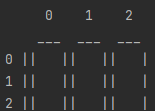
Izvestaj za drugu fazu projekta

Tablu predstavljamo kao listu koja u sebi sadrzi n listi, a svaka ta podlista sadrzi jos m elementa koji su tipa lista, a ta lista kao prvi element ima broj vrste, kao drugi element broj kolone, a kao treci element ima informaciju da li je polje zauzeto. Ako polje nije zauzeto ono ima vrednost None, a ako jeste onda ima ili vrednost ‘x’ ili ‘o’.

Ovo je primer jedne table dimenzija 3x3:



def izborIgraca():

Na pocetku pozivamo funkciju izborIgraca() gde imamo mogućnost da odlučimo da li hoćemo mi da izberemo koji igrač je prvi na potezu ili želimo da to bude igrač x. Postoji promenljiva igrač koja pamti koji igrač je na potezu, ako ona ima vrednost 1 onda je na potezu ‘x’, a ako ima vrednost 2 onda je na potezu ‘o’.

def unesiVelicinuTable(pocetniIgrac):

Nakon unosa igraca pozivamo funkciju unesiVelicinuTable, koja zahteva unos dimenzija table.

def napraviTablu(board, brojVrsta, brojKolona)

Zatim se poziva funkcija napraviTablu kojom se definise pocetno stanje table, tj. postavlajju se sva polja u tabli na None. U ovoj funkciji pozivamo takodje i funkciju prikaziTrenutnoStanjeTable koja prikazuje popunjenost polja na tabli. Prikaz je kao na prvoj slici gore, u konzoli.

def prikaziTrenutnoStanjeTable(board, brojVrsta, brojKolona)

U ovoj funkciji se ispisuje trenutno stanje na tabli u konzoli u formatiranom obliku. Kao argumente fukciji se prosledjuju trenutno stanje table,broj vrsta i broj kolona.

def ZapocniIgru(board, igrac, brojVrsta, brojKolona, krajIgre)

Kao argumente ovoj funkciji prosledjujemo tablu,igraca koji je na potezu,broj vrsta i broj kolona. Za pocetak unose se koordinate polja na koja igrac zeli da odigra potez, zatim se poziva funkcija odigrajPotez, kojom se odigrava zeljeni potez. Ukoliko smo dosli do kraja igre ispisuje se poruka korisniku “Stigli ste do kraja igre”.

def odigrajPotez(board, potez, igrac, brojVrsta, brojKolona, krajIgre)

Argumenti ove funkcije su trenutno stanje table,potez koji igrac zeli da odigra,igraca koji je na potezu,broj vrsta,broj kolona i provera da Ii je doslo do kraja igre. Funkcija proverava da li je potez koji igrac zeli da odigra validan, ukoliko jeste odigrava se potez I prikazuje se trenutno stanje table,nakon kog je sledeci igrac na potezu. Nakon toga proveravamo da li smo stigli do kraja igre, ukoliko trenutni igrac moze da odigra potez poziva se funkcija ZapocniIgru. Ukoliko potez nije validan, u konzoli se ispisuje “NE MOZETE NA OVOJ POZICIJI DA STAVITE PLOCICU”. Nakon toga se prikazuje trenutno stanje table i zahteva se od koriisnika da odigra novi potez.

def proveraKrajaIgre(board,igrac, brojVrsta, brojKolona)

Parametri ove funkcije su stanje table,igrac koji je na potezu I dimenzije table. Za igraca koji je na potezu proverava se da li postoji polje na koje moze da odigra potez, ukoliko ne postoji protivnik je pobedio, u suprotnom igrac je u mogucnosti da odigra zeljeni potez. Kraj igre se ispituje tako što se za igrača ‘x’ traže dva horizontalna polja koja su susedna i imaju vrednost None, a za igraca ‘o’ se traže dva susedna vertikalna polja koja imaju vrednost None.

def validno(board, potez, igrac, brojVrsta, brojKolona)

Kao parametre ove funkcije prosledjujemo trenutno stanje table,potez,igraca koji je na potezu I dimenzije table. Proverava se da li igrac moze da odigra zeljeni potez. Prvo ispitujemo da li se potez koji smo prosledili uklapa u dimenzije naše table, a onda ako se uklapa ispitujemo da li su ta polja slobodna.

