  
Predmet:

CS324 - Skripting jezici  
  
Ime : Đurica Đuričić  
Broj indeksa: 2727  
Profesor: Igor Franc  
Mentor: Lazar Mrkela

1. Uvod

1.1 Zadatak projekta

Cilj ovog projekta je bio prikazivanje znanja stečenim na predmetu CS324. Kao tema projektnog zadatka je izabrana igra Tic-Tac-Toe ili Iks-Oks čija se pravila nalaze na <https://en.wikipedia.org/wiki/Tic-tac-toe> .  
  
1.2 O projektu

Projekat je rađen u Python programskom jeziku.

Pri izradi projekta se kao interpreter koristila Python 3.6.4 verzija za Windows operativni sistem.

Pri učenju Python sintaxe je korišćen PyCharm Edu verzija, dok je za izradu projekta korišćen PyCharm Professional razvojno okruženje.  
Dijagrami koji će se pojavljivati u dokumentu su rađeni alatu za dijagramovanje Lucidchart.

Dokumentacija ovog projekta je napisana u Word 2016.

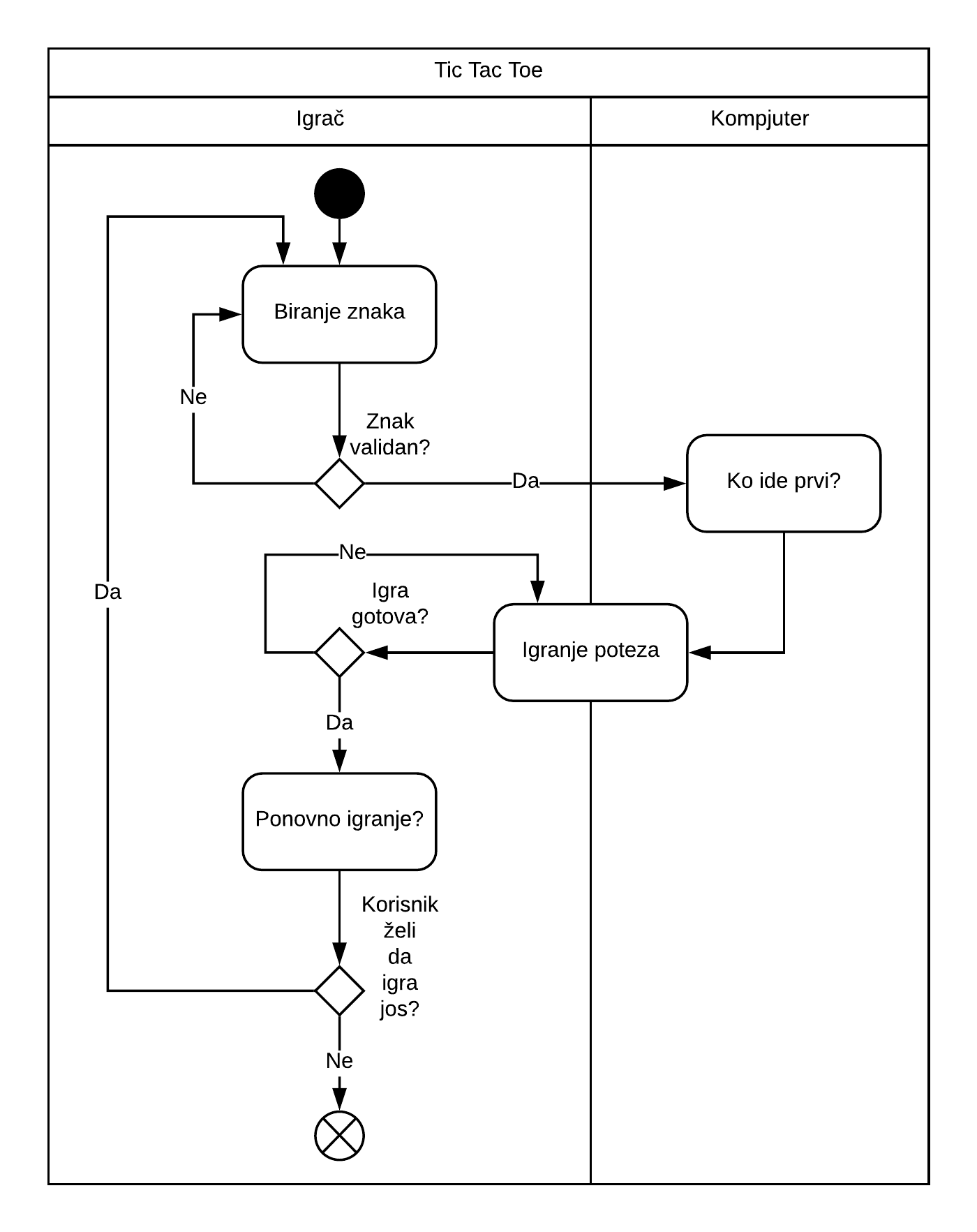
1.3 Reference

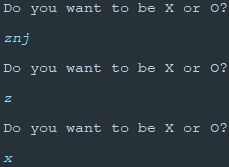
1. Prof. Igor Franc, Materijali za predavanja u elektronskom formatu iz predmeta CS324-Skripting jezici, Univerzitet Metropolitan, 2017/2018.

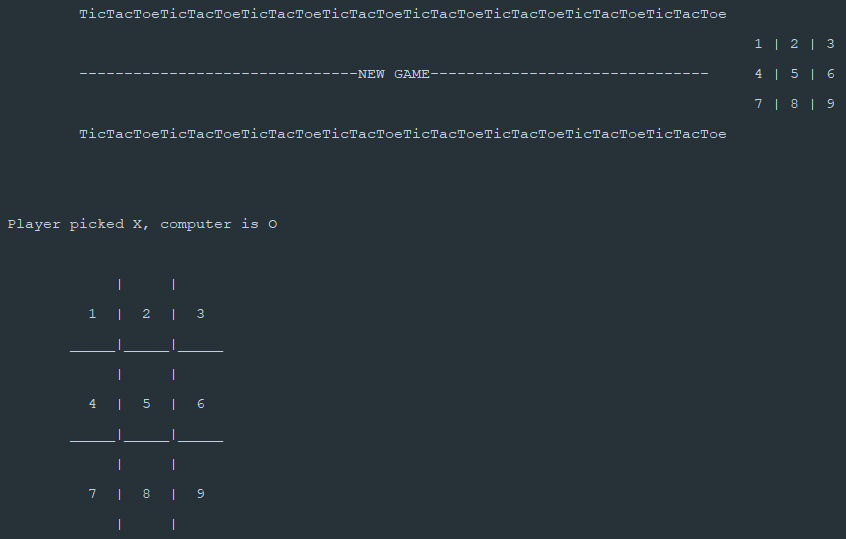
2. Python Software Foundation, Python 3.6 documentation, <https://docs.python.org/3/>

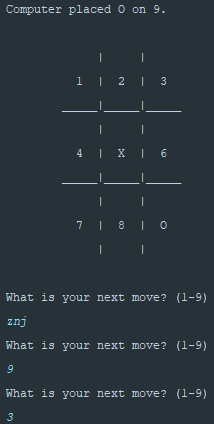
3.

2. Pokretanje projekta

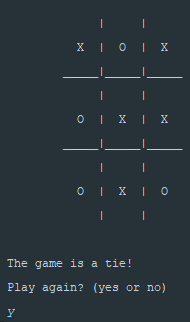
Na slici ispod možemo videti dijagram aktivnosti, sa kojeg možemo videti šta se u programu sve izvršava prilikom njegovog rada. Svaka od ovih aktivnosti, biće ispracena i slikom korisničkog interfejsa prilikom izvršavanja određene aktivnosti.  
  
