Igra Križić-kružić za NxN polja u Pythonu

Vuletić, Mihaela

SVEUČILIŠTE U SPLITU
PRIRODOSLOVNO MATEMATIČKI FAKULTET

ZAVRŠNI RAD

IGRA KRIŽIĆ-KRUŽIĆ ZA NxN POLJA U PYTHONU

Mentor: prof. dr. sc. Ani Grubišić

Student: Mihaela Vuletić

Split, rujan 2024.

Temeljna dokumentacijska kartica

Završni rad

Sveučilište u Splitu
Prirodoslovno-matermatički fakultet
Odjel za informatiku i tehniku
Ruđera Boškovića 33, 21000 Split, Hrvatska

IGRA KRIŽIĆ-KRUŽIĆ ZA NxN POLJA U PYTHONU

Mihaela Vuletić

SAŽETAK

Tema ovog završnog rada bila je napraviti funkcionalnu igricu Križić-kružić za NxN polja u programskom jeziku Python. Za izradu aplikacije korišten je PyCharm, integrirano razvojno okruženje koje se koristi za programiranje u programskom jeziku Python. Netom nakon korisnik pokrene igru javlja mu se mogućnost odabira veličine ploče za igranje te odabira znaka s kojim korisnik želi igrati protiv kompjutera. Nakon završetka igre, korisniku se na ekran ispisuje rezultat, ali isto tako i upit o ponovnom igranju igre. Ukoliko odabere ponovno igranje igre, igra se ponovno pokreće, u suprotnom se igra gasi.

Ključne riječi:	Tic-tac-toe, aplikacija, PyCharm, Python, ploča, pobjednik, klasa
Rad sadrži:	24 stranica, 23 grafička prikaza, 7 literaturnih navoda Izvornik je na hrvatskom jeziku.
Mentor:	**prof. dr. sc. Ani Grubišić,** *redoviti profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*
Ocjenjivači:	**prof. dr. sc. Ani Grubišić,** *redoviti profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu* **prof. dr. sc. Branko Žitko,** *redoviti profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu* **Ines Šarić-Grgić, dipl.ing.,** *asistent Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Rad prihvaćen: **rujan, 2024.**

Basic documentation card

Thesis

University of Split
Faculty of Science
Department o Computer science and Engineering
Ruđera Boškovića 33, 21000 Split, Croatia

TIC-TAC-TOE FOR NxN FIELDS IN PYTHON

Mihaela Vuletić

ABSTRACT

The topic of this paper is to make a functional game Tic-Tac-Toe for NxN fields in programming language Python. For the making of the game is used PyCharm, an integrated development enviroment used for programming in Python. When the user turns on the game he has the option to choose the size of the board he wants to play on, also to choose the sign he wants to play with against computer. When the game is finished, user gets the results of the game as well as a question if he wants to play again. If he chooses to play again, the game starts again, if not the game turns off.

Keywords: Tic-tac-toe, application, Python, PyCharm, board, winner, class

Thesis consists of: 24 pages, 23 figures, 7 references
Originial language: Croatian

Supervisor: **Ani Grubišić, PhD,** Full p*rofessor, Faculty of Science, University of Split*

Reviewers: **Ani Grubišić, PhD,** Full p*rofessor, Faculty of Science, University of Split*

Branko Žitko, PhD, *Full professor, Faculty of Science, University of Split*

Ines Šarić-Grgić, mag. ing. comp., *asistent, Faculty of Science, University of Split*

Thesis accepted: **September 2024.**

SADRŽAJ

1 Uvod

Igra Križić-Kružić je društvena igra koja potječe još od starog Egipta (prvi znaci igre datiraju još od 1300. godine prije Krista). Isto tako i za vrijeme Rimskog carstva se igrala prva verzija igre za koju znamo danas. Znakovi da je igra bila aktualna u to vrijeme pronađeni su diljem zidova Rima, ispisane sa kredom. [1]

Za igranje igre nam je tipično potreban samo jedan papir na kojem je ucrtana rešetka sa 3x3 polja te jedno pisaće pomagalo. Svaki igrač odabire jedan znak (O ili X) te ga unutar praznih polja ucrtavaju sve dok se igra ne završi. Igrač pobjeđuje kada ostvari 3 svoja znaka uzastopno bilo u redu, glavnoj ili sporednoj dijagonali ili stupcu. Ako nijedan igrač to ne ostvari, igra se proglašava neriješenom.

Danas se igra može na isti način igrati i na računalu. Možemo ju čak i igrati protiv računala, tako da nam nije potreban drugi igrač koji bi nam u stvarnom svijetu bio potreban za igranje igre. Isto tako, imamo i mogućnost izabrati veličinu ploče na kojoj ćemo igrati koja nije tradicionalnih 3x3 dimenzija.

Zadatak ovog završnog rada je prikazati način izrade igrice Križić-kružić za neki određeni N (N označava veličinu ploče) koju igramo protiv kompjutera. Za izradu aplikacije koristit ćemo razvojno okruženje PyCharm, a kao programski jezik ćemo koristiti Python.

.

1. Python i PyCharm

Programski jezik Python je kreirao Guido van Rossum kasnih 1980-ih godina u nacionalnom istraživačkom institutu za matematiku i računalnu znanost u Nizozemskoj. Jezik je prvi put objavljen 1991. godine kao Python 0.9.0. S vremenom je postao jedan od najpopularnijih programskih jezika današnjice.

Kao programski jezik, Python je vrlo lako čitljiv, lako je s njim postupati, njegovo oblikovanje je vizualno čisto i jednostavno. Ne koristi vitičaste zagrade za omeđivanje blokova kao neki drugi programski jezici već koristi uvlačenje razmaka što ga uvelike razlikuje od drugih programskih jezika. [2]

Python dolazi s intergriranim razvojnim okruženjem (IDE). IDE je softverska aplikacija koja daje razne mogućnosti za razvoj softvera. Jedan od onih koji je korišten za izradu ove aplikacije je integrirano razvojno okruženje PyCharm koje se uvelike preporuča za programiranje u Pythonu.

PyCharm je razvila češka tvrtka JetBrains 2010. godine. To je aplikacija koja je višeplatformna, radi na Microsoft Windowsima, macOS-u, a i na Linuxu. Pruža korisniku vrlo lako dovršavanje koda, inspekciju napisanog koda, isticanje pogrešaka u hodu te brze popravke. [3]

1.1 Značajke PyCharma

1. Inteligentni uređivač koda:
    - Olakšava nam pisanje visokokvalitetnih kodova
    - Označavanje sintakste sa bojama nam pomaže oko pisanja čitljivijeg koda
    - Pomaže nam u prepoznavanju pogrešaka
    - Omogućava nam značajku automatskog dovršavanja te upute za dovršavanje koda
2. Navigacija koda:
    - Pomoć pri uređivanju i poboljšanju koda uz manje truda i vremena

- Navigacija koda pomaže programeru doći do određene funkcije, klase ili datoteke
- Programer može locirati element, simbol u izvornom kodu u kratkom roku

