazuju svih 8 mogućih kombinacija i primjer pripadajućeg binarnog kôda:

 000	 001	 010	 011
 100	 101	 110	 111

RAČUNALNA POVEZANOST

Ovaj se zadatak bavi binarnim brojevima, povezan je s binarnim sustavima i osnovama kombinatorike.

Učenici se trebaju upoznati s matematičkim konceptima kao što su binarni brojevi, faktorijeli, i sl., jer je matematika jedna od osnovnih komponenti računalne znanosti.

Iako ovaj zadatak može biti jednostavan starijim učenicima, mlađima može pomoći da razmišljaju o kombinatorici

Zadatak 12:

GRADNJA BRANE

Oznaka zadatka:2017-IR-02	Tip pitanja: Višestruki odgovori
Ključne riječi:	

ZADATAK

Dabar Ante živi u šumi u kojoj je svako stablo visoko 10 metara. Da bi izgradio branu, dabru Anti je potrebno više komada drveta različitih dužina i to: 7 komada drveta dužine 4 metra i 7 komada drveta dužine 3 metra.

Dužina	Potrebno je
4m	7x
3m	7x

Ante treba ispiliti stabla od 10 metara na manje dijelove, ali pri tome treba paziti da ispila najmanji mogući broj stabla.

PITANJE/IZAZOV

Koji je najmanji broj stabla koji Ante mora ispiliti da bih dobio odgovarajuće dijelove za svoju branu?

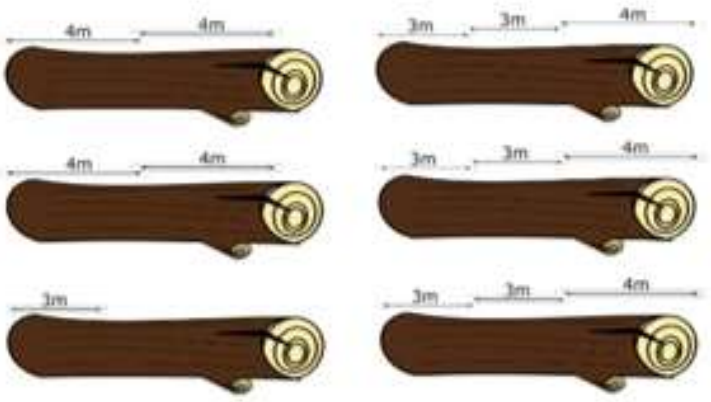
PONUĐENI ODGOVOR

a) 5
b) 6
c) 7
d) 8

TOČAN ODGOVOR

6

OBJAŠNJENJE



ŽIVOTOPIS I POPIS JAVNO OBJAVLJENIH RADOVA

Nikolina Bubica rođena je 8. rujna 1970. u Dubrovniku. Osnovnu školu završava u Dubrovniku kao i srednju školu, COUO Elektrotehničke struke, smjer matematika–informatika. Diplomirala je na Fakultetu prirodoslovno-matematičkih znanosti i odgojnih područja u Splitu, smjer matematika-informatika u siječnju 1997. godine u zvanju profesora matematike i informatike. U Osnovnoj školi Mokošica, Dubrovnik zapošljava se 1994. godine kao učitelj informatike, gdje radi i danas na poslovima učitelja Matematike i Informatike. Kratki dio svog radnog vijeka provela je u Gimnaziji Dubrovnik kao nastavnik Matematike.

Od 2005. do 2018. godine djeluje kao: ECDL predavač i ispitivač za osnovne i napredne module u sklopu KING-ICT OŠ Mokošica ispitnog centra; predavač na Pučkom otvorenom učilištu „Žižić“ Omiš, Podružnica Mokošica za predmete Matematika, Računarstvo i Statistika; edukator visokog učilišta Algebra u sklopu projekta „e-Škole“ za Dubrovačko-neretvansku županiju. Od 2005./2006. školske godine obavlja poslove voditelja ŽSV učitelja Informatike osnovnih škola Dubrovačko-neretvanske županije.

Od 2006. godine aktivno djeluje u nekoliko skupina za unapređenje informatičkog obrazovanja na području RH (Povjerenstvo za završnu izradu Plana i programa informatike, Stručna radna skupina za izradu prijedloga predmetnog kurikula Informatike u sklopu CKR, Stručna radna skupina za doradu nacrta prijedloga i izradu konačnog prijedloga predmetnog kurikula, Povjerenstvo za uvođenje Informatike kao obveznog predmeta u osnovnoškolski odgoj i obrazovanje, Radna skupina za pripremu i uvođenje kurikula Informatike za osnovnu i srednju školu).

U veljači 2017. godine izabrana je u naslovno suradničko zvanje asistenta, za znanstveno područje tehničkih znanosti, polje računarstvo pri Prirodoslovno-matematičkom fakultetu u Splitu. U lipnju 2018. godine sudjeluje u radu međunarodne konferencije o računalnom razmišljanju u obrazovanju, CTE2018 (Hong Kong), kao dobitnik njihove stipendije za visoko kvalitetan rad u području istraživanja računalnog razmišljanja.