Pri pokretanju igre, od korisnika se traži da izabere znak. Kao što možemo videti sa slike ispod, ova funkcionalnost se ponavlja sve dok korisnik u konzolu ne unese “X” ili “O”, bilo malim ili velikim slovima.



Ukoliko je znak validan, program određuje koji će korisnik biti prvi na potezu. Šanse za ovo su 50-50, jer se biranje vrši funckcijom random modula. Nakon biranja ko ide prvi, štampa se početak igre koji sadrži, koji igrač je izabrao koji znak, i štampanje početne “prazne” table, koja sadrži brojeve koji predstavljaju broj polja unošenja znaka.  
  


U oba slučaja, bilo da prvi igra igrač ili kompjuter, sledi igranje tog poteza. Na kasnijim dijagramima, videćemo da se ova aktivnost “igranje poteza”, sastoji iz više koraka I provera, I u slučaju igrača I u slučaju kompjutera.  
  


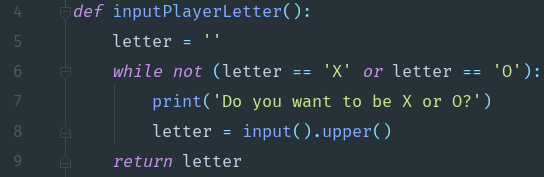
Sa slike vidimo, da nakon završenog poteza “protivnika”, program obaveštava igrača gde je kompjuter igrao, i štampa tablu apdejtovanu sa potezom protivnika. Nakon toga se od korisnika traži da unese svoj potez, i kao što vidimo, korisnik kao validan potez ne može uneti nepostojeće polje, niti polje koje je već zauzeto. Tabla se štampa nakon svakog validnog poteza.

Sa dijagrama aktivnosti smo videli, da se „završetak igre“, proverava nakon svakog poteza. Ova provera prvo podrazumeva proveru, da li je nerešetno, tj da li su sva polja popunjena, a zatim se vrši provera pobede igrača koji je igrao potez. U bilo kom slučaju, bilo da je pobedu odneo igrač, kompjuter, ili je pak izjednačeno, program pita igrača da li želi da igra ponovo.  
  
  
  
Na slici je prikaz konzole, u slučaju kada je igra nerešena.  
Kao što vidimo sa dijagrama aktivnosti, i ovog prikaza konzole, ukoliko se igrač odlučio za novu igru, uneće “yes” ili “y”, sadržaj konzole se čisti, i nova igra počinje, i sve aktivnosti sa dijagrama se vrše ispočetka. Ukoliko korinsik ne želi da igra više, nakon unosa njegove odluke, program se gasi.

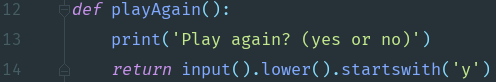
3. Rešenje zadatka

3.1 Main.py

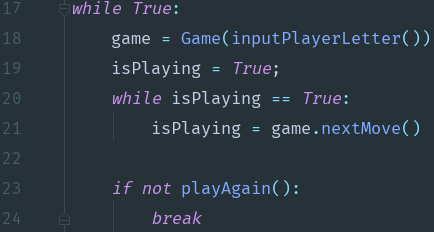
Prvi fajl, jednostavnih funkcionalnosti, koje ne zalaze toliko u samu logiku igre.  
https://i.gyazo.com/7d200fb87765c75b592f3a02018cf91b.png  
Ovaj fajl importuje sve iz našeg modula, klase Game, kako bi smo kasnije mogli napraviti instancu ove klase.

3.1.1 inputPlayerLetter()  


Funkcija koja je zadužena za validno unošenje znaka igrača. Vraća taj znak samo ukoliko je “X” ili “O” I takođe obezbedjuje da je uneti znak uppercase, zbog lepse prezentacije na kasnijim tablama igre.