3. Pomoć za mnoge druge web tehnologije:
    - Pomaže programerima u stvaranju web aplikacija u Pythonu
    - Podržava web tehnologije kao što su HTML, CSS, JavaScript
    - Mogućnost praćenja promjene izravno u web pregledniku
4. Pomoć za Python znanstvene knjižice:
    - PyCharm podržava Pythonove znanstvene biblioteke kao što su Matplotlib, NumPy i Anaconda
    - One pomažu pri izgradnji projekata znanosti o podacima i strojnog učenja
    - Interaktivni grafikoni od kojih se sastoji programerima pomažu razumjeti podatke [4]

1.2 Prednosti i nedostatci PyCharma

Neke od prednosti pisanja koda pomoću PyCharm okruženja su:

- **Povećana produktivnost**: PyCharmova inspekcija koda te njegove različite značajke uvelike pomažu pri bržem i ispravnijem pisanju koda
- **Smanjen broj pogreški:** Prije nego što uopće pokrenemo kod, inspekcija koda nam pomaže otkriti pogrešku u samom startu
- **Lakše otklanjanje pogrešaka**: Pomoću takvog programa za ispravljanje pogrešaka olakšava nam se pregled vrijednosti varijabli, postavljanje prijelomnih točaka i sl.

Dok neki od nedostataka Pycharm okruženja su:

- **Zahtijeva resurse**: Neki od izdanja PyCharma mogu uzrokovati sporije perfomanse na starijem ili na manje moćnom hardveru
- **Strma krivulja učenja**: Početnicima koji tek kreću s učenjem programiranja u Pythonu može trebati više vremena da se prilagode ovom okruženju
- **Sporije vrijeme pokretanja:** U usporedbi s laganim uređivačima tekstualnog sadržaja, PyCharm obično ima sporije vrijeme pokretanja [5]

2. Igra Tic-tac-toe

Igra Tic-Tac-Toe (hrv. Križić-Kružić) je jedna od najpoznatijih igara strategije i duhovitosti koja je osvojila umove i mladih i starih generacija. Početci igre zapisani su još kod hramova drevnog Egipta pa sve do srednjovjekovnih katedrala u Engleskoj. Takve rane izvedbe igre u Rimu, Egiptu i drugdje, razvile su temelje za razvoj igre kakvu danas svi rado igramo u bilo gdje i u bilo kojem trenutku.

Zanimljivo je kako se igra tic-tac-toe danas smatra uvelike dječjom igrom, dok u prošlosti nije bilo tako. Naime, povezivali su je s poganskim ritualima posvećenim magičnim svojstvima devet - kvadratne mreže. Ta rešetka bila je poznata kao Magični kvadrat (eng. Magic Square) jer bi zbroj brojeva od 1 do 9 koji su bili upisani davao uvijek isti zbroj vodoravno, okomito ili dijagonalno. Vjerovali su da takav Magični kvadrat nosi nekakvu numerološku poruku svijetu.

Originalni naziv igre je „tic-tat-toe“ te takav dolazi iz davnog 16. stoljeća. Tit sam po sebi znači šamar, dok tat označava odmazdu (kao tit for tat), a toe predstavlja treću figuru u nizu koja osigurava dobitnu kombinaciju za pobjedu. [6]

Jednostavnost igre i njezina pristupačnost učinile su ju omiljenom među djecom i odraslima, često je igrajući na kojekakvim komadićima papira ili kredom nacrtanim rešetkama na podu ulice u bilo kojem trenutku, bilo gdje.

Ploča igre tipično se sastoji od tri reda i tri stupca. Pravila igre su vrlo jednostavna, igrači odaberu znak s kojim žele igrati, naizmjenice jedan od igrača crta „X“ na prazno mjesto, drugi „O“. Pobjednik je onaj igrač koji stvori vodoravnu, okomitu ili dijagonalnu crtu sa svoja tri ista znaka. Ako dođe do popunjenja svih devet polja ploče odnosno do toga da ne postoji trojka u nizu, tada igra završava izjednačenjem.

Tic-tac-toe igra ne mora biti 3x3 dimenzija kao što je to uobičajeno. Postoje različite varijacije igre u kojoj ploča može biti dimenzija za različit N, odnosno ploča dimenzija NxN. Tada, sva pravila i dalje su ista, jedina razlika je što igra duže traje.

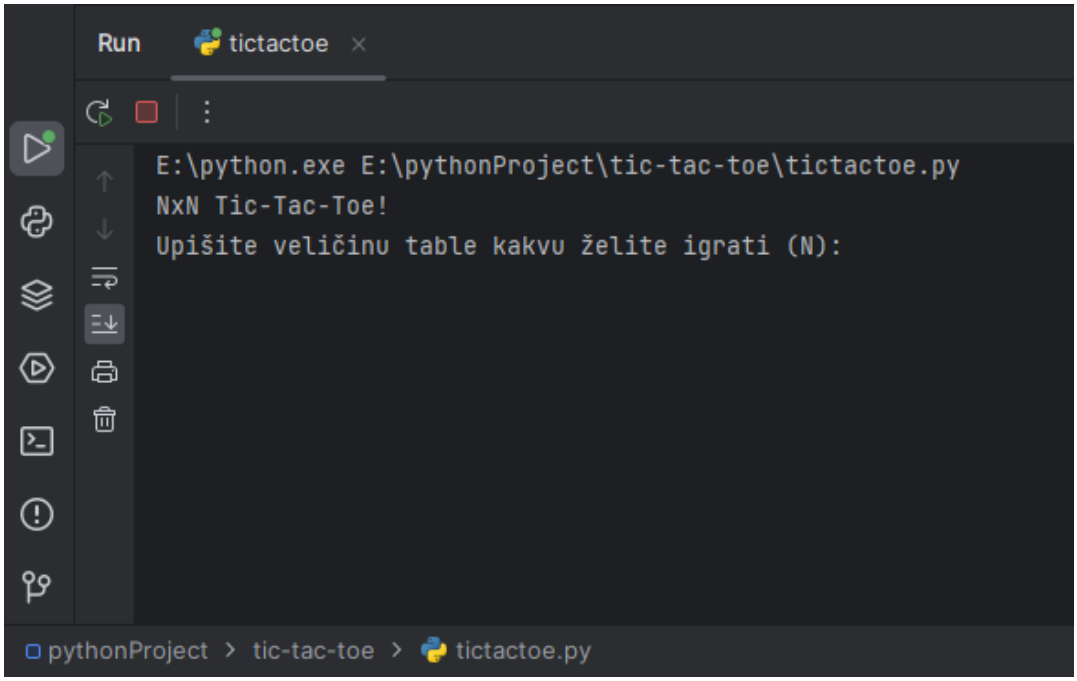
U ovom projektu je objašnjena izrada Tic-tac-toe igre koju korisnik igra protiv kompjutera uz dodatnu mogućnost promjene dimenzije ploče na samom početku igre.

