Dokumentacija - “Spoji 4”

1. UVOD

Temeljna ideja projekta je bila napraviti igricu “Spoji 4” za jednog igrača u Pythonu. Iako to nije neka orginalna ideja, odlučila sam se za taj projekt zbog toga što me je oduvijek zanimalo kako rade igrice u kojima igrate protiv programa tj. u kojima postoji algoritam koji bira svoje poteze na temelju trenutnog stanja igre.

O autoru bismo mogli reći samo to da se zove Leonarda Lovrić, ide u 2.5 GAM-a te da radi ovaj projekt kao zadatak iz informatike. Zbog toga što baš i nije moguće napraviti program koji ima barem neke šanse pobijediti igrača bez ikakvog znanja o sličnim programima koji ‘sami razmišljaju’, moglo bi se reći da su bile potrebne razne web stranice (kao Wikipedia) i nekolicina Youtube videa (koji objašnjavaju princip rada minimax algoritma i alpha-beta pruninga) za izradu ove igrice.

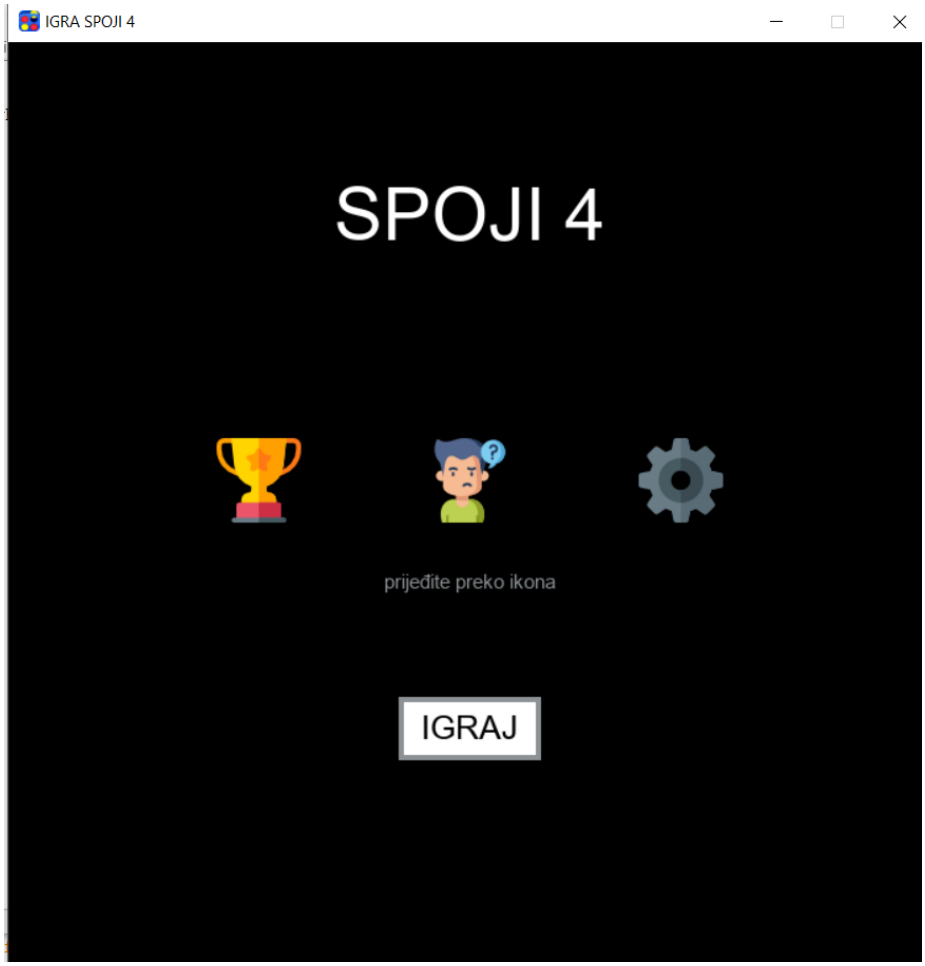
2. DETALJAN OPIS RADA

2.01 OPĆE INFORMACIJE O PRIKAZU

‘Spoji 4’ je grafička igra te će vam pygame pri ulasku u program izbaciti novi prozor veličine 700*700 piksela na kojemu će se igra odvijati.