**Ovde su sada dodatne funkcije koje su dodate u fazi dva:**

def napraviNovoStanjeTable(board, igrac, potez,brojVrsta, brojKolona, prethodna):

Pomocu ove funkcije mi kreiramo novo stanje table tako što ako je promenljiva prethodna jednaka True onda se pamti prethodno stanje table i mi tada ovu funkciju koristimo da odigramo potez, a kada je promenljiva prethodna jednaka False onda ona sluzi za kreiranje svih mogucih poteza koje igrac moze da odigra.

def prikaziPrethodnaStanja(prdethodnaStanja, brojVrsta, brojKolona):

Ova funkcija služi da se prikažu prethodna stanja koja su odigrana od pocetka partije.

def sviMoguciPoteziIgraca(igrac, board, brVrsta, brKolona):

Ova funkcija prolazi kroz čitavu tablu i proverava za svako polje da li je validno i ako je validno onda kreira generiše novu tablu sa tim mogućim potezom, zatim tu tablu pamti u listi svaMogucaStanja koja sadrzi kao prvi element tuple koji predstavlja broj vrste i broj kolone poteza, a kao drugi element pamti ovo novo stanje table.o

**Funkcije koje su dodate u projektu za trecu fazu:**

def minmax\_alpha\_beta(board, depth, igrac, brojVrsta, brojKolona, alpha=(None,-math.inf), beta=(None,math.inf)):

Kao argumente ovoj funkciji prosledjujemo trenutno stanje table, dubinu, igraca koji je na potezu, dimenzije table, alpha i beta. Ukoliko je igrac 1 na potezu poziva se funkcija *max\_value*, a ukoliko je igrac 2 poziva se *min\_value*. Ovo je rekurzivna funkcija i ova rekurzija se završava kada je unutar samog algoritma dubina jednaka 0 ili je došlo do kraja igre.

def max\_value(board, depth, igrac, brojVrsta, brojKolona, alpha, beta, potez=None):

Parametri u ovoj funkciji su trenutno stanje table, dubina, igrac koji je na potezu, dimenzije table, alpha, beta i potez. Prvo se generisu svi moguci potezi igraca, zatim se u if delu proverava da li je dubina 0 ili nema mogucih poteza koje igrac moze da odigra ili se doslo do kraja igre, i ukoliko je jedan od tih uslova ispunjen funkcija vraca potez i heuristiku tog poteza. Ukoliko ni jedan od uslova nije ispunjen, prolazimo kroz sve moguce poteze koje igrac moze da odigra i odredjujemo koji od tih poteza ide najvise u korist trunutnog igraca. Za svaki potez se poziva fukcija *min\_value* i odredjuje se nova vrednost za alpha kao maksimum od trenutne vrednosti za alpha i vrednosti koju je funkcija min\_value vratila.

def min\_value(board, depth, igrac, brojVrsta, brojKolona, alpha, beta, potez=None):

Parametri u ovoj funkciji su trenutno stanje table,dubina, igrac koji je na potezu, dimenzije table, alpha,beta i potez. Prvo se generisu svi moguci potezi igraca, zatim se u if delu proverava da li je dubina 0 ili nema mogucih poteza koje igrac moze da odigra ili se doslo do kraja igre, i ukoliko je jedan od tih uslova ispunjen funkcija vraca potez i heuristiku tog poteza. Ukoliko ni jedan od uslova nije ispunjen, prolazimo kroz sve moguce poteze koje igrac moze da odigra i odredjujemo koji od tih poteza ide najvise u korist trunutnog igraca. Za svaki potez se poziva fukcija *max\_value* i odredjuje se nova vrednost za alpha kao minimum od trenutne vrednosti za alpha i vrednosti koju je funkcija max\_value vratila.

def oceni(stanje, brojVrsta, brojKolona, igrac):

Ova funcija izracunava heuristiku. Heuristika se odredjuje na osnovu razlike broja mogucih poteza koji igraci mogu da odigraju i na osnovu broja vec odigranih poteza. U zavisnosti od igraca vraca se pozitivna vrednost ili negativna vrednost.

Postoje neke male izmene u kodu kada se određuje koji igrač je na potezu i kod još nekih funkcija od prošle faze.