Slika 1 Različiti načini igranja igre

Postoje razno razni kreativni načini za igranje Tic-tac-toe igre. Kao na primjeru na slici 1, koristeći stijene i par grančica, igra koja je inače osmišljena da se igra sa X i O oznakama, ovdje se igra sa stijenama koje su prebojane u obliku insekata. [7]

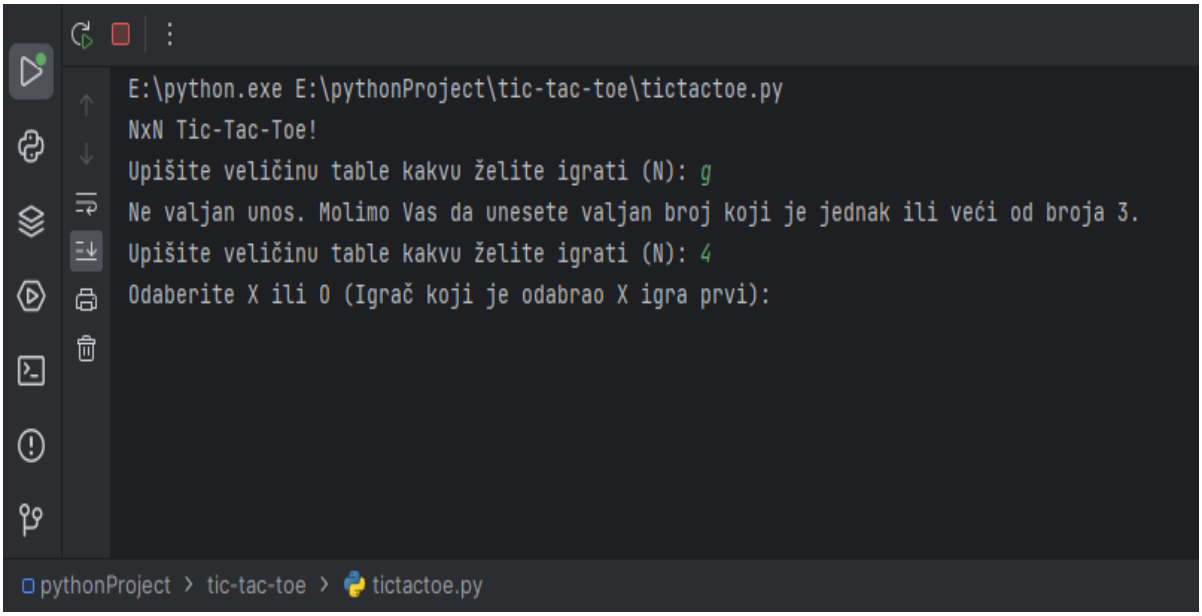
3. Implementacija igre Tic-tac-toe u Pythonu

U ovom projektu igru Tic-tac-toe može igrati samo jedan igrač protiv kompjutera s mogućnošću odabira veličine ploče. Nakon pokretanja aplikacije na ekranu nam se pojavljuje upit u kojem igrač ima opciju upisivanja veličine ploče na kolikoj želi igrati. Pri tome ukoliko upiše na primjer broj 3, tada će ploča za igranje igre biti veličine 3x3, ako upiše broj 4 tada će biti 4x4 itd. Ova opcija daje priliku igranja drugačije verzije kultne igre Tic-tac-toe koja je tipično dimenzija 3x3. U ovo polje nije moguće unijeti nijedan drugi tip podatka osim broja te nijedan broj koji je manji od broja 3. Upit nakon pokretanja igre, odnosno odabir veličine ploče na kojoj želimo igrati prikazan je na slici broj 2.



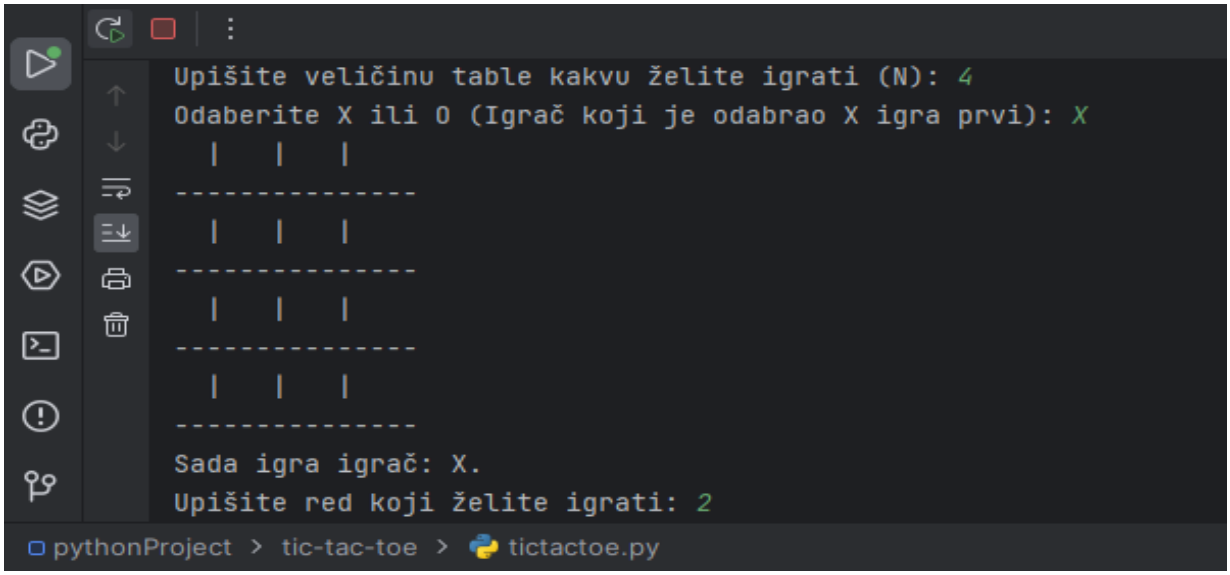
Slika 2 - Upit nakon pokretanja igre

Nakon što igrač odabere veličinu ploče na kojoj želi igrati protiv kompjutera, idući upit koji mu se pojavljuje na ekranu je vezan za odabir znaka s kojim želi igrati. Igrač ima mogućnost izabrati između znaka X ili znaka O. Ukoliko igrač upiše bilo koji drugi znak (slovo, broj...) različito od ponuđena dva znaka, aplikacija ponovno upućuje upit za odabir znaka sve dok igrač ne upiše točan podatak. Upit o odabiru znaka kojeg korisnik želi igrati prikazan je na slici broj 3.



