**3D Connect Four - Izveštaj**

**Faza: I**

**Tim: MADTeam**

* **Izgled Tabele :**

Tabela za igru je veličine n = 4 ili n = 6.

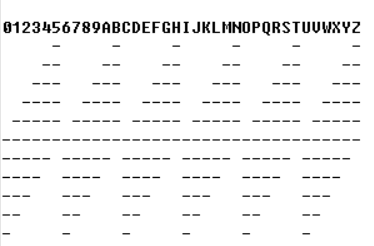
****

* **Reprezntacija ove matrice u memoriji za n=4:**

(((- - - - )(- - - - )(- - - -)(- - - -))

((- - - - )(- - - - )(- - - -)(- - - -))

((- - - - )(- - - - )(- - - -)(- - - -))  
((- - - - )(- - - - )(- - - -)(- - - -)))



* **Reprezntacija ove matrice u memoriji za n=6:**

(((- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -))

((- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -))

((- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -))

((- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -))

((- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -))

((- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)(- - - - - -)))

* **Način pamćenja elemenata u memoriji:**

Matrica se pamti kao asocijativna lista gde je ključ indeks vrste, a vrednost asocijativna lista u kojoj je ključ indeks kolone, a vrednost lista koja koja predstavlja vrednost polja sa trenutnim sadržajem belih( O ) ili crnih( X ) ili praznih( - ) pločica.

* **Bitne funkcije:**
* **main()**

Objašnjenje funkcije: Inicijalizuje igru . Ucitava se dimenzija table, dodjeljuje se globalnoj promenljivoj **tabela**, i zove se **inicijalizujTabelu** koja definise matricu sa pocetnim stanjem igre odnosno plocica rasporedjenih na odgovarajucim pozicijama.

* **inicijalizujTabelu( rbrReda)**

Objašnjenje funkcije: Inicijalizuje matricu označenu kao **tabela** koja predstavlja nasu asocijativnu listu.

Parametri : **rbrReda –** definiše indeks vrste koja se inicijalizuje.

* **InicijalizujRed( rbrReda rbrKolone )**

Objašnjenje funkcije: Inicijalizuje vrstu matrice.

Paramteri: **rbrReda, rbrKolone –** Vrednost parametra rbrReda je odogvarajuća vrsta, a vrednost rbrKolone je velicinaTabele, odnosno dimeznija zadata od strane korisnika za veličinu kocke. U okviru funkcije vrši se provera vrednosti velicinaTabele. Razlikujemo slučajeve kada je:

1. Uneta je vrednost 4 I tada se na (list '- '- '- '-) nadovezuje rekurzivni poziv funkcije InicijalizujRed sa inkrementiranom vrednošću rbrKolone.

2. Uneta je vrednost 6 I tada se na (list '- '- '- '- '- '-) nadovezuje rekurzivni poziv funkcije InicijalizujRed sa inkrementiranom vrednošću rbrKolone.

3. I ostale slučajeve u kojima se na ekranu ispisuje “ Nije uneta ispravna vrednost za velicinu kocke”.

* **potpunPrikaz()**

Objašnjenje funkcije: U funkciji potpunPrikaz() štampa se izgled tabele, sem same tabele štampa se i indeks kolona, odnosno stubića kocke. Ispitivanjem unete vrednosti za velicinaTabele na ekranu se štampa:

1. "0123456789ABCDEF" ukoliko je vrednost za velicinaTabele 4 i

2. "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ" ukoliko je vrednost za velicinaTabele 6.

Zatim se poziva funkcija prikazStanja i nakon toga se ponovo štampaju indeksi kolona.

* **prikazStanja( od do ind1 ind2 )**

Objašnjenje funkcije: Funkcija **prikazStanja** prikazuje tabelu, odnosno štampa je. U okviru nje poziva se funkcija **stampajRed** , a zatim se funkcija **prikazStanja** poziva rekurzivno kako bi se u sledećem pozivu pozvala funkcija stampajRed za naredni red koji treba prikazati na ekranu.

* **stampajRed( od do count ind1 ind2 )**

Objašnjenje funkcije: Funkcija **stampajRed** korisi se za prikaz reda na ekranu, koji se sastoji od elemenata redova tabele sa kojom radimo i blanko znakova. U okviru ove funkcije pozivaju se funkcije: stampajBlankoOd, stampajPolja i stampajBlankoDo koje treba da odštampaju jedan red iz tabele, a zatim se poziva **stampajRed** rekurzivno kako bi se završila štampa za ostatak reda iz tabele čiji se elementi trebaju prikazati na ekranu , funkcija se poziva rekurzivno sve dok vrednos count argumenta ne postane jednaka sa vrednošću veličine kocke, odnosno jednaka sa velicinaTabele.

