**Izveštaj projekta “Domineering”**

**II faza**

Tim ELISEF:

Miodrag Janić 17139

Antonije Mitić 17806

Nemanja Stojadinović 17974

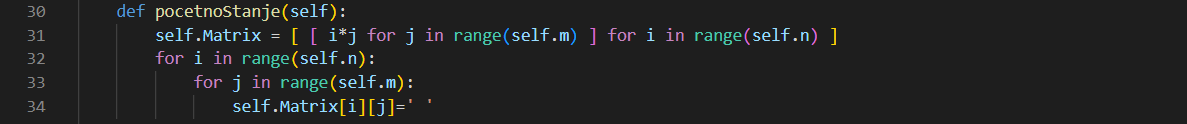
17.12.2022



Za potrebe definisanja stanja I svih ponašanja igre, definisana je klasa *Game* koja sadrži neophodne atribute koji omogućavaju pravilno funkcionisanje. Parametrima *n* I *m* definišemo dimenzije table, atribut *Matrix* predstavlja matricu koja je sama tabla, I atribut *kraj*, koji označava kraj igre.



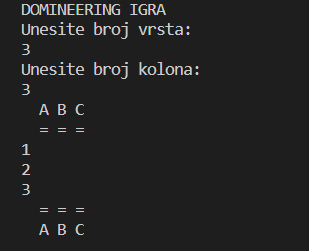
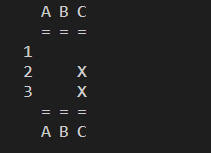
Funkcija *dimenzije* traži od korisnika da unese dva parametra preko konzole, kojima će se inicijalizovati parametri *n* I *m* instance klase *Game*.



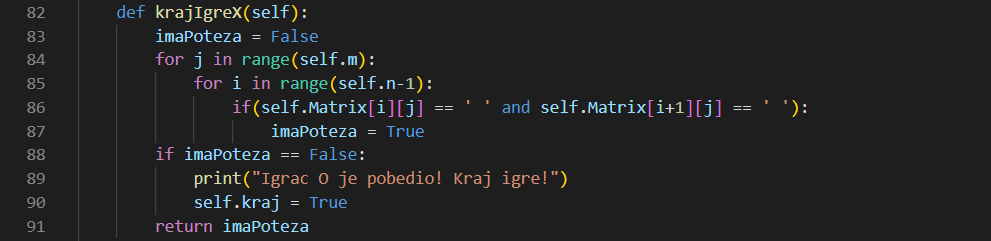
Funkcija *pocetnoStanje*, inicijalizuje matricu *Matrix*, koja je dimenzija *n* I *m*, blanko simbolima.



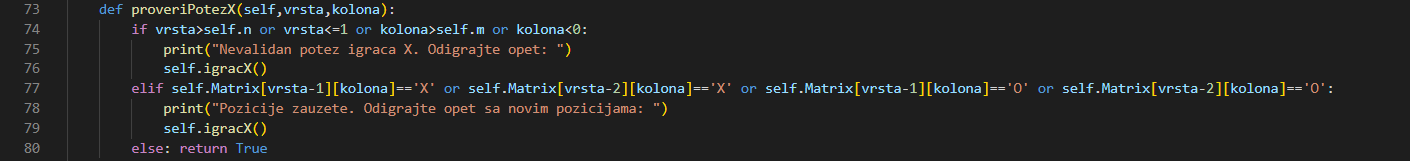
Funkcija *prikazTable* štampa tablu u bilo kom stanju (sa I bez pločica). Prvo se štampaju slova na vrhu table. Ona predstavljaju odredište kolone koje će unositi igrač u svakom svom potezu. Zatim se štampaju znaci jednako. Nakon toga, vertikalno se štampa niz brojeva koji će predstavljati odredište vrste koje će unositi igrač u svakom svom potezu. Zatim se štampa I sama tabla. Na kraju se štampaju, još po jednom, nizovi znakova jednako I slova, koji se nalaze na dnu table.