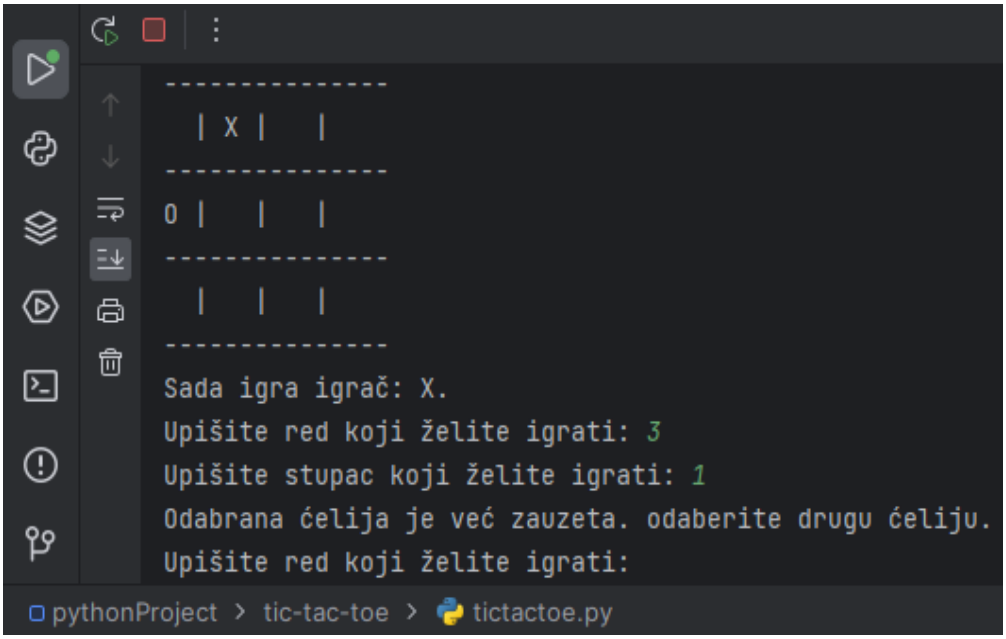
Slika 3 - Odabir znaka

Nakon odabira znaka, u ovom slučaju odabira znaka X. Igraču se prvo pojavljuje upit za unos reda kojeg želi odigrati, koji isto tako ne može biti nijednog drugog tipa podatka nego broj. Nakon odabira reda, postavlja mu se upit o odabiru stupca kojeg želi igrati. Nakon što korisnik odigra svoj potez, nakon njega kompjuter odigra svoj koji je naravno uvijek različit od poteza kojeg je korisnik odigrao. Prikaz ploče te upit o odabiru reda prikazan je na slici broj 4.



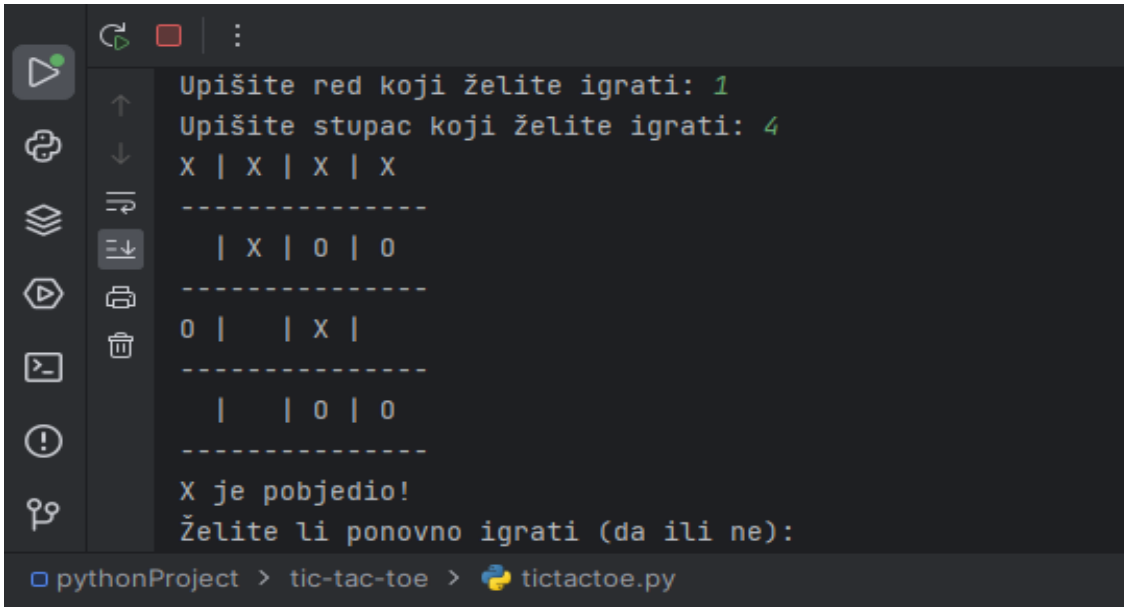
Slika 4 - Prikaz ploče te upit o odabiru reda

U slučaju odabira ćelije koja je već zauzeta, aplikacija nam javlja grešku te ponovno postavlja upit za odabir reda pa tako i stupca sve dok ne unesemo red i stupac koji već nisu zauzeti. Primjer pokušaja odabira zauzete ćelije prikazan je na slici broj 5.



Slika 5 - Primjer pokušaja odabira zauzete ćelije

Na ovakav način je realizirana Tic-tac-toe igra koja se odvija sve dok se sve ćelije ne popune pa rezultat ostane neriješen ili dok netko ne pobjedi. Ukoliko dođe do pobjede igrača ili pobjede kompjutera te ukoliko dođe do izjednačenja, aplikacija nam ispisuje rezultat te netom nakon nam postavlja upit o ponovnom igranju igre. Ukoliko želimo ponovno igrati, moramo upisati riječ „da“. Unos bilo koje druge riječi označava da ne želimo ponovno igrati, te tada igra završava. Prikaz kraja te upita o ponovnom igranju prikazan je na slici broj 6.



Slika 6 - Kraj ili ponovna igra

1.3 Modeliranje zahtjeva

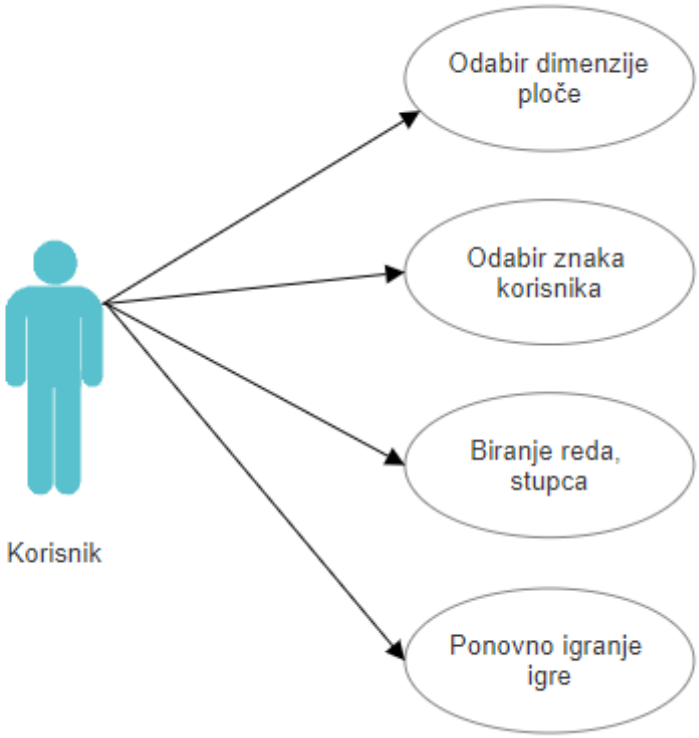
Za izradu modela zahtjeva koristiti ćemo SmartDraw alat (https://www.smartdraw.com/) , pomoću njega ćemo prikazati dijagram sa određenim funkcijama igre.

Glavna funkcionalnost igre je naravno samo igranje igre Tic-tac-toe koje je vrlo poznato i jednostavno za svakog pojedinca.