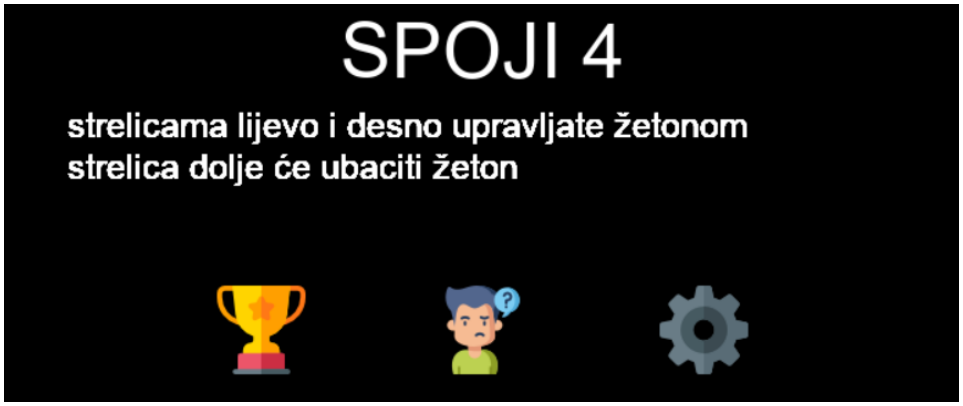
2.1 POČETNI EKRAN

Kada uđete u igricu prvo što ćete uočiti je početni ekran na kojemu naravno piše ime igrice tj. “SPOJI 4” te se uz gumb “IGRAJ” kojim možete započeti igrati nalaze još tri interaktivne ikone. Prelaskom miša preko neke ikone dobiti ćete odgovarajuće informacije.



2.12 KONTROLE ZA IGRU

Prelaskom preko ikone ‘upitnik’ koja se nalazi u sredini dobiti ćete upute za igranje tj. saznati ćete da žeton pomičete lijevom i desnom strelicom, a ubacujete ga u željenu kolonu strelicom prema dolje.