* **stampajBlankoOd( od trenutni )**

Objašnjenje funkcije: Funkcija štampa blanko sa leve strane reda matrice. Funkcija se poziva rekurzivno sa inkrementiranom vrednošću parametra trenutni.

* **stampajBlankoDo( stampajDo granica )**

Objašnjenje funkcije: Funkcija štampa blanko sa desne strane reda matrice. Funkcija se poziva rekurzivno sa inkrementiranom vrednošću parametra trenutni.

* **stampajPolja(od do red ind1 ind2)**

Objašnjenje funkcije: Funkcija štampa jedno polje na ekranu sa indeksima red, ind1, ind2.

* **potez()**

Objašnjenje funkcije: U ovoj funkciji se poziva funkcija potpunPrikaz kako bu se prikazalo trenutno stanje na ekranu. Zatim se učitava potez u promenljivu potez. Zatim se poziva funkcija proveriPotez i poziva se rekurzivno funkcija potez.

* **ProveriPotez( potez )**

Objašnjenje funkcije: Funkcija proverava da li je potez koji je odigran validan. Uzima prvi element iz stringa „potez“ i određuje njegov char code. Oduzima 55 jer karakter A ima kod 65, a nama predstavlja broj 10. Pošto postoji 7 mesta između velikih slova i brojeva u ascii tabeli ukoliko je broj dodajemo mu 7. Određujemo „rbrReda“ celobrojnim deljenjem izračunatog poteza sa veličinom tabele. Takođe, određujemo „rbrStubica“ kao ostatak deljenja izračunatog poteza sa veličinom tabele. Ukoliko je vrh stubića prazan poziva se funkcija odigrajPotez.

* **odigrajPotez( rbrReda rbrStubica trenutni )**

Objašnjenje funkcije: Funkcija ispituje da li je element jednak znaku „-” .Provera se vrši od dna ka vrhu. Ukoliko jeste, to znači da se potez može odigrati, odnosno upisuje se u matricu. Ukoliko nije, poziva se odigrajPotez rekurzivno za naredni element.

* **promeniIgraca()**

Objašnjenje funkcije: Funkcija menja vrednosti promenljivih trenutniIgrac i trenutnaBoja. Ukoliko je trenutni igrač čovek, trenutniIgrac se postavlja na C, i obrnuto. Ukoliko je trenutna boja X, trenutnaBoja se postavlja na O, i obrnuto.

* **jeKrajIgre()**

Objašnjenje funkcije: Proverava vrhove svih stubića, ukoliko su svu sva polja polja na vrhu popunjena, znači da nema više poteza koji bi mogli da se odigraju, što znači da je kraj igre.

* **odabirPrvogIgraca()**

Objašnjenje funkcije: Funkcija na ekranu ispisuje: "Izbor prvog igraca (H/C): ", a zatim čita ono sto korisnik unese i upisuje tu vrednost u promenljivu prviIgracTemp. Zatim postavlja globalnu promenljivu trenutniIgrac na vrednost prviIgracTemp, ukoliko je jedna od te dve vrednosti (H/C) u suprotnom štampa na ekranu "Ulaz mora biti H ili C".

* **odabirBoje()**

Objašnjenje funkcije: Funkcija na ekranu ispisuje: "Izbor boje prvog igraca (X/O): ", a zatim čita ono sto korisnik unese i upisuje tu vrednost u promenljivu bojaIgracaTemp. Zatim postavlja globalnu promenljivu bojaIgracaTemp na vrednost bojaIgracaTemp, ukoliko je jedna od te dve vrednosti (X/O) u suprotnom štampa na ekranu "Ulaz mora biti X ili O".

**Faza: II**

**Implementirane funkcije u fazi II:**

* Funkcija **ispitajPoteze(tabelaZaIspitivanje)**

Pozivamo funkciju **proveriPoteze** , i na kraju kreiramo listu svih mogućih sledećih poteza na osnovu trenutnog stanja.

* Funkcija **proveriPoteze (potez tabelaZaIspitivanje ostatakliste)**

Popunjavamo listu svih mogućih poteza na osnovu trenutnog stanja.

