# Stanje igre je predstavljeno na sledeci nacin:

**‘( (tabla) (“XiliO” trenutnoNaPotezu) (brojStekovaX brojStekovaO) velicinaTable )**

Tabla je implementirana kao lista koja sadrzi liste koje predstavljaju redove. Svaki red sadrzi listu polja. TrenutnoNaPotezu—0 ukoliko je na potezu racunar, 1 ukoliko je na potezu covek.

**Opisi funkcija**

1. **praviTablu**—vraca tablu za igranje zadate velicine.
2. **napuniTablu—**postavlja odgovarajuci broj zetona na tablu.
3. **inicijalnoStanje**—generise pocetno stanje (unutar ove fje se odredjuje ko igra prvi).
4. **proveriCiljnoStanje**—provera da li imamo pobednika.
5. **zetoniNaTabli**—vraca polja koja sadrze zetone (pomocna fja kod provere validnosti poteza).
6. **rastojanje** – vraca broj koraka izmedju dva polja na tabli (pomocna fja kod provere validnosti poteza).
7. **meniNajblizi**—vraca polje sa zetonima koje je najblize prosledjenom polju (ili vise ukoliko ih ima, pomocna fja kod provere validnosti poteza).
8. **daLiSePriblizavaNajblizem**—provera da li se prilikom odigravanja poteza priblizivamo najblizem polju koje sadrzi zetone (pomocna fja kod provere validnosti poteza).
9. **crtajMatricu**—generise interfejs na osnovu trenutnog stanja.
10. **validanPotez1—**Proverava da se pokusalo da se krene sa belog polja, proverava da li je kretanje po dijagonali I poziva funkciju **proveriValidnoVisinuPriblizavanje** koja vrsi ostale provere
11. **proveriValidnoVisinuPriblizavanje**-poziva fju koja ispituje da li se priblizavamo najblizem za spajanje, ako je ispunjen uslov da se priblizavamo, imamo grananja u odnosu na stanje na table, poziv funkcija ako su ispunjeni uslovi, **pomeriStekNaPrazno,** **pomeriStekNaStek.**
12. **pomeriStekNaStek- i**spituje da li su ispunjeni uslovi za spajanje dva steka, ako jesu, poziva funkciju koja **pomeriDiskove** koja vrsi promunu stanja.
13. **pomeriDiskove-**pomera diskove sa jednog steka na drugi, ispituje da li stek pun, I poziva funkciju za proveru ciljnog stanja