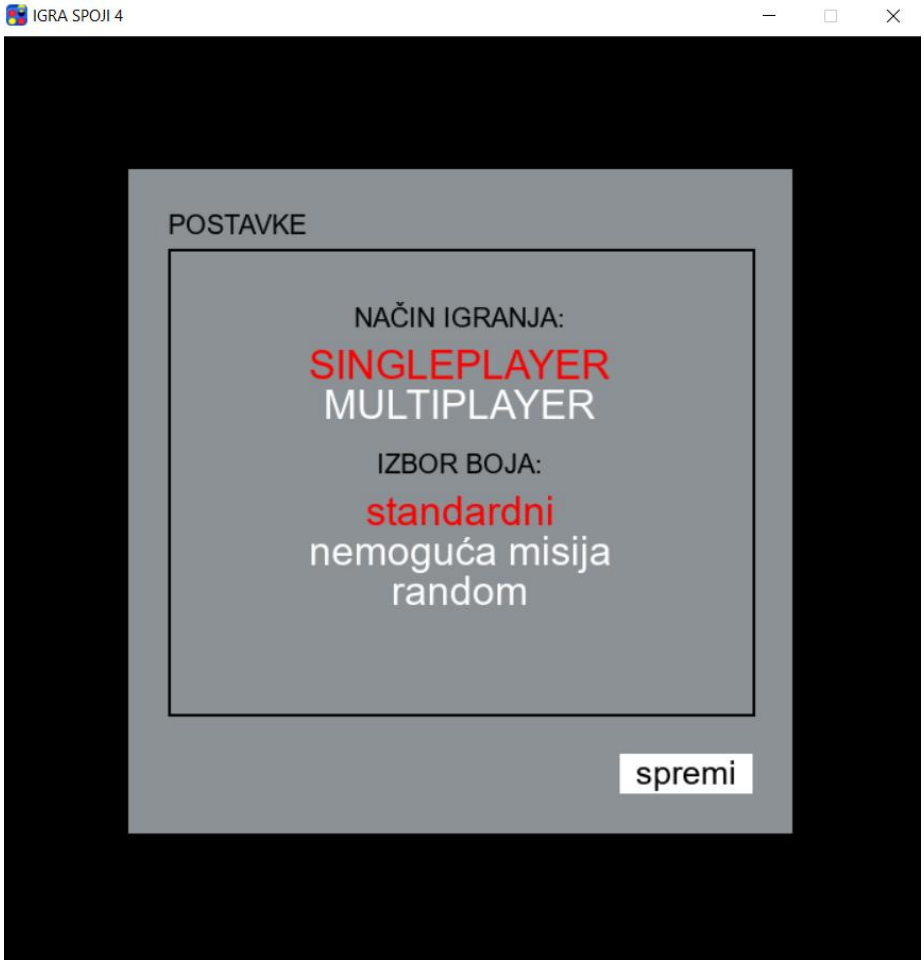


2.12 POSTAVKE

Ukoliko ste tek ušli u igricu automatski je postavljen mode ‘SINGLEPLAYER’ i izbor boja ‘standardni’; ako to želite promijeniti, učinite to uz pomoć ikone ‘postavke’. Prelaskom preko te ikone ispisati će se daljnja uputa tj. naputak da ju morate kliknuti te upozorenje da će otvaranje postavki resetirati trenutni rezultat pobjeda i poraza kako bi se spriječilo lažiranje pobjeda (bez toga biste mogli promijeniti mode u ‘MULTIPLAYER’ i napraviti da ‘IGRAČ 1’ pobijedi 100 puta te opet vratiti u ‘SINGLEPLAYER’ gdje biste tada imali 100 pobjeda)

Klikom na ikonu ‘postavke’ otvori se izbornik na kojemu možete izabrati između ‘SINGLEPLAYER’ ili ‘MULTIPLAYER’ načina igranja tj. možete izabrati želite li igrati protiv računala ili osobe koja se nalazi pored vas. Također, možete uz ‘standardni’ izbor boja (plava ploča, žuti žetoni i protivnički crveni žetoni) izabrati i ‘nemoguću misiju’ u kojoj je jedini način da pobijedite zapamtiti sve pozicije svojih žetona (siva ploča, svi žetoni su crveni). Uz to postoji i ‘random’ izbor boja koji vam nudi drugačije obojane dijelove igre (ploča i dvije vrste žetona) svaki put kada ga odaberete! U izboru ‘random’ moguće je dobiti boje: bijela, siva, crvena, tirkizna, žuta i plava.

Trenutno izabrani ‘način igranja’ i ‘izbor boja’ prikazan je crveno, a ostale moguće opcije prikazane su bijelo. Klikom na gumb ‘spremi’ izlazite iz postavki te će vam ponovno biti prikazan početni zaslon.