Neke od sporednih funkcionalnosti igre su:

- Odabir dimenzije ploče – korisnik sam odabire dimenziju ploče na kojoj želi igrati protiv kompjutera
- Odabir znaka – korisnik ima opciju biranja između dva znaka za igranje: X ili O
- Biranje reda i stupca – korisnik sam odabire red i stupac koji želi igrati
- Ponovno igranje igre – na kraju odigrane igre ponuđena je mogućnost ponovnog igranja

Nabrojene funkcije su pomoću nacrtanog dijagrama prikazane na slici 7.

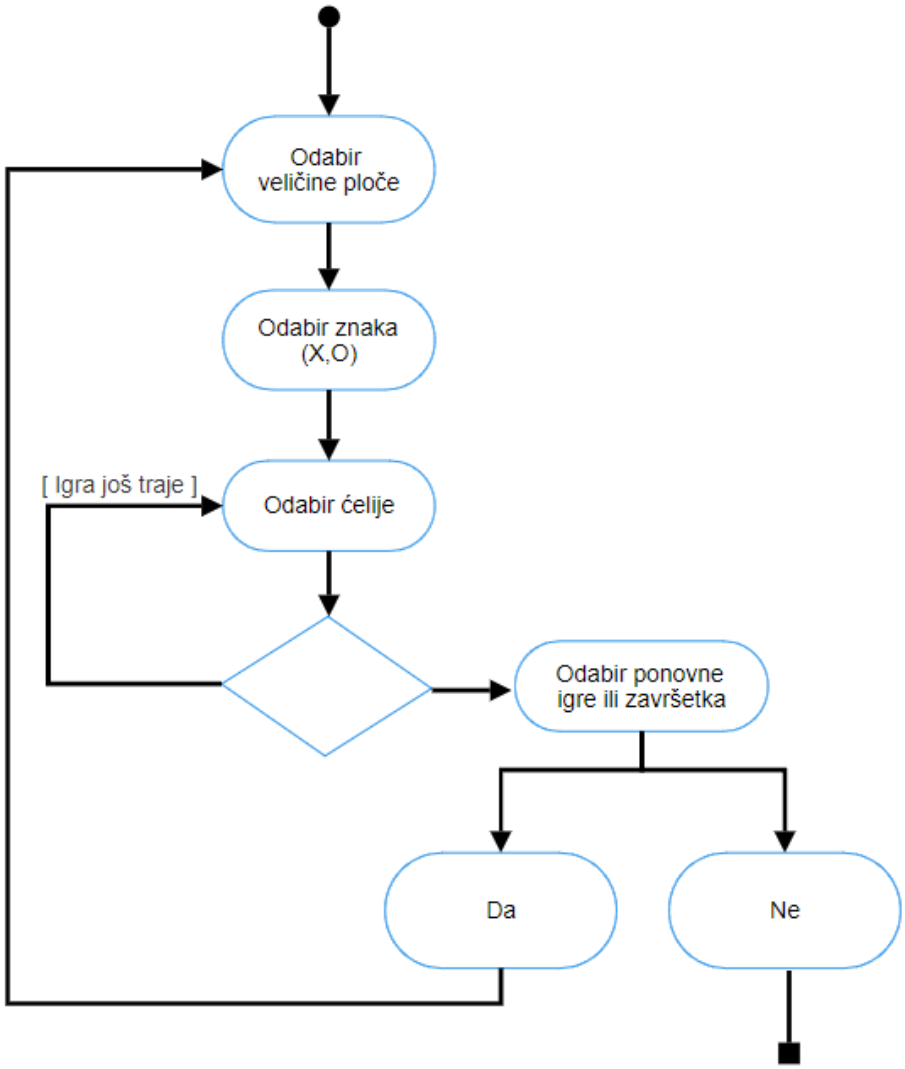


Slika 7 Dijagram - funkcija igre

1.4 Modeliranje tijeka

Nabrojene su sve funkcionalnosti ove aplikacije, od njezine glavne funkcionalnosti – igranja same igre Tic-tac-toe, do njezinih sporednih funkcionalnosti. U ovom odjeljku ćemo prikazati detaljniji opis svake određene funkcionalnosti ove igre uz pomoć dijagrama aktivnosti.

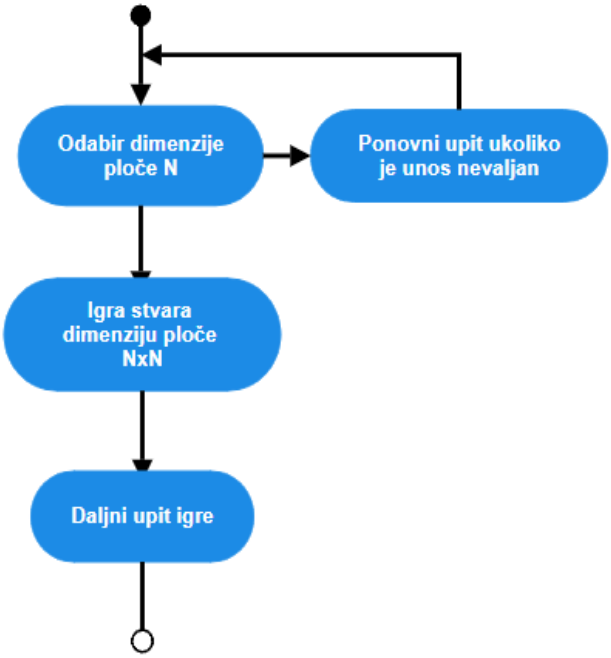
Vrlo jednostavan način igranja Tic-tac-toe igre odnosno njena glavna funkcionalnost prikazana je dijagramom aktivnosti na slici 8. Nakon što korisnik odabere koliku želi veličinu ploče i znak, korisnik igra igru protiv kompjutera sve dok igra ne bude završena.



Slika 8 - Dijagram aktivnosti igre Tic-tac-toe

1.4.1 Sporedna aktivnost: Odabir dimenzije ploče

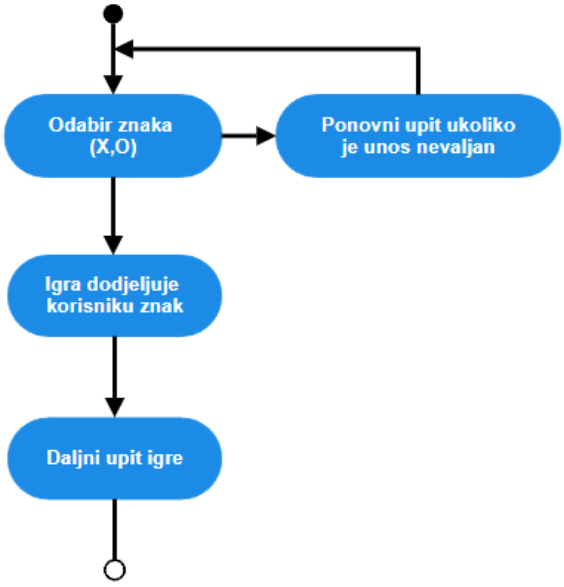
Ova sporedna aktivnost nam postavlja upit o odabiru veličine dimenzije ploče na kojoj želimo igrati protiv kompjutera. Kada igrač upiše željeni broj N, dimenzija ploče se postavi na NxN. Ukoliko korisnik upiše nevaljan unos, odnosno podatak koji nije broj, aplikacija ga upozorava te mu ponovno postavlja isti upit. Na slici 9 je prikazan dijagram ove aktivnosti.