3.1.2 playAgain()  


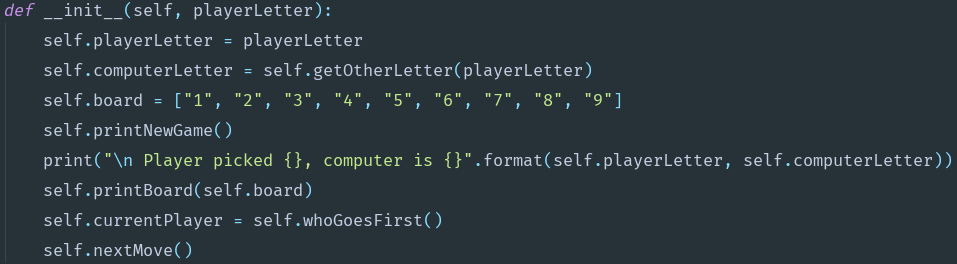
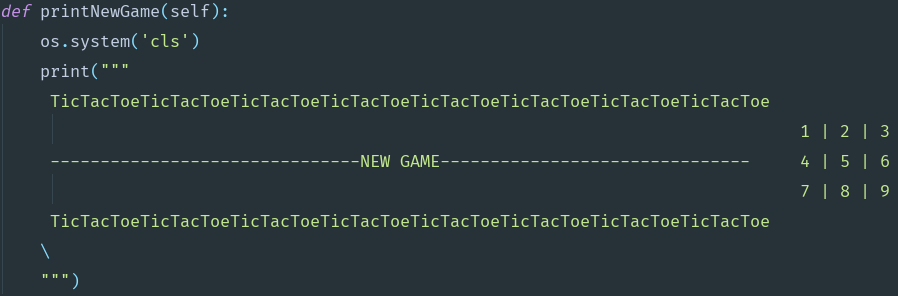
Jednostavna funckija izdvojena iz glavne petlje zbog preglednosti. Vraća True ukoliko je korisnik uneo “yes”, “y”…

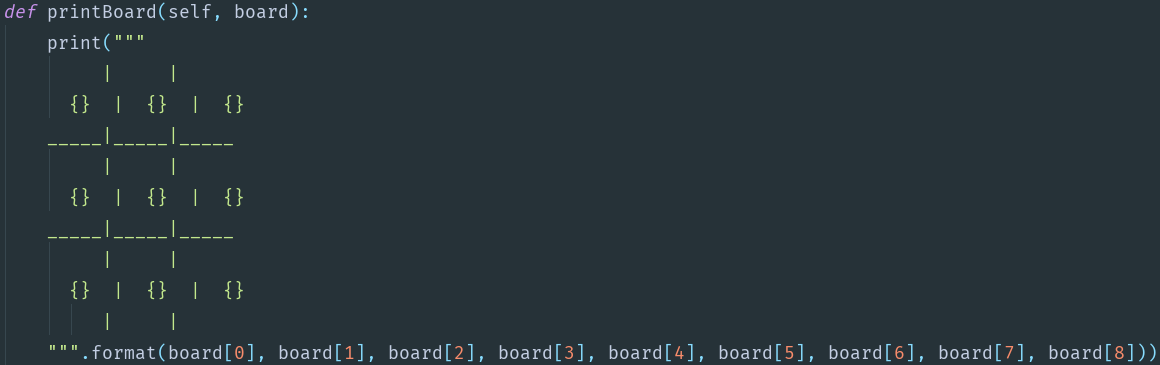
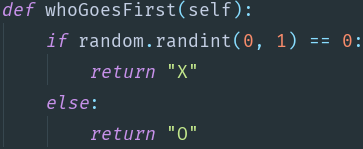
3.1.3 glavna While petlja  