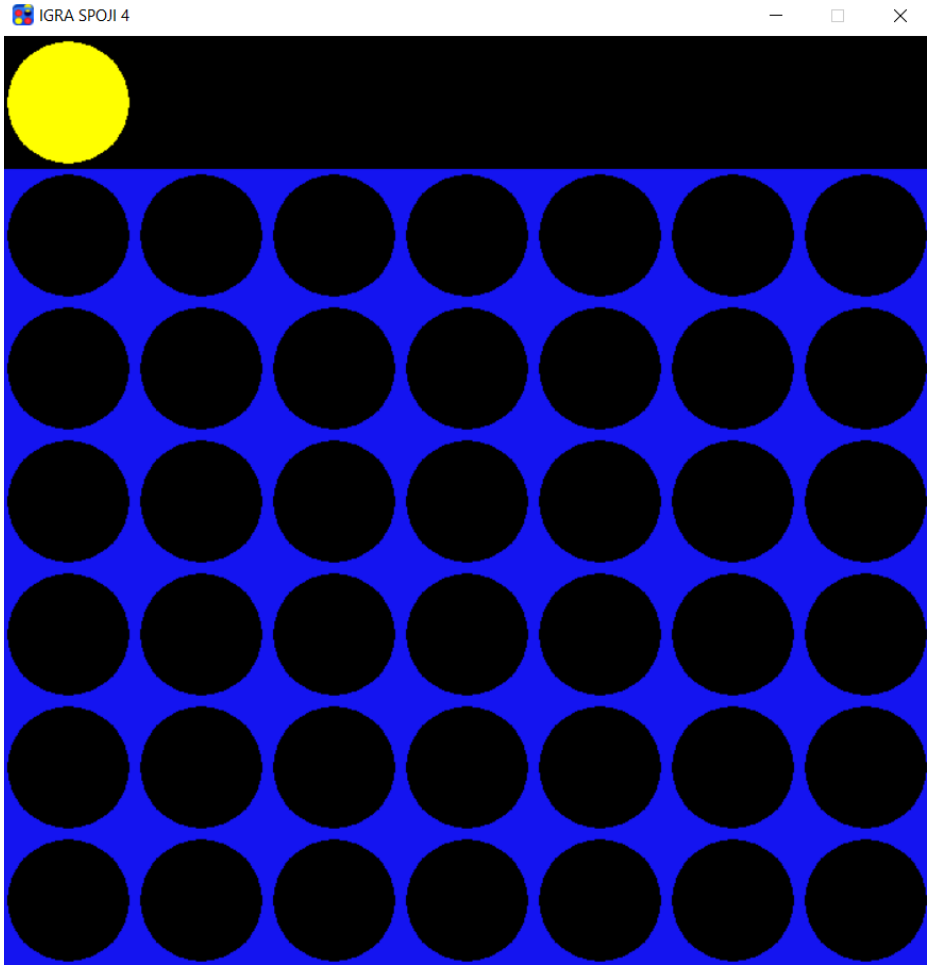
2.13 TRENUTNI REZULTAT

Ako pokazivačem miša prijeđete preko ikone ‘pehar’ saznati ćete trenutni rezultat. Ukoliko ste trenutno u ‘SINGLEPLAYER’ mode ispisati će vaše pobjede i poraze protiv programa, a ako ste u ‘MULTIPLAYER’ mode ispisati će pobjede ‘IGRAČA 1’ i ‘IGRAČA 2’.

Zanimljivo je to da su boje kojima su ispisane pobjede svakog igrača (ili pobjede vas i pobjede računala tj. porazi) sinkronizirane sa trenutno odabranim bojama za žetone igrača. Na slikama iznad bi to značilo da ‘IGRAČ 1’ ima žute žetone, a ‘IGRAČ 2’ crvene. Dakle ako promijenite ‘izbor boja’ promijeniti će se i boja ispisa trenutnog rezultata.

2.2 EKRAN IGRE

Klikom na gumb ‘IGRAJ’ pokrenuti ćete novu igru te će vam prvo biti prikazana prazna ploča i žeton iznad prve kolone (tj. ‘nulte’ – jer u kodu ima indeks nula).



2.21 TIJEK IGRE

Žeton pomičete strelicama do kolone u koju ga želite ubaciti. Kada stisnete strelicu prema dolje iznad željene kolone vaš se žeton pojavi na prvom slobodnom mjestu u toj koloni. Ako je ploča prazna pojaviti će se u prvom (u kodu je nulti) redu tj. najdoljnjem redu, ali ako je neki žeton već tamo pojaviti će se u prvom sljedećem. Kada ubacite žeton u neku kolonu on tamo ostaje do kraja igre.

Dakle, igrači ili igrač i program naizmjence ubacuju svoje žetone s jednim ciljem – spojiti 4 žetona u bilo kojem smjeru.