Slika 9 - Dijagram aktivnosti odabira dimenzije ploče

1.4.2 Sporedna aktivnost: Odabir znaka

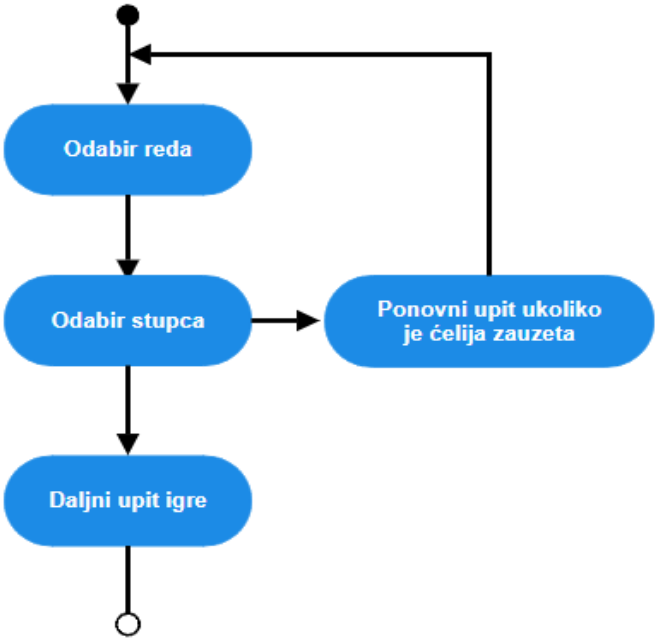
Nakon odabira dimenzije ploče korisnik ima mogućnost odabira znaka s kojim će igrati protiv kompjutera. Ima mogućnost odabira između X i O, ukoliko upiše znak drugačiji od tih dvaju ponuđenih znakova, aplikacija ponavlja upit. Sporedna aktivnost odabira znaka prikazana je dijagramom na slici broj 10.



Slika 10 - Dijagram aktivnosti odabira znaka

1.4.3 Sporedna aktivnost: Biranje reda i stupca

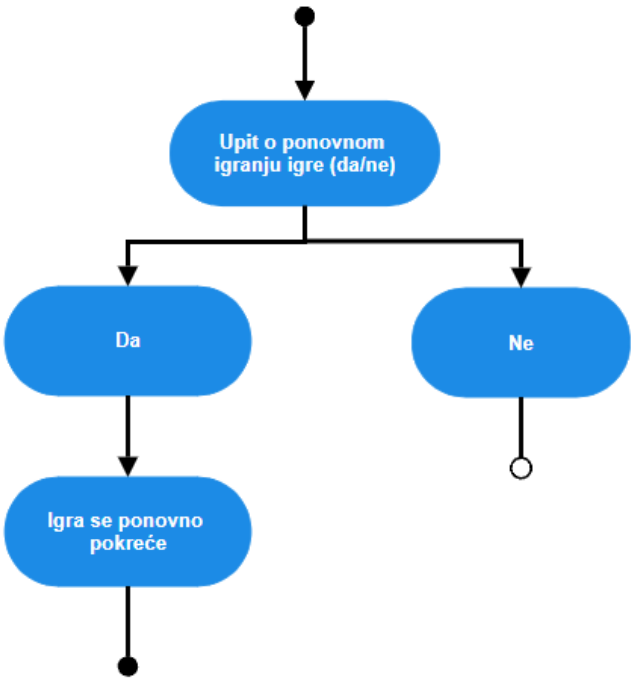
Ova sporedna aktivnost vezana je za odabir reda, pa tako i stupca kojeg korisnik želi igrati. S ovim odabirom korisnik zapravo bira ćeliju koju želi odigrati. Ukoliko je ćelija koju korisnik želi odigrati već zauzeta, javlja se upit o ponovnom unosu prvo reda pa onda i stupca. Na slici broj 11 prikazan je dijagram aktivnosti biranja reda i stupca.



Slika 11 - Dijagram aktivnosti odabira reda i stupca

1.4.4 Sporedna aktivnost: Ponovno pokretanje igre

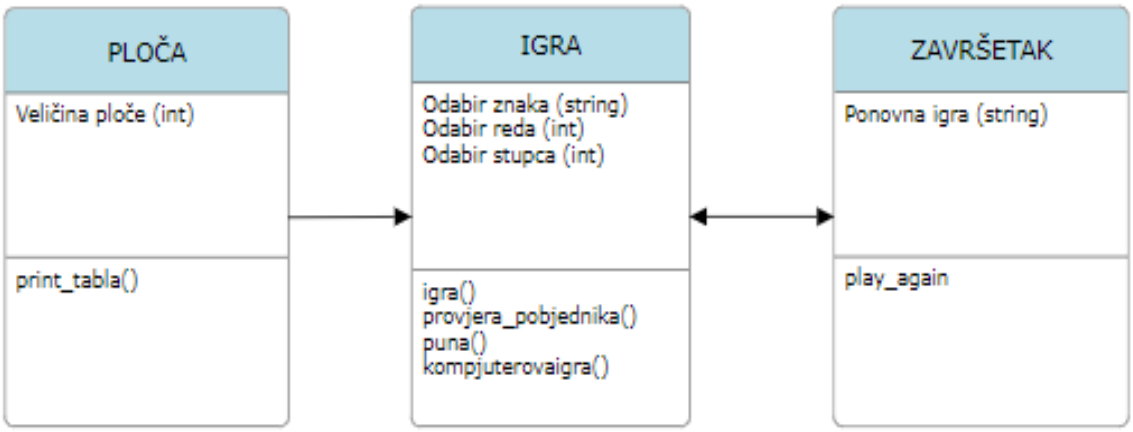
Nakon što se igra završi, igra nam pruža mogućnost ponovnog pokretanja igre (revanš) ili mogućnost prestanka igre. Ukoliko odaberemo ponovno igranje sa upisivanjem ključne riječi „da“, igra se ponovno pokreće. Inače, igra se završava. Ova aktivnost ponovnog pokretanja igre je prikazana dijagramom na slici broj 12.



Slika 12 - Dijagram aktivnosti ponovnog pokretanja igre

1.5 Logička struktura

Tic-tac-toe igra je realizirana uz pomoć raznih funkcija koje čine njezinu logičku strukturu. Svaka pojedina funkcija koja čini ovu igru funkcionalnom će biti detaljno objašnjena pomoću isječaka programskog koda u sljedećim potpoglavljima ovog rada. Na slici broj 13 prikazan je dijagram klasa igre.