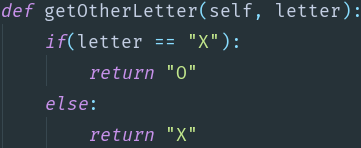
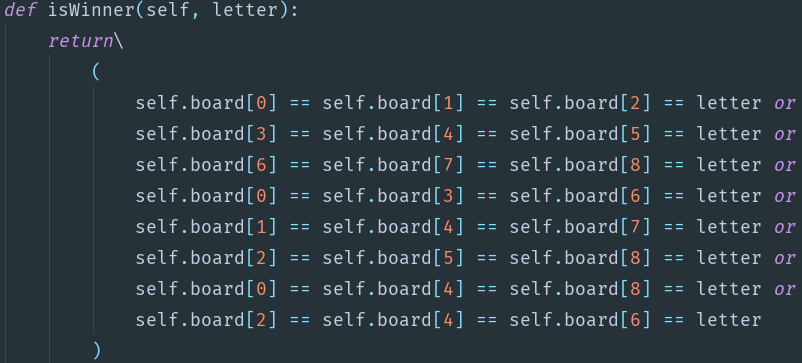
Petlja zadužena za “trajanje” igre. Kreira novu instancu igre, na kojoj se vrši igranje, sve dok funkcija nextMove() vraća true, koje signalizira da igra nije još završena. Ukoliko igrač želi igrati novu igru, kreira se nova instanca igre, a u suprotnom program se završava.

3.2 Game.py

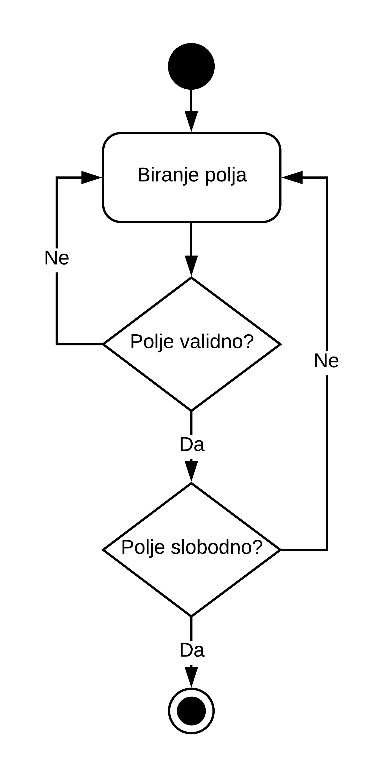
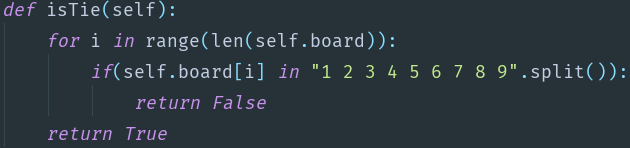
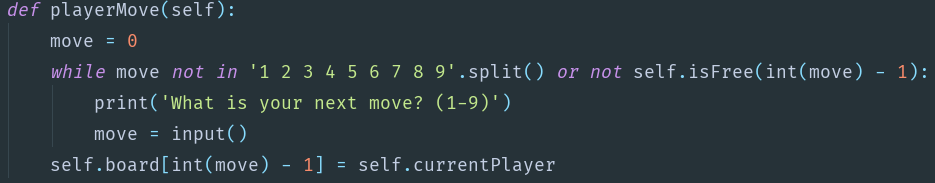
Game je glavni python file, klasa koja sadrži svu core logiku igre.  
https://i.gyazo.com/42574c52733f790c949685423d2caf9a.png  
Kao što vidimo, klasa koristi tri modula:  
- random module koji nam omogućuje biranje nasumične vrednosti u različitim slučajevima, i čije funkcije ćemo koristiti u više funkcionalnosti naše Game klase  
- os module, koji nam omogućuje korišćenje funkcija našeg operativnog sistema, i čiju funkcionalnost ćemo koristiti za brisanje konzole  
- time module, čije nam funkcionalnosti omogućuju manipulaciju s vremenom, što ćemo koristiti u jednoj od funkcija našeg sistema