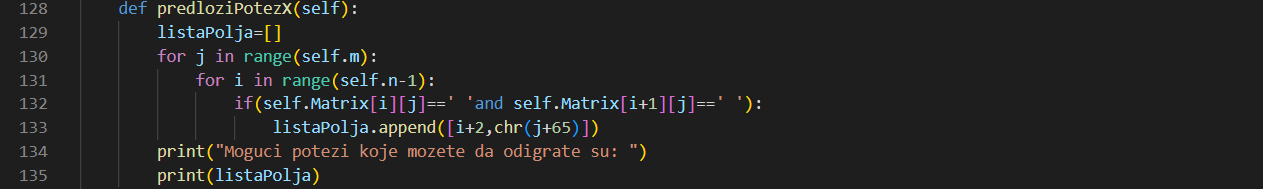
*Sl1. Unos početnih parametara I prikaz* *Sl2.Prikaz proizvoljnog stanje table*



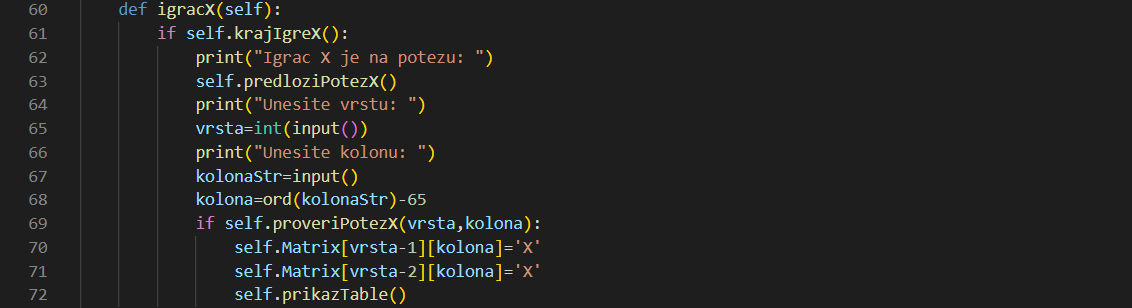
Funkcija *krajIgreX* proverava da li je došlo do kraja igre za igrača X. Parametar *imaPoteza* je na početku postavljen na *false*. Dvema petljama prolazimo kroz celu tablu, i ako postoje dva susedna polja matrice, koja su u istoj koloni (jer igrač X uvek postavlja pločice vertikalno), postavljamo parametar *imaPoteza* na *true*, sto označava da igrač X može da odigra potez. Ukoliko nema praznih polja, to znači da je igrač O pobedio, I atribut *kraj* se postavlja na *true*.



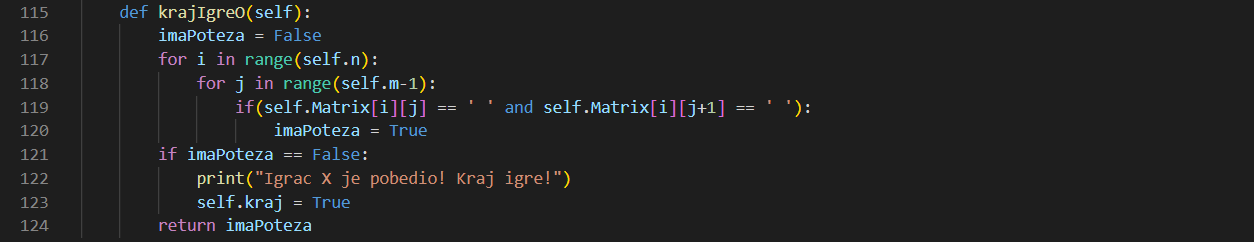
Funkcija *proveriPotezX* proverava da li pozicije koje je igrač X uneo preko konzole odgovaraju stanju table. Prvo se proverava da li su pozicije u okvirima dimenzija table (da pozicije nisu van table, ili takve da pločica viri sa table). Ako su pozicije loše, ispisuje se odgovarajuća poruka I poziva se funkcija koja traži da igrač X opet odigra. Zatim se proverava da li su polja koja odgovaraju zadatim pozicijama slobodna (poređenjem polja sa X I O). Ukoliko nisu, ispisuje se odgovarajuća poruka I poziva se funkcija koja traži da igrač X opet odigra. Ako je sve u redu, vraća se *true*.