Slika 13 - Dijagram klasa

1.5.1 Funkcija print_tabla

Ova funkcija vezana je za izradu ploče na kojoj će korisnik igrati protiv kompjutera. Širina ploče koja je označena sa „n“ služi nam kako bi lakše došli do željene dimenzije ploče na kojoj želimo igrati. Pomoću nje izraditi će se ploča uz pomoć oznaka „ | “ te „ - “. Kod ove funkcije vidljiv je na slici 14.

```
3   #Pritanje table
    6 usages
4   def print_tabla(tabla):
5
6       n = len(tabla)
7       for red in tabla:
8           print(" | ".join(red))
9           print("-" * (4 * n - 1))
10
```

Slika 14 - Funkcija print_tabla

1.5.2 Funkcija provjera_pobjednika

Slijedeća funkcija u kodu vezana je za provjeru pobjednika igre. U prvom dijelu koda provjerava se je li jedan red na ploči ispunjen sa istim znakom, odnosno je li došlo do pobjede na način da se popunio jedan red na ploči. Drugi dio koda provjerava je li možda stupac na ploči ispunjen sa istim znakom, odnosno je li na taj način došlo do pobjede. Isto tako u trećem dijelu koda se provjerava je li ijedna dijagonala na ploči ispunjena sa istim znakom. Ukoliko nijedna od ovih provjera ne vraća „True“, igra se nastavlja sve dok netko ne pobijedi ili ne dođe

do popunjenja svih ćelija na ploči, odnosno do izjednačenja. Kod spomenute funkcije vidljiv je na slici broj 15.

```
#Provjera tko je dobio igru
2 usages
def provjera_pobjednika(tabla, igrac):

    n = len(tabla)

    # Provjera redova
    for red in tabla:
        if all(celija == igrac for celija in red):
            return True

    # Provjera stupaca
    for stupac in range(n):
        if all(tabla[red][stupac] == igrac for red in range(n)):
            return True

    # Provjera dijagonala
    if all(tabla[i][i] == igrac for i in range(n)) or all(tabla[i][n-i-1] == igrac for i in range(n)):
        return True
    return False

```

Slika 15 - Funkcija provjera_pobjednika

1.5.3 Funkcija puna

Funkcija puna vezana je za provjeru popunjenja svih ćelija ploče. Ukoliko kroz ovu provjeru aplikacija naiđe na praznu ćeliju na ploči, igra će se nastaviti sve dok se sve ćelije ne popune sa znakovima X ili O. Spomenuta funkcija prikazana je na slici broj 16.

```
#Gledamo jesu li sve ćelije zauzete
2 usages
def puna(tabla):
    for red in tabla:
        if any(celija == " " for celija in red):
            return False
    return True

```

Slika 16 - Funkcija puna

1.5.4 Funkcija kompjuterovaigra

Da bih kompjuter mogao randomizirano igrati, napravljena je funkcija kompjuterovaigra koja je vezana za poteze koje igra kompjuter. Njegovi potezi su generirani pomoću metode random.choice() koja iz određenog raspona (moguca_igra) uzima jednu ćeliju na ploči te tu stavlja njegov znak ukoliko je ta ćelija slobodna za igranje. Prikaz funkcije nalazi se na slici 17.

```
38  #Kompjuter randomizirano igra
    1 usage
39  def kompjuterovaigra(tabla):
40      n = len(tabla)
41      moguca_igra = [(red, stupac) for red in range(n) for stupac in range(n) if tabla[red][stupac] == " "]
42      return random.choice(moguca_igra)
43
```

Slika 17 - Funkcija kompjuterovaigra

1.5.5 Funkcija igra

Ovu funkcija je radi lakše preglednosti prikazana u dijelovima. Prvi dio koda baziran je na samoj izradi ploče čiju veličinu korisnik sam odabire odgovorom na postavljeno pitanje o dimenziji ploče. Korisnik mora unijeti broj, ukoliko unese tip podatka koji nije broj ili unesen broj koji je manji od broja 3, igra obavještava o nevaljanom unosu te ponavlja svoj upit. Ovaj dio koda vezan za izradu ploče prikazan je na slici broj 18.

```
2 usages
def igra():
    print("NxN Tic-Tac-Toe!")

    while True:
        try:
            n = int(input("Upišite veličinu table kakvu želite igrati (N): "))
            if n < 3:
                print("Veličina table mora jednaka ili veća od broja 3.")
                continue
        except ValueError:
            print("Ne valjan unos. Molimo Vas da unesete valjan broj koji je jednak ili veći od broja 3.")
            continue
        else:
            break

    #Radimo tablicu da bi je mogli "punit" sa ili X ili O

    tabla = [[" " for _ in range(n)] for _ in range(n)]

```

Slika 18 - Izrada ploče

U ovom dijelu koda aplikacija postavlja upit korisniku o odabiru znaka. Ukoliko se upisani znak ne nalazi u rasponu ['X','O'] unos se smatra nevaljalim te se upit ponavlja. Radi lakše izrade igre postavljeno je da prvo kreće znak X, drugi O, ali ukoliko korisnik izabere O onda on naravno igra prvi. Prikaz koda vezan za odabir znaka prikazan je na slici broj 19.

```
    #Biramo zelimo li igrati sa X ili sa O

    while True:
        igrac = input("Odaberite X ili O (Igrač koji je odabrao X igra prvi): ").upper()
        if igrac not in ["X", "O"]:
            print("Ne valjan unos. Molimo Vas da upišete X ili O.")
        else:
            break

    #Radimo listu sa X i O. Defaultno je stavljeno da je X = [0] (prvi igrac),O = [1] (drugi igrac)

    igraci = ["X", "O"]

    #Ali mi to mozemo promjeniti ako odlučimo igrati sa O

    if igrac == "O":
        igraci = ["O", "X"]

```

Slika 19 - Odabir znaka

U ovom dijelu koda aplikacija postavlja upit korisniku koji red te koji stupac želi odigrati. Ukoliko broj reda kojeg je korisnik unio prelazi veličinu ploče, te ukoliko korisnik unese broj manji od 1, upit se ponavlja. Isto vrijedi i za odabir stupca kojeg želi odigrati. Ukoliko korisnik odabere ćeliju koja je već zauzeta aplikacija javlja pogrešku te postavlja ponovni upit. Dio koda vezan za odabir ćelije koju korisnik želi odigrati nalazi se na slici broj 20.

```
while True:

    # Igrač igra

    print_tabla(tabla)

    print(f"Sada igra igrač: {igraci[0]}.")
    while True:
        try:
            red = int(input("Upišite red koji želite igrati: "))
            stupac = int(input("Upišite stupac koji želite igrati: "))
            if red < 1 or red > n or stupac < 1 or stupac > n:
                print("Ne točan unos. Broj reda ili stupca ne smije biti veći od veličine table te ne smije biti manji od 1.")
                continue
            if tabla[red - 1][stupac - 1] != " ":
                print("Odabrana ćelija je već zauzeta. odaberite drugu ćeliju.")
                continue
        except ValueError:
            print("Nevaljan unos. Molimo Vas da unesete valjan broj.")
        else:
            break

    tabla[red - 1][stupac - 1] = igraci[0]
```