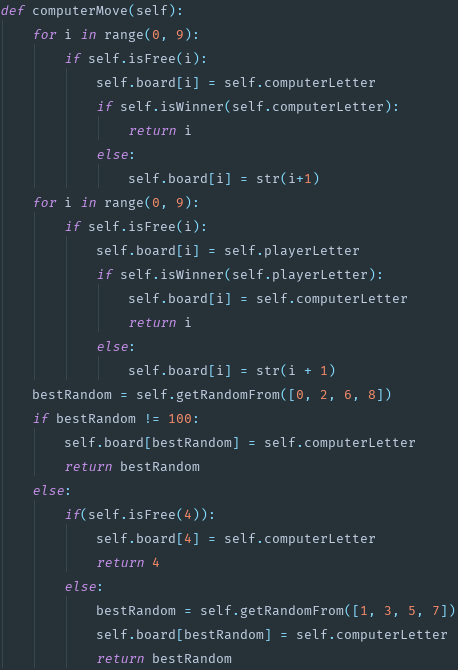
3.2.1 konstruktor klase (playerLetter)  
Funkcija koja se poziva prilikom kreiranja nove instance ove klase.  
  
U konstruktoru naše klase inicijalizujemo stvari neophone za izvršavanje naše aplikacije, kao što su određivanje znakova igrača, biranje igrača koji ide prvi, štampanje početne table za iks-oks, I naravno – započinjanje prvog poteza.  
  
3.2.2 printNewGame()  
  
Jednostavna funkcionalnost koja koristi funckiju operativnog sistema, kako bi prvo očistila ekran od predhodne igre, a zatim i ištampala “naslov”, koji označava početak nove igre.

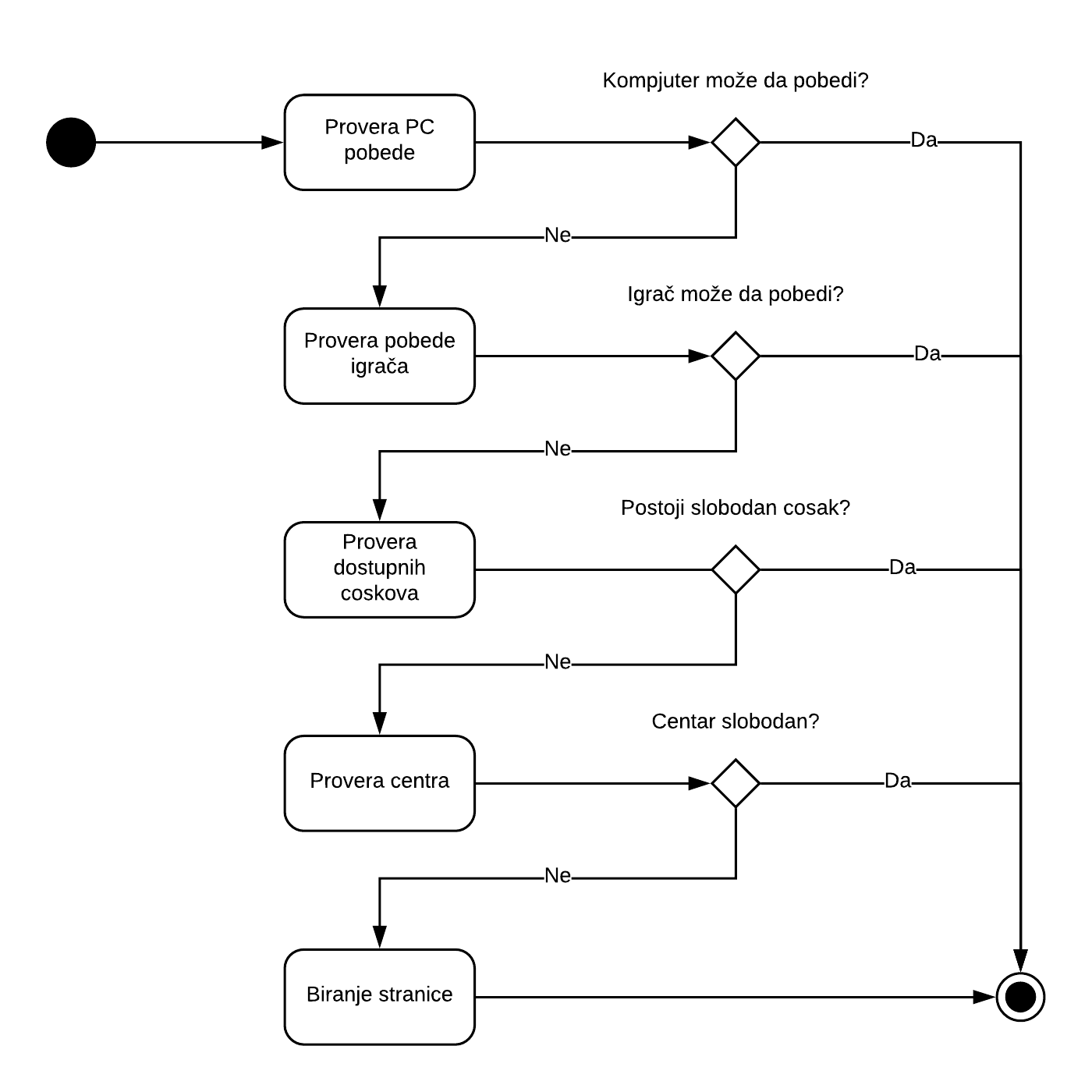
3.2.3 printBoard(board)  
  
