*ispuniJedanRed (el n brojac)*

Ispunjava jedan red stubovima pozivanjem funkcije ispuniListu, koja će vratiti jedan stub. Argumenti: el – element koji treba postaviti na svim pozicijama; n – broj elemenata reda; brojac – brojač koji ima početnu vrednost koja je ista kao n.

*izvuciBroj (lista)*

Vraća drugi element liste koja ima dva elementa.

*ispuniStub (el n)*

Kreira jedan stub tako što pravi listu od n elemenata, gde je svaki element karakter el.

*kreirajTablu (el n brojac)*

Kreira početnu tablu pozivanjem funkcije ispuniJedanRed, koja će vratiti jedan red. Argumenti: el – element koji treba postaviti na svim pozicijama; n – broj elemenata reda; brojac – brojač koji ima početnu vrednost koja je ista kao n.

*vratiRed(lista brojac)*

Funkcija koja iz svake podliste „vadi“ odredjeni broj elemenata, onoliko koliko se prosledi argumentu brojac.

*vratielemente (lista brojac)*

Ova funkcija je zaduzena za vracanje elementa iz liste. Poziva se u funkciji vratiRed, kako bismo vratili onoliko elemenata koliko jedan red u interfejsu treba da ispise, iz razloga sto se broj elemenata u svakom redu menja.

*sortirajTablu(lista brojac &optional(p 1))*

Funkcija koja vraca sortiranu tablu kako bi bila pogodna za kasnije prikazivanje, od argumenata uzima celu tablu (lista), brojac i opcioni argument koji pomaze pri sortiranju drugog dela table ciji se elementi smanjuju (sredina ima najvise elemenata).

*brisiRed (lista brojac)*

Brise podlistu iz liste koja je vec prikazana.

*brisiIzSvakogReda (lista brojac)*

Kako funkcija vratiRed ne bi vracala iste elemente svaki put, ova funkcija to resava. Poziva funkciju brisiRed, kako bi se rekurzivno primenilo brisanje za svaku od 4 podlista glavne table.

*stampaj\_listu (l brojac)*

Funkcija sluzi za stampanje prvog dela table i sredine. U odnosu na argument brojac prikazuje odredjeni broj razmaka pre prikaza elementa.

*stampaj\_listu\_dole (l brojac)*

Kao i prethodna funkcija, ova funkcija sluzi za stampanje donjeg dela tabele.

*stampa (lista brojac &optional(p 0))*

Glavna funkcija za sam prikaz table, od argumenata uzima celu tablu, brojac i opcioni argument koji sluzi za proveru koju funkciju za prikazivanje cemo pozvati, stampaj\_listu ili stampaj\_listu\_dole, u prevodu kako bismo znali kada smo stigli do polovine. Rezultat ove funkcije je prikazana tabla.

*lepPrikazListe (lista)*

Stampa elemente prosledjene liste

*prikaziSlova (lista)*

Sluzi za prikazivanje slova iznad i ispod tabele. Poziva funkciju prikaziSlovaPomocna kako bi se nakon svakog cetvrtog elementa liste napravio razmak.

*prikaziSlovaPomocna (lista)*

Stampa elemente koji su prosledjeni kao lista.

*krajIgre ()*

Proverava da li je igra završena (budući da je korišćena 4x4 tabla, dolazi do kraja igre kada je odigran 64. potez). Kada se odigra 64. potez poziva se funkcija Pobednik kojoj ce biti kao argument prosledjena popunjena tabla.

*Pobednik(lista)*

Konacna funkcija za proglasenje pobednika kojoj se prosledjuje cela tabla. Uz pomoc funkcije xILIo koja kao argument poziva funkciju proveriPobednika, ciji je argument cela tabla, odredjuje da li u odigranoj tabli imamo vise poklapanja X-a ili O-ksa.

*xILIo(lista element)*

Broji koliko X-eva i O-kseva ima u prosledjenoj listi, argument element predstavlja karakter za koji proverava broj poklapanja u prosledjenoj listi.

*proveriPobednika(lista)*

Ovo je funkcija kojoj se prosledjuje cela tabla u kojoj se 3 puta poziva funkcija proveriZaKolone ali sa razlicitim argumentima kako bi svaka od njih napravila posebnu listu x-eva i o-kseva zbog provere poklapanja po koloni, redu i dijagonali. Povratak iz ovih listi appenduje u novu listu.

