Predmet: Veštačka inteligencija
Projekat: Havana
Tim: 7A3
Faza: II

Uvodna napomena - modifikacije u fazi I:

- Indeksi vrste matrice u memoriji su promenjeni sa slova na brojeve. Konverzija će biti vršena samo pri unosu polja i pri štampanju matrice.
- Promenljiva *playground* je do sada sadržala samo 1 matricu koja je predstavljala trenutno stanje. U fazi II je to promenjeno u listu stanja zbog zahteva projekta.
- Promenljiva *playground* je dostupna globalno, pa nije bilo potrebe prosleđivati je kao parametar funkcije. To je promenjeno za potrebe novokreiranih funkcija radi izbegavanja redundanse koda.

Opis funkcija:

Na osnovu trenutne situacije na tabli i igrača na potezu formira se lista mogućih poteza.

- ***generatePossibleStates (state)***
  Prosleđuje se trenutno stanje na tabli, a zatim se za svaki red generišu mogući potezi koristeći funkciju *generateRowStates.* Povratne vrednosti pomenute funkcije se sada nadovezuju i dobija se lista mogućih poteza.

- ***generateRowStates (rowIndex row matrix)***
  Prosleđeni su indeks vrste koja se obrađuje, sama vrsta i trenutno stanje na tabli. Vrši se provera za svaki element u vrsti da li je moguće na njemu odigrati potez i ukoliko jeste taj potez se nadovezuje kao jedna od mogućnosti. U suprotnom, to polje se preskače.

Funkcije zadužene za proveru *mosta* i *vile*:

- ***dfs (waiting visited matrix)***
  Funkcija koja radi obilazak table po dubini od koordinata poslednje odigranog poteza. Povratna vrednost je tačnost kraja igre. Ta provera se nalazi na početku same funkcije i implementirana je u funkcijama *checkBridge* i *checkFork*. Da li je igra završena vrši se proverom globalne promenljive *isEnd*.

- ***getCorners ()***
  Pre početka igre pribavlja listu uglova table.

- ***checkBridge (move)***
  Za svaki potez se proverava da li je u listi uglova table. Ukoliko jeste inkrementira se globalna promenljiva *visitedCorners.* U trenutku kada ova promenljiva, čija je inicijalna vrednost na nuli, dostigne vrednost dva, igra je završena i dva ugla na tabli su povezana.
  Po obavljenoj pretrazi, vrednost brojača se resetuje.

- ***getSides ()***
  Pre svake pretrage kraja igre pribavlja listu strana table. Struktura strane table je lista cvorova koji se nalaze na samoj strani.

- ***checkFork (move)***
  Za svaki potez se ispituje da li se nalazi na nekoj od strana table. Ukoliko to jeste slučaj ta cela strana se izbacuje iz liste strana. Ovo je učinjeno da ne bi dva čvora koji leže na istoj strani bila računata kao dve povezane strane. U trenutku kada je broj preostalih strana dostigao tri, to znači da su druge tri strane uspešno povezane i postignut je kraj igre.

- ***removeFromSides (element sides)***
  Vrši se provera postojanja prosleđenog elementa u listi strana table. To se obavlja funkcijom *checkExistance*. Ukoliko ovo jeste slučaj cela strana, sa svim svojim elementima, se briše iz liste strana.

- ***checkExistance (element list)***
  Obavlja sličnu funkcionalnost kao i ugrađena funkcija *member* sem što za element koji se vrši provera može imati i listu.

- ***resetField (rowIndex columnIndex matrix)***
  Za prosleđeni index vrste se vrši pretraga u tabeli. Tako nađena vrsta se prosleđuje funkciji *resetRow* koja radi resetovanje vrednosti na prosledjenim koordinatama postavljajući vrednost na prazno polje, odnosno "-".

- ***resetRow (columnIndex row)***
  Ponašanje ove funkcije je opisano u prethodnoj.

Funkcije zadužene za proveru prstena:

Ispitivanje table radi utvrđivanja postojanja prstena je bio kompleksniji problem od rešavanja slučajeva mosta i vile.
Zato je nakon istraživanja na internetu bila implementirana ideja iz navedenog naučnog rada: https://project.dke.maastrichtuniversity.nl/games/files/bsc/bscHavannah.pdf
Ideja je sledeća: Vrši se pretraga potencijalnog prstena od poslednje odiganog poteza. Ukoliko se u toku pretrage naiđe na čvor koji ima manje od dva suseda, ili ima tačno dva suseda, ali su oni

međusobno takođe susedi ovaj čvor se briše iz konekcije. Na kraju pretrage ukoliko je u konekciji ostalo barem šest čvorova, smatra se da je prsten pronađen.

- ***dfsRing (waiting visited connectionNodes matrix)***
  Vrši se provera gorepomenutih uslova za čvor koji se trenutno obilazi. Ukoliko je neki od njih ispunjen vrši se filtriranje konekcije funkcijom *filterConnected*, odnosno taj čvor se uklanja iz konekcije i vrši se nova provera uslova za čvorove koji ulaze u konekciju. Po završetku pretrage se ispituje dužina konekcije.

- ***filterConnected (waiting visited connectionNodes matrix)***
  Ovoj funkciji kao najbitiniji parametar se prosleđuje kopija matrice koja predstavlja trenutno stanje na tabli. Za svaki čvor koji treba da bude izbrisan iz konekcije se na njegovo mesto u kopiji matrice setuje prazno polje koristeći funckiju *resetField*. Na taj način je omogućeno validno pribavljanje suseda čvora koji se obrađuje.

- ***areAdjacent (node1 node2 matrix)***
  Proverava da li su dva čvora susedi.

- ***getNeighbours (parent visited matrix)***
  Pribalja neposećene susede.

- ***checkChild (rowIndex columnIndex value visited matrix)***
  Pomoćna funkcija funkciji *getNeighbours*. Vrši proveru da li je sused odgovarajuće vrednosti, u okviru granica table i da li nije u listi posećenih.

- ***removeFromList (element list)***
  Briše prosleđeni element iz liste.

Primer detektovanja kraja igre:

```
Enter row and column indexes as a list:
(e 1)
       0 1 2 3 4 5
A      O X - - - -  6
B     X X - - - - -  7
C    O - X O - - - -  8
D   - X X - - - - - -  9
E  - X O - - - - - - -  10
F O X - - - - - - - - -
G  X - - - - - - - - -
H   - - - - - - - - -
I    O X O - - - - -
J     - - O - - - -
K      O O - - - X

"Izigraste se!"
```

Miloš Stanojević 15891
Darko Stevanović 15900