Funkcija za štampanje table, koja se poziva na početku igre, i na kraju svakog validnog poteza. Koristi novu sintaksu formatiranja Stringova dodatu u Python 3 verziji, kako bi vrednosti prosleđene table ubacivala u String koji vizualizuje tablu.  
  
3.2.4 whoGoesFirst()  
    
Funkcija koja vraća “X” ili “O” nasumičnim izborom. Koristi funkciju randint spomenutog random modula, koja vraća vrednost od 0 ili 1, od koje kasnije zavisi vraćanje izabranog znaka.

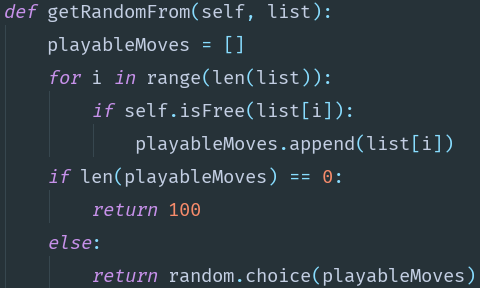
3.2.5 getOtherLetter(letter)  
  
Pomoćna funkcija koja je u nekoliko drugih funkcionalnosti skratila ovaj ponavljajući kod. Jednostavno vraća znak suportan od znaka koji joj se prosledi.  
  
3.2.6 isFree(move)  
https://i.gyazo.com/44923f69077f326c1d817658b1c0ddb4.png  
Funkcija koja vraća Boolean u zavisnosti da li je polje prosleđeno kao argument zauzeto, bilo znakom igrača ili znakom kompjutera.  
  