* Funkcija **proveriPoene ()**

Proveravamo, čuvamo i prikazujemo ukupan broj poena koji su ostvarili crni i beli igrač na osnovu trenutnog stanja.

* Funkcija **proveriSveY (z x)**

Pozivamo funkciju **proveriY** za svaki stubić.

* Funkcija **proveriY (z y x xNiz oNiz)**

Tražimo nizove X ili O i na osnovu njih povećavamo poene.

* Funkcija **proveriSveZ (y x)**

Pozivamo funkciju **proveriZ** za svaku ravan gde su x i y fiksni.

* Funkcija **proveriZ (z y x xNiz oNiz)**

Tražimo nizove X ili O i na osnovu njih povećavamo poene.

* Funkcija **proveriSveX (z y)**

Pozivamo funkciju **proveriX** za svaku ravan gde su z i y fiksni.

* Funkcija **proveriX (z y x xNiz oNiz)**

Tražimo nizove X ili O i na osnovu njih povećavamo poene.

* Funkcija **proveriDijagonalu (z y x kraj xNiz oNiz incZ incY incX uslov uslovInc)**

Tražimo nizove X ili O, i na osnovu njih povećavamo poene za bilo koju dijagonalu u bilo kojoj ravni.

* Funkcija **proveriSveZY4 (x)**

Pozivamo proveravanja dijagonala za obe dijagonale u ravni kada je x fiksno, pored toga to vazi kada je veličina tabele jednaka 4.

* Funkcija **proveriSveZY6 (x)**

Pozivamo sva proveravanja dijagonala za sve dijagonale koje imaju 4 ili više polja, u ravni kada je x fiksno i kada je veličina tabele 6.

* Funkcija **proveriSveYX4 (z)**

Pozivamo proveravanja dijagonala za obe dijagonale u ravni kada je z fiksno, pored toga to vazi kada je veličina tabele jednaka 4.

* Funkcija **proveriSveYX6 (z)**

Pozivamo sva proveravanja dijagonala za sve dijagonale koje imaju 4 ili više polja, u ravni kada je z fiksno i kada je veličina tabele 6.

* Funckija **proveriSveZX4 (y)**

Pozivamo proveravanja dijagonala za obe dijagonale u ravni kada je y fiksno, pored toga to vazi kada je veličina tabele jednaka 4.

* Funkcija **proveriSveZX6 (y)**

Pozivamo sva proveravanja dijagonala za sve dijagonale koje imaju 4 ili više polja, u ravni kada je y fiksno i kada je veličina tabele 6.

* Funckija **proveri3DDijagonale4 ()**

Proveravamo sve 3D dijagonale kocke kada je veličina tabele jednaka 4.

* Funckija **proveri3DDijagonale6 (y)**

Proveravamo sve 3D dijagonale koje imaju 4 ili više polja na sebi, kada je veličina kocke jednaka 6.

**Faza: III**

**Implementirane funkcije u fazi III:**

* Funkcija **proveriPotezAlfaBeta (potez stanje boja)** sluzi da odigrava potez u Alfa-Beta algoritmu
* Funkcija **promeniIgracaAlfaBeta (value)** služi da menja prosledjenog igrača
* Funkcija **alfaBeta (stanje potez alfa beta trenutnaDubina igracNaPotezu)** podrazumeva min-max algoritam sa Alfa-Beta pruningom
* Funkcija **maxIgrac (listaPoteza najboljiPotez dubina alfa beta predhodnoStanje igracNaPotezu)** je petlja za MaxIgraca u min-max algoritmu
* Funkcija **minIgrac (listaPoteza najboljiPotez dubina alfa beta predhodnoStanje igracNaPotezu)** je petlja za MinIgraca u min-max algoritmu

**Faza: IV**

**Implementirane funkcije u fazi IV:**

* Funkcija **!daLiXPobedjuje (stanje)** provara da li je X pobednik u datom stanju
* Funkcija **!daLiOPobedjuje (stanje)** provara da li je O pobednik u datom stanju
* Funkcija **!daLiXDobijaPoene (stanje)** provara da li je X ima vise poena u datom stanju, za razliku od starog stanja
* Funkcija **!daLiODobijaPoene (stanje)** provara da li je O ima vise poena u datom stanju, za razliku od starog stanja
* Funkcija **generisiCinjenice (stanje z y x)** generise činjenice formata (z y x vredostNaPoljuZYX)
* Funkcija **heuristika(stanje igracNapotezu)** predstavlja mašinu za zaključivanje