*ispuniVrstu (el n brojac)*

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

*izvuciBroj (lista)*

Vraća drugi element liste koja ima dva elementa.

*ispuniListu (el n)*

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

*kreirajTabluPom (el n brojac)*

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

*novaIgra (n igracXiliO)*

Definiše promenljive potrebne za funkcionisanje igre. Argumenti: n – veličina kocke; igracXiliO – određuje da li čovek igra prvi ili ne.

*krajIgre ()*

Proverava da li je igra završena (budući da je korišćena 4x4 tabla, dolazi do kraja igre kada je odigran 64. potez).

*nadjiVrstu (brVrste matrica)*

Vraća n-ti element niza. Argumenti: brVrste – 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.