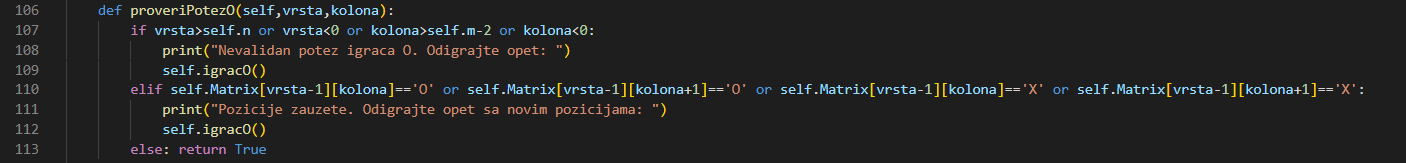
Funkcija *predloziPotezX* daje sve moguće poteze koje igrač X može da odigra u zavisnosti od trenutnog stanja na tabli. Deklarisana prazna lista će sadržati podliste sa oznakama vrsta i kolona na kojima igrač X može da stavi pločicu. Prolaskom kroz celu matricu pomoću dve petlje, traže se susedna polja koja su u istoj koloni i koja su prazna. Ako postoje, indeks (oznaka vrste i kolone) donjeg polja se dodaje u listu.



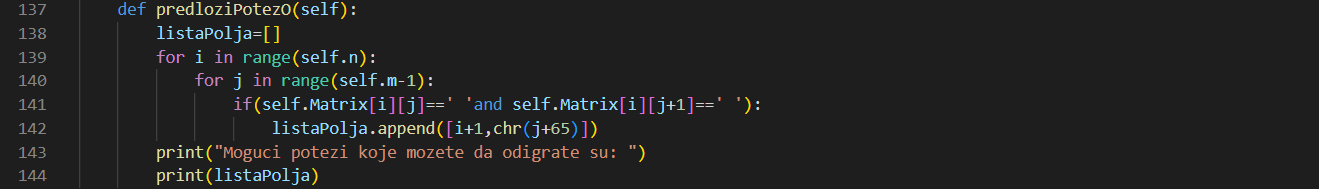
Funkcija *igracX* zadužena je za postavljanje pločice igrača X na tablu. Prvo se proverava da li uopšte igrač X može da odigra. Ako da, daje mu se predlog svih mogućih poteza. Preko konzole unosi pozicije na kojima želi da postavi pločicu. Nakon unosa pozicija (vrsta I kolona), ukoliko su validne, izvršiće se postavljanje pločice na odgovarajuće mesto na tabli. Nakon toga, prikazuje se tabla sa ažuriranim stanjem.



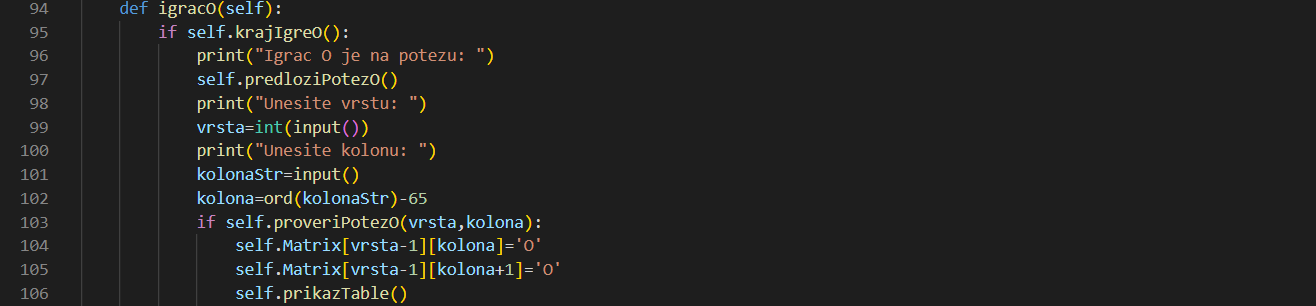
Funkcija *krajIgreO* proverava da li je došlo do kraja igre za igrača O. Parametar *imaPoteza* je na početku postavljen na *false*. Dvema petljama prolazimo kroz celu tablu, i ako postoje dva susedna polja matrice, koja su u istoj vrsti (jer igrač O uvek postavlja pločice horizontalno), postavljamo parametar *imaPoteza* na *true*, sto označava da igrač O može da odigra potez. Ukoliko nema praznih polja, to znači da je igrač X pobedio, I atribut *kraj* se postavlja na *true*.