Slika 20 - Odabir ćelije

Slijedeći dio koda koristi funkciju provjere pobjednika da provjeri je li došlo do pobjede. Ukoliko je netko od igrača pobjedio ispisuje se poruka te se igra zaustavlja. Isto tako, pomoću funkcije puna provjerava se je li došlo do izjednačenja. Dio koda baziran na igru kompjutera nadalje je prikazan. Te isto kao i za korisnika provjera pobjednika ili izjednačenja se isto obavlja nakon odigrane igre kompjutera. Na slici 21 je prikazan ovaj dio koda koji je vezan za provjeru pobjednika igre.

```
105         #Gledamo tko je pobjedio ili je li izjednačeno
106
107         if provjera_pobjednika(tabla, igraci[0]):
108             print_tabla(tabla)
109             print(f"{igraci[0]} je pobjedio!")
110             break
111
112         if puna(tabla):
113             print_tabla(tabla)
114             print("Izjednačena igra!")
115             break
116
117         # Kompjuter igra
118
119         print_tabla(tabla)
120         print("Kompjuterov red.")
121
122         red, stupac = kompjuterovaigra(tabla)
123         tabla[red][stupac] = igraci[1]
124
125         if provjera_pobjednika(tabla, igraci[1]):
126             print_tabla(tabla)
127             print("Kompjuter je pobjedio!")
128             break
129
130         if puna(tabla):
131             print_tabla(tabla)
132             print("Izjednačena igra!")
133             break
134
```

Slika 21 - Provjera pobjednika igre

Ukoliko je došlo do izjednačenja ili pobjednika bilo kompjutera ili korisnika, aplikacija nam postavlja pitanje želimo li ponovno igrati ili ne. Odgovorom sa ključnom riječju „da“ igra se ponovno pokreće, a bilo kojim drugim odgovorom koji nije ta riječ igra se zaustavlja.

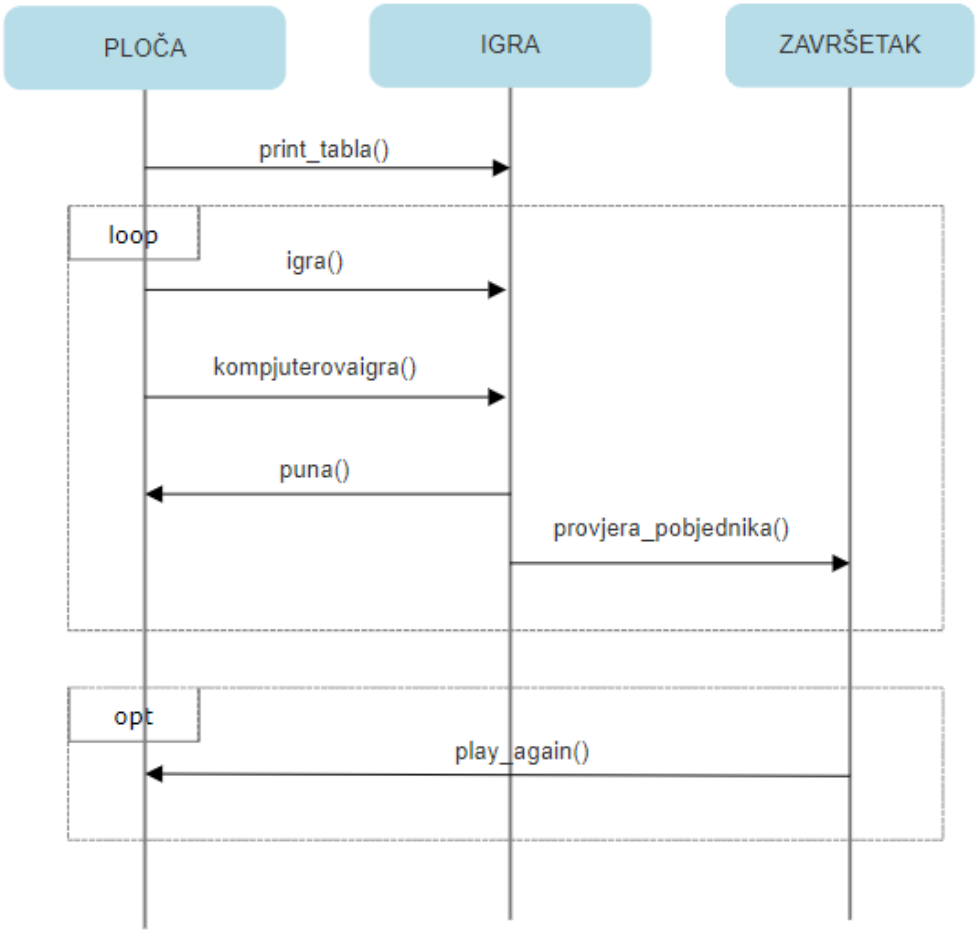
Isto tako, Donji i zadnji dio koda je uvjetna naredba koja govori pod kojim uvjetima glavna metoda treba biti izvršena. Ta naredba nam omogućava izvršavanje koda kada se datoteka izvodi kao skripta, ali ne kada je uvezena kao modul. Kod vezan za kraj ili početak nove igre nalazi se na slici broj 22.

```
135     #Kad završi igra odlučujemo hoćemo li igrati ponovno ili ne.
136
137     play_again = input("Želite li ponovno igrati (da ili ne): ").lower()
138     if play_again == "da":
139         igra()
140     else:
141         print("Hvala na igranju!")
142
143 if __name__ == "__main__":
144     igra()
145
```

Slika 22 - Kraj igre ili ponovna igra

1.5.6 Dijagram slijeda

Funkcije navedene u prethodnim potpoglavljima i redoslijed njihovih poziva mogu se prikazati dijagramom slijeda na slici broj 23.



Slika 23 - Dijagram slijeda

2 Zaključak

U svijetu u kojem se često čini kao da se kreće sve bržim tempom, Tic-tac-toe igra služi kao podsjetnik na ljepotu koja se nalazi u jednostavnosti. Prijateljsko natjecanje koje nastaje igrajući ovu igru bilo da se igra na tradicionalan način s olovkom i papirom ili na digitalnom ekranu je nešto što je doista posebno i nostalgično.

Za izradu aplikacije korišten je Python, jedan od temeljnih programskih jezika kojeg bi svaki početnik u programiranju morao znati. Međutim, da bih što jednostavnije kod bio napisan potrebno je bilo instalirati integrirano razvojno okruženje PyCharm koje služi za što lakše i čitljivije pisanje koda u programskom jeziku Pythonu.

Ovaj završni rad započinje sa kratkim opisom programskog jezika Python. U slijedećem dijelu rada opisan je PyCharm okruženje koje nam uvelike pomaže pri pisanju koda, zajedno sa njegovim specifikacijama, prednostima i nedostatcima. Kako bi lakše predočili aplikaciju koja je izrađena korištene su njezine snimke zaslona, kao i snimke zaslona samog koda u PyCharmu, svaka popraćena sa svojim kratkim opisom. Za još bolje razumijevanje koda aplikacija je još detaljnije pojašnjena i sa dijagramima aktivnosti za pojedine segmente koda.

3 Literatura

[1] Artdaily.com: https://artdaily.com/news/168505/A-Journey-Through-the-History-of-Tic-Tac-Toe

[2] Python (programming language): https://en.wikipedia.org/wiki/Python_(programming_language)

[3] PyCharm: https://en.wikipedia.org/wiki/PyCharm

[4] Intellipaat.com: https://intellipaat.com/blog/what-is-pycharm/

[5] Geeksforgeeks: https://www.geeksforgeeks.org/what-is-pycharm/

[6] Gamescrafters: http://gamescrafters.berkeley.edu/games.php?game=tictactoe

[7] Slika bubamare: https://sproutingwildones.com/tic-tac-toe-bugs/