Radovi u časopisima (Znanstveni i pregledni radovi)

Izvorni znanstveni i pregledni radovi u CC časopisima

Bubica, Nikolina; Boljat, Ivica Assessment of Computational Thinking – A Croatian Evidence-Centered Design model // Informatics in Education Vol. 2021, No.3 (2022, September).

Znanstveni radovi u drugim časopisima

Bubica, Nikolina; Boljat, Ivica It Competition From The Students' Peρpective: Their Motivation And Attitudes Toward Success. // TOJET: The Turkish Online Journal of Educational Technology, 2016 (2016), Spec. issue for INTE; 1361-1368. (https://www.bib.irb.hr/851885)

Znanstveni radovi u zbornicima skupova

Boljat, Ivica; Bubica, Nikolina; Matana, Antonela Stressors and burnout symptoms of math teachers in Croatian primary and high schools // Proceedings of EDULEARN21 Conference / Gómez Chova, L. ; López Martínez, A. ; Candel Torres, I. (ur.); Valencia, Spain: IATED Academy, 2021. str. 9307-9315 (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Bubica, Nikolina; Boljat, Ivica Stresori, simptomi izgaranja i motivacija za rad učitelja STEM predmeta u hrvatskim osnovnim školama// Knjižica znanstvenih sažetaka, (ur.): S. Dobrota, S. Tomaš, I. Restović, L. Maleš, I. Blažević, E. Jakupčević, M. Bulić/ Filozofski fakultet sveučilišta u Splitu, 2021

Bubica, Nikolina; Boljat, Ivica Assessment of Computational Thinking // Proceedings of the International Conference on Computational Thinking Education 2018 / Kong, S.C., Andone, D., Biswas, G., Crick, T., Hoppe, H.U., Hsu, T.C., Huang, R.H., Li, K.Y., Looi, C.K., Milrad,

M., Sheldon, J., Shih, J.L., Sin, K.F., Tissenbaum, M., & Vahrenhold, J. (ur.).
Hong Kong: The Education Univeρity of Hong Kong, 2018. str. 121-124.
(https://www.bib.irb.hr/958252) (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Bubica, Nikolina; Boljat, Ivica Primjena e-učenja u podučavanju programiranja mlađih učenika osnovne škole. // Treća međunarodna znanstvena konferencija - pedagogija, obrazovanje i nastava Mostar: FPMOZ, 2016.. (https://www.bib.irb.hr/851891) (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Bubica, Nikolina; Boljat, Ivica Programming novices' mental models. // EDULEARN15 Proceedings / L. Gómez Chova, A. López Martínez, I. Candel Torres (ur.).
Valencia, Spain: IATED (International Academy of Technology, Education and Development), 2015. str. 5882-5891. (https://www.bib.irb.hr/858862) (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Bubica, Nikolina; Boljat, Ivica Predictors of novices programmeρ' performance. // ICERI2014 Proceedings / L. Gómez Chova, A. López Martínez, I. Candel Torres (ur.).
Seville, Spain: IATED AInternational Academy of Technology, Education and Development, 2014. str. 1536-1545. (https://www.bib.irb.hr/738991) (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Bubica, Nikolina; Boljat, Ivica Teaching of Novice Programmeρ: Strategies, Programming Languages and Predictoρ. // Proceedings International conference on information technology and development of education ITRO 2014 / Pavlović, Milan (ur.).
Zrenjanin: Univerzitet Novi Sad, Tehnički fakultet, 2014. str. 180-185.
(https://www.bib.irb.hr/702045) (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Bubica, Nikolina; Mladenović, Monika; Boljat, Ivica Students motivation for competition in computer science. // 8th International Technology, Education and Development Conference (INTED2014) : proceedings Valencia: International Academy of Technology, Education and Development IATED, 2014. str. 288-295. (https://www.bib.irb.hr/702148) (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Bubica, Nikolina; Boljat, Ivica Strategies for teaching programming - the state of the art. // Contemporary issues in Economy & Technology CIET 2014 Split: Univeрity of Split Univeрity Department of Professional Studies, 2014. str. S-248. (https://www.bib.irb.hr/701577) (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Bubica, Nikolina; Mladenović, Monika; Boljat, Ivica Programiranje kao alat za razvoj programskog mišljenja. // 15. CARNetova korisnička konferencija - CUC 2013 - Zbornik radova / Orlović, Ana (ur.). Zagreb: Hrvatska akademska i istraživačka mreža - CARNet, 2013.. (https://www.bib.irb.hr/702093) (predavanje, domaća recenzija, cjeloviti rad (in extenso), znanstveni)

Stručne knjige

Moj portal 3.0 – udžbenik iz informatike u petom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Mario Stančić, Branko Vejnović, 2014, Školska knjiga, Zagreb

Moj portal 3.0 – radna bilježnica iz informatike u petom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Mario Stančić, Branko Vejnović, 2014, Školska knjiga, Zagreb

Kurikulum nastavnog predmeta Informatika za osnovne škole i gimnazije; Brođanac, P., Bubica, N., Kralj, L., Markučić, Z., Mirković, M., Rubić, M., Sudarević, D., 2018. Ministaрtvo znanosti i obrazovanja, Zagreb