*proveriZaKolone(lista)*

Ovoj funkciji se prosledjuje cela tabla, iz koje se poziva funkcija proveriZaKolonePomocna zbog rekurzije

*proveriZaKolonePomocna(lista)*

Pomocna funkcija za funkciju proveriZaKolone kojoj se prosledjuje jedna lista koja se sastoji od 4 podliste. Ova funkcija poziva funkciju proveriJednakostListe kako bi se proverilo da li su ugnjezdene 4 liste sekvence od 4 ista elementa (X ili O). Ova funkcija vraca listu x-eva i o-kseva.

*proveriJednakostListe(lista element)*

Proverava da li su elementi u jednoj listi jednaki sa prosledjenik elementom koji je argument funkcije (X ili O). Vraca X ukoliko su svi elementi liste X, vraca O ukoliko su svi elementi liste O

*nadjiJedanRed (brReda matrica)*

Vraća n-ti element niza. Argumenti: brReda – broj elementa niza; matrica – niz.

*proveriValidan (stub)*

Proverava da li može da se odigra potez na prosleđenom stubu. Ako postoji bilo koji element, čija je vrednost ’’-’’, znači da je moguće odigrati potez na tom stubu (u tom slučaju se vraća true), u suprotnom nije i vraća se nil.

*proveriPotezAtom (potez)*

Proverava da li je potez validan, ako je potez unet kao atom (npr. ’C).

*proveriPotezLista (lista)*

Proverava da li je potez validan, ako je potez unet kao lista (npr. (1 3); napomena: potezi počinju od nule).

*potezValidan (potez)*

Proverava da li je unet potez validan tako što proverava da li je uneti potez atom ili ne, pri čemu poziva odgovarajuću funkciju provere poteza.

*vratiElement (element ubacenElement)*

Ako je element tipa ’’-’’ i nije prethodno ubačen element na štapić, vraća ’’figuru’’ trenutnog igrača, a ako jeste, vraća ’’-’’.

*ubaciNaStapic(stapic ubacenElement)*

Ubacuje figuru trenutnog igrača na štapić prosleđen kao argument.

*nadjiStapic (vrsta brojac)*

Vraća vrstu sa modifikovanim štapićem.

nadjiVrstu (tabela brojacX brojacY)

Vraća tabelu sa modifikovanom vrstom.

*konvertujUKoordinate (broj)*

Konvertuje broj štapića u njegove koordinate (npr. konvertuje 12 u (3 0)).

*konvertujPotez (potez)*

Konvertuje potez iz tipa karakter u tip lista (npr. konvertuje potez C u (3 0)).

*povuciPotez (potez)*

Odigrava potez i vraća modifikovanu tabelu (ne menja originalnu tabelu).

*vratiSledecuFiguru ()*

Vraća karakter sledeće figure (npr. ako je na potezu X, vraća se O).

*odigrajIPrikazi ()*

Odigrava potez, modifikuje tabelu, prikazuje novo stanje na ekranu, inkrementira brojač poteza i menja karakter trenutnog igrača.

*nevalidanPotez ()*

Štampa nevalidan potez i vraća korisnika da ponovo unese potez.

*potez ()*

Čita korisnikov potez sa tastature, proverava da li je validan i ako jeste, poziva funkciju odigrajIPrikazi(), a ako nije, poziva funkciju nevalidanPotez().

*sviMoguciPotezi (globalSlova)*

Vraća listu svih mogućih poteza trenutnog stanja tabele. Ako potez nije validan, dodaje se NIL u listu.

*sviMoguciPoteziBezNil ()*

Izbacuje NIL elemente iz liste, koju generiše funkcija sviMoguciPotezi (globalSlova).

*alfabeta (stanje graf alfa beta dubina maxdub rezultat minmax)*

Vraća najbolji potez po min-max algoritmu sa alfa-beta odsecanjem. Stanje – trenutno stanje; graf – graf prelaza stanja (u projektu će se ovde pozivati funkcija za određivanje svih mogućih stanja na osnovu trenutnog stanja table); alfa – vrednost alfa (u početku kao argument se unosi veoma mala vrednost); beta – vrednost beta (u početku kao argument se unosi veoma velika vrednost); dubina – trenutna dubina (u početku se kao argument unosi 0); maxdub – maksimalna dubina; rezultat – ako je true, rezultat min-max algoritma vratiće stanje, ako je false, vratiće vrednost za taj čvor (heuristika); minmax – ako je true, radi se max, ako je false, radi se min (u početku se kao argument unosi true). **Napomena**: algoritam je napisan da radi za slučaj sa prezentacija (treba vratiti ,,B’’).