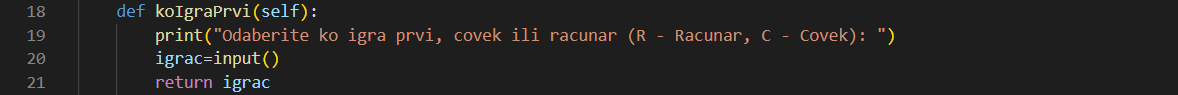
Funkcija *proveriPotezO* proverava da li pozicije koje je igrač O uneo preko konzole odgovaraju stanju table. Prvo se proverava da li su pozicije u okvirima dimenzija table (da pozicije nisu van table, ili takve da pločica viri sa table). Ako su pozicije loše, ispisuje se odgovarajuća poruka I poziva se funkcija koja traži da igrač O opet odigra. Zatim se proverava da li su polja koja odgovaraju zadatim pozicijama slobodna (poređenjem polja sa X I O). Ukoliko nisu, ispisuje se odgovarajuća poruka I poziva se funkcija koja traži da igrač O opet odigra. Ako je sve u redu, vraća se *true*.



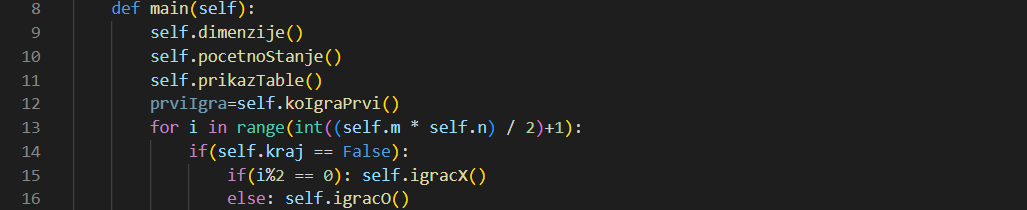
Funkcija *predloziPotezO* daje sve moguće poteze koje igrač O može da odigra u zavisnosti od trenutnog stanja na tabli. Deklarisana prazna lista će sadržati podliste sa oznakama vrsta i kolona na kojima igrač O može da stavi pločicu. Prolaskom kroz celu matricu pomoću dve petlje, traže se susedna polja koja su u istoj vrsti i koja su prazna. Ako postoje, indeks (oznaka vrste i kolone) levog polja se dodaje u listu.



Funkcija *igracO* zadužena je za postavljanje pločice igrača O na tablu. Prvo se proverava da li uopšte igrač O može da odigra. Ako da, daje mu se predlog svih mogućih poteza. Preko konzole unosi pozicije na kojima želi da postavi pločicu. Nakon unosa pozicija (vrsta I kolona), ukoliko su validne, izvršiće se postavljanje pločice na odgovarajuće mesto na tabli. Nakon toga, prikazuje se tabla sa ažuriranim stanjem.



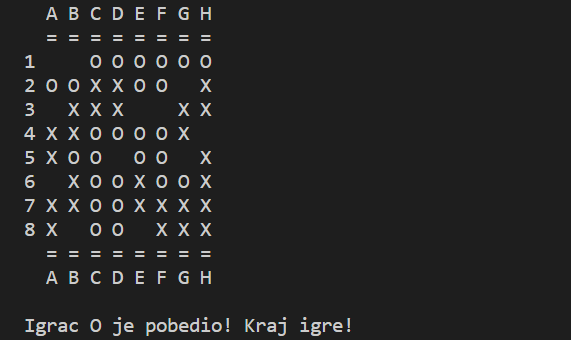
Funkcija *koIgraPrvi* traži od korisnika da unese da li želi da prvo igra čovek ili računar.



Funkcija *main*, koja predstavlja glavnu funkciju izvršenja igre. U njoj se pozivaju funkcije za unos dimenzija, inicijalizaciju I prikaz početnog stanja table. U petlji, koja se izvršava sve dok je moguće postavljati pločice na tablu, ukoliko nije došlo do kraja igre, pozivaće se funkcije koje koriste igrači X I O, naizmenično, s tim da igrač X igra prvi.



Instanciranje objekta klase *Game* I njegovo izvršenje.



*Sl3. Prikaz table nakon pobede igrača O*