#mojportal5 - udžbenik informatike u petom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Zoran Dimovski, Stanko Leko, Nikola Mihočka, Ivana Ružić, Mario Stančić, Branko Vejnović, 2018, Školska knjiga, Zagreb

#mojportal5 – radna bilježnica iz informatike u petom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Zoran Dimovski, Stanko Leko, Nikola Mihočka, Ivana Ružić, Mario Stančić, Branko Vejnović, 2018, Školska knjiga, Zagreb

#mojportal6 - udžbenik informatike u šestom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Zoran Dimovski, Stanko Leko, Nikola Mihočka, Ivana Ružić, Mario Stančić, Branko Vejnović, 2018, Školska knjiga, Zagreb

#mojportal6 – radna bilježnica iz informatike u šestom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Zoran Dimovski, Stanko Leko, Nikola Mihočka, Ivana Ružić, Mario Stančić, Branko Vejnović, 2018, Školska knjiga, Zagreb

#mojportal7 - udžbenik informatike u sedmom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Zoran Dimovski, Stanko Leko, Nikola Mihočka, Ivana Ružić, Mario Stančić, Branko Vejnović, 2018, Školska knjiga, Zagreb

#mojportal7 – radna bilježnica iz informatike u sedmom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Zoran Dimovski, Stanko Leko, Nikola Mihočka, Ivana Ružić, Mario Stančić, Branko Vejnović, 2018, Školska knjiga, Zagreb

#mojportal8 - udžbenik informatike u osmom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Zoran Dimovski, Stanko Leko, Nikola Mihočka, Ivana Ružić, Mario Stančić, Branko Vejnović, 2018, Školska knjiga, Zagreb

#mojportal8 – radna bilježnica iz informatike u osmom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Zoran Dimovski, Stanko Leko, Nikola Mihočka, Ivana Ružić, Mario Stančić, Branko Vejnović, 2018, Školska knjiga, Zagreb

#mojportal 5 – udžbenik iz informatike u petom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Zoran Dimovski, Mario Stančić, Nikola Mihočka, Ivan Ružić, Branko Vejnović, 2019, Školska knjiga, Zagreb

#mojportal 5 – radna bilježnica iz informatike u petom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Zoran Dimovski, Mario Stančić, Nikola Mihočka, Ivan Ružić, Branko Vejnović, 2019, Školska knjiga, Zagreb

e-SVIJET 1 - radni udžbenik za informatiku s dodatnim digitalnim sadržajima u prvom razredu osnovne škola; Josipa Blagus, Nataša Ljubić Klemše, Ana Flisar Odorčić, Nikola Mihočka, Ivana Ružić, Nikolina Bubica, 2020, Školska knjiga, Zagreb

#MOJPORTAL6 udžbenik za informatiku u šestom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Zoran Dimovski, Mario Stančić, Ivana Ružić, Nikola Mihočka, Branko Vejnović, 2020, Školska knjiga, Zagreb

#MOJPORTAL6 - radna bilježnica za informatiku u šestom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Zoran Dimovski, Mario Stančić, Ivana Ružić, Nikola Mihočka, Branko Vejnović, 2020, Školska knjiga, Zagreb

#MOJPORTAL7 - udžbenik za informatiku u sedmom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Zoran Dimovski, Mario Stančić, Ivana Ružić, Nikola Mihočka, Branko Vejnović, 2020, Školska knjiga, Zagreb

#MOJPORTAL7 - radna bilježnica za informatiku u sedmom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Zoran Dimovski, Mario Stančić, Ivana Ružić, Nikola Mihočka, Branko Vejnović, 2020, Školska knjiga, Zagreb

#MOJPORTAL8 - udžbenik za informatiku u osmom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Zoran Dimovski, Mario Stančić, Ivana Ružić, Nikola Mihočka, Branko Vejnović, 2021, Školska knjiga, Zagreb

#MOJPORTAL8 - radna bilježnica za informatiku u osmom razredu osnovne škola; Magdalena Babić, Nikolina Bubica, Stanko Leko, Zoran Dimovski, Mario Stančić, Ivana Ružić, Nikola Mihočka, Branko Vejnović, 2021, Školska knjiga, Zagreb

#mojportal6 – priručnik za učitelje uz udžbenik iz informatike u šestom razredu osnovne škole; Nikolina Bubica, Zoran Dimovski, Nikola Mihočka, Ivana Ružić, Mario Stančić, 2020, Školska knjiga, Zagreb

#mojportal7 – priručnik za učitelje uz udžbenik iz informatike u sedmom razredu osnovne škole; Nikolina Bubica, Zoran Dimovski, Nikola Mihočka, Ivana Ružić, Mario Stančić, 2020, Školska knjiga, Zagreb

#mojportal8 – priručnik za učitelje uz udžbenik iz informatike u osmom razredu osnovne škole; Nikolina Bubica, Zoran Dimovski, Nikola Mihočka, Ivana Ružić, Mario Stančić, 2021, Školska knjiga, Zagreb