**Faza 1**

## Vodeća ideja pri razmatranju problema:

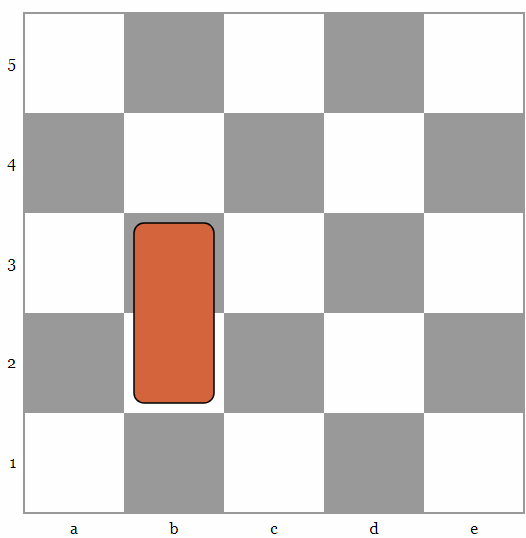
# U svakom trenutku igre je moguće predstaviti stanje u obliku (xVal,oVal,tabla) pri čemu xVal služi za pamćenje broja mogućih poteza igrača X dok oVal pamti broj mogićih poteza za igrača O (za trenutno stanje na tabli). Tabla pamti matricu stanja na tabli.

# Ako posmatramo praznu tablu sa N vrsta i M kolona, uočavamo da postoji N\*(M-1) mogućih poteza za igrača X (koji postavlja figure vertikalno) i M\*(N-1) za igrača O (koji postavlja figure horizontalno).

# Odigravanjem jednog poteza igrač X smanjuje broj mogućih poteza (sebi) za 2,3 ili 4. U isto vreme, taj potez smanjuje broj mogućih poteza protivnika za 0,1,2, 3 ili 4.

# Primer smanjivanja svojih poteza za 3:

# (igrač X više ne može odigrati poteze: b1,b2 i b3)



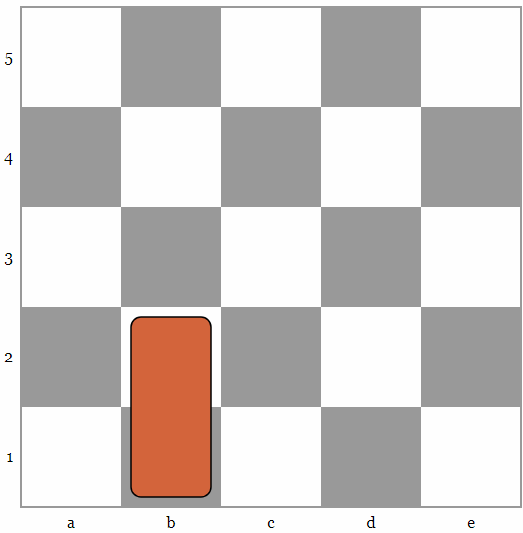
# Primer smanjivanja svojih poteza za 4:

# (nakon poteza b2 igrač X više ne može odigrati poteze: b1,b2,b3 i b4)

# 

# Primer smanjivanja svojih poteza za 2:

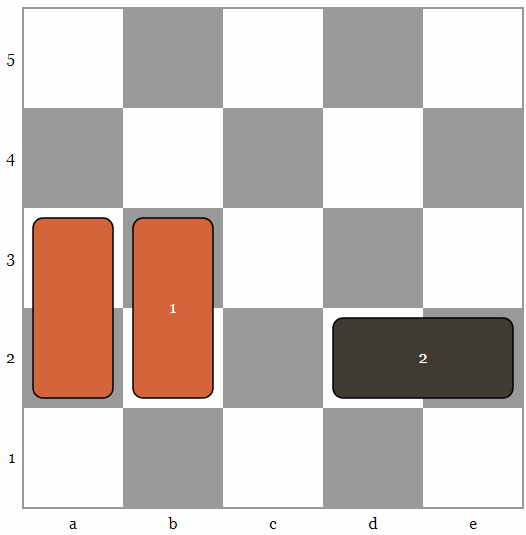
# (nakon poteza b1 igrač X više ne može odigrati poteze: b1 i b2)



# Odigravanje poteza u odnosu na protivnika:

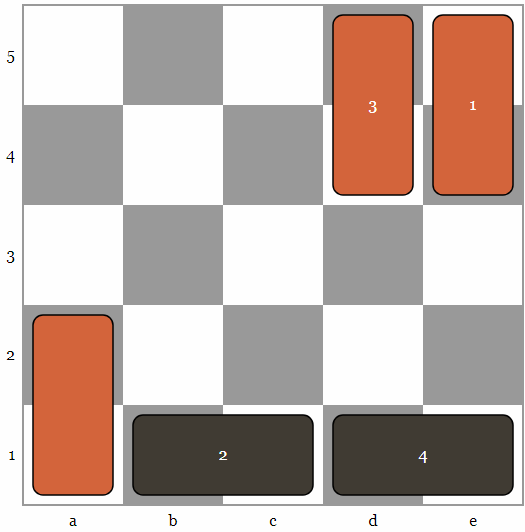
# Primer smanjivanja protivnikovih poteza za 0:

# (nakon X-ovog poteza a2 igrač O može odigrati isti broj poteza kao i pre X-ovog odigravanja a2)



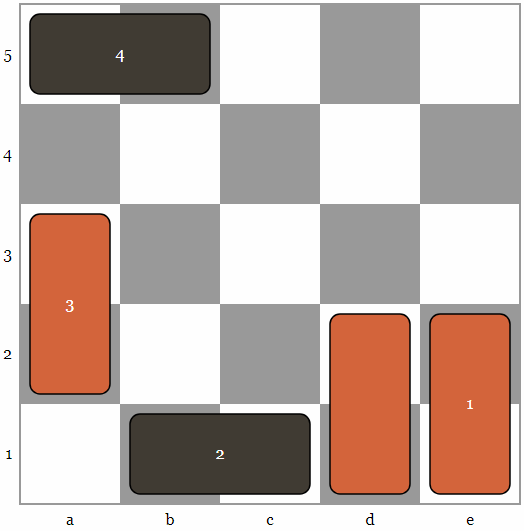
# Primer smanjivanja protivnikovih poteza za 1:

# (nakon X-ovog poteza a1, igrač O može odigrati 1 potez manje (a2))



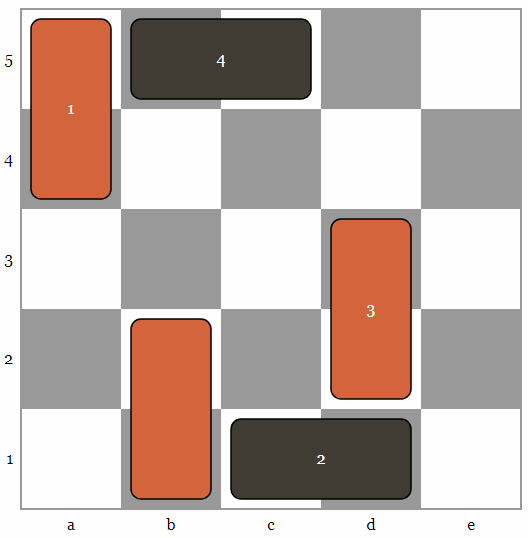
# Primer smanjivanja protivnikovih poteza za 2:

# (nakon X-ovog poteza d1, igrač O može odigrati 2 poteza manje (b2,c2))



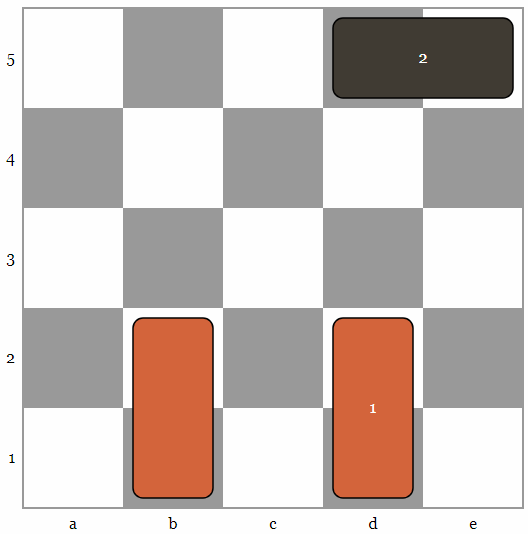
# Primer smanjivanja protivnikovih poteza za 3:

# (nakon X-ovog poteza b1, igrač O može odigrati 3 poteza manje (a1,a2,b2))



# Primer smanjivanja protivnikovih poteza za 4:

# (nakon X-ovog poteza b1, igrač O može odigrati 4 poteza manje (a1,a2,b2))



# Sve do sada izloženo važi kako za igrača X tako i za igrača O, odnosno za bilo kog igrača koji odigrava potez.

## Funkcije

# start\_game – na osnovu unesenih podataka pravi početno stanje (xVal,oVal,tabla) ukoliko su uneseni podaci validni (u suprotnom vraća False).

# show\_table – štampanje table na osnovu trenutnog stanja

# is\_valid – za unetog igrača na potezu i uneti potez proverava da li su polja u okvirima matrice i da li su slobodna

# igraj\_potez – proverava da li je potez validan (koristeći is\_valid) i odigrava potez (za igrača koji je trenutno na potezu) ukoliko jeste validan, u suprotnom vraća False. Ne bavi se time ko je prosledio potez (igrač ili kompjuter / drugi igrač).

# game\_in\_progress – proverava da li igra još uvek traje (da li igrač koji je na potezu ima mogućnosti za odigravanje poteza).