3.2.7 isWinner(letter)  
  
Funkcija koja vraća Boolean u zavisnosti od toga da li je bilo kojom kombinacijom pobedio igrač čiji je znak prosleđen kao argument ovoj funkciji.  
3.2.8 nextMove()  


Jedna od bitniih funkcionalnosti našeg programa. Kao što vidimo, prvo proverava da li je igra već završena, I ukoliko jeste, obaveštava igrača o tome I vraća False. Zatim, u zavisnosti od toga koji igrač trenutno ima potez, izvršava njegov potez, funkcijama playerMove() i computerMove() o kojima će kasnije biti reči. Možemo videti i funkciju sleep(t) modula time, gde t predstavlja broj sekundi na koje program treba da “spava” tj. prekine izvršavanje daljeg koda. Ova funkcija je u našem slučaju isključivo dekorativna, kako ona treba da predstavi vreme koje kompjuter “razmišlja” o svom potezu. Dalje, vidimo da ostatak funkcije proverava da li je igrač čiji je potez upravo odigran dobio igru. Ako je ovo slučaj, funkcija vraća False. Na kraju svakog poteza se naravno menja I igrač.  
Ukratko, ova funkcija odigrava potez ukoliko za njega ima mesta, a vraća False u Main.py ukoliko je igra gotova.

3.2.9 isTie()  
  
Malo pre pozivana funkcija, koja proverava da li je igra završena kao nerešena. Ovo radi time što proverava da li neko od polja sadrži “nepopunjeno” polje, i vraća false ako postoji barem jedno “slobodno” polje.  
  
3.2.10 playerMove()  
  
  
Funkcija koja prima input korisnički input. Proverava da li je polje postoji, i da li je polje slobodno. Ukoliko su oba zahteva ispunjena, vrednost unešenog polja se u “tabli” zamenjuje sa znakom igrača trenutnog igrača. S obzirom da se ova funkcija poziva samo kad igra igrač, moguće je zameniti self.currentPlayer sa self.playerLetter kako imaju istu vrednost.

3.2.11 computerMove()  
  
Funkcija koja nosi “core” logike igranja protiv kompjutera. Ovo je funkcija koja AI protivnika čini “pametnim” I biće pojednostavljena dijagramom i objašnjena na sledećoj stranici.

  
Dakle, program prvo prolazi kroz sva polja, i proverava da li sledećim potezom može završiti igru u svoju korist. Ukoliko može, to će i uraditi. Ukoliko ne može, kompjuter proverava da li može da spreči korisnika u dobijanju runde. Ukoliko može to će i uraditi. Ukoliko ne može, pokušaće da zauzme ćošak table, nasumično izabran od svih slobodnih ćoškova. Ukoliko su svi ćoškovi zauzeti, sledeća najpovoljnija pozicija je centar. Ukoliko je slobodan, kompjuter će odigrati taj potez. Ukoliko ni centar nije slobodan, kompjuter će izabrati nasumičnu “stanicu” od slobodnih “stranica”. Primetiti da kompjuter u ovom slučaju mora odigrati stranicu, jer je barem jedna sigurno slobodna. Razlog ovoga je što je nepostojanje slobodnih polja provereno pre pozivanja computerMove funkcije.

3.2.12 getRandomFrom()  
  
Funkcija koju smo koristili pri biranju nasumičnog ćoška ili stranica pri potezu kompjutera. Pozicije ćoškova ili stranica ovoj funkciji argumentom prosledjujemo kao listu. Funkcija prvo proverava koja polja od prosleđene liste su slobodna, i kreira listu sa njima. Ukoliko nema slobodnih pozicija, lista vraća neki int ( namerno ne vraća False – jer se false tumači isto kao 0, što je jedno od naših polja ). A ukoliko ima slobodnih polja, getRandomFrom vraća vrednost koju izabere funkcija choice, modula random, koja prima listu, i vraća nasumičnu vrednost iz te liste.

4. Zaključak  
Pri izradi ovog projekta naučeno je dosta. Iako se većina koda zasniva na jednostavnim algoritmima, u tim algoritmima je korišćeno mnogo stvari koje su specifične za Python. Od korišćenja raznih manipulacija Stringova i lista, do korišćenja postojećih Python 3 modula, koji nam olakšavaju obavljanje raznih funkcionalnosti. Ovim projektom je naučena većina osnova koje bi trebao da zna jedan Python programer.