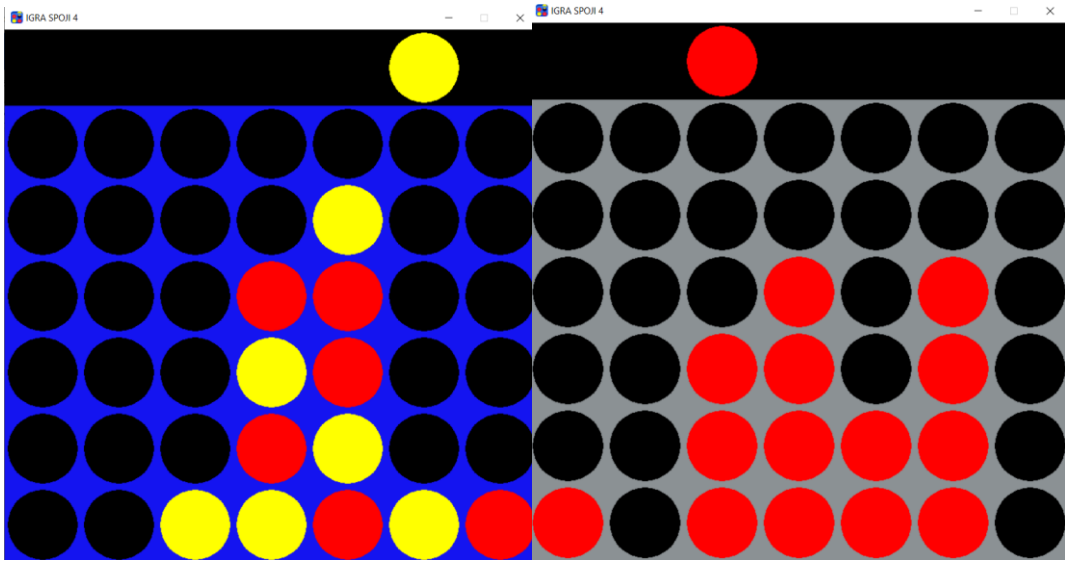
2.22 PRVI POTEZ

Nije nužno da u ‘SINGLEPLAYER’-u vaš žeton bude prvi, niti da u ‘MULTIPLAYER’-u ‘IGRAČ 1’ bude prvi na redu zato što se pomoću biblioteke random bira tko će prvi igrati. Ukoliko program prvi igra, njegov će se žeton također stvoriti iznad prve kolone te će izgledati kao da se i on kreće pomoću strelica do željene kolone.

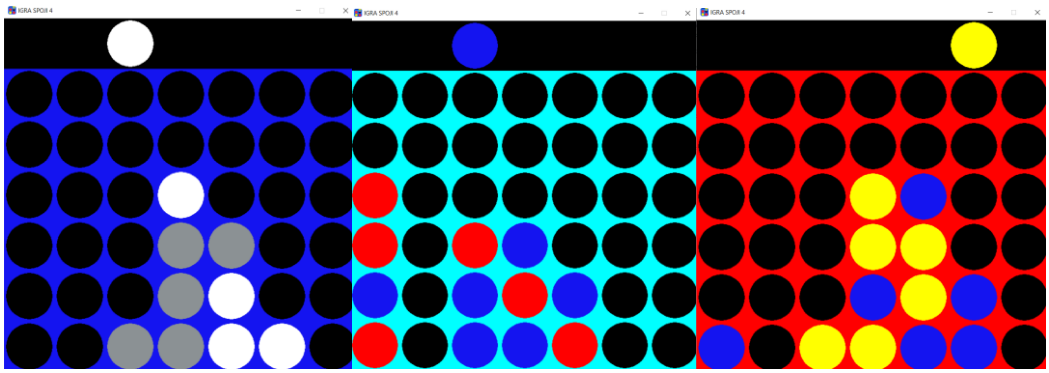
2.23 PRIKAZ IGRE S OBZIROM TRENUTNI IZBOR BOJA

Žetoni i ploča mogu biti različitih boja ovisno koji ste 'izbor boja' odabrali.

'standardni' izbor boja (lijevo) i 'nemoguća misija' (desno) :



Neki od 'random' odabira:

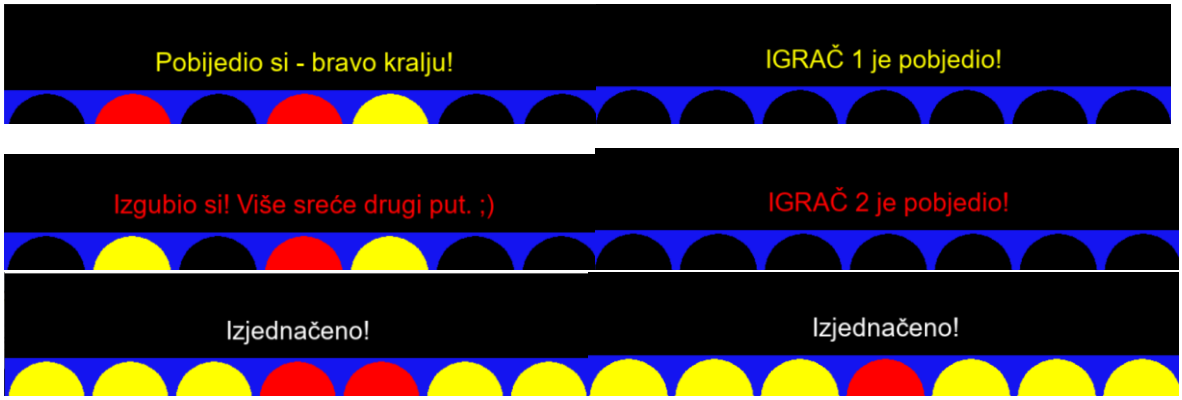


2.24 KRAJ IGRE

Igra završava kada jedan od igrača uspije spojiti 4 u bilo kojem smjeru (horizontalno, okomito, dijagonalno lijevo i desno) ili ako se sva prazna mjesta na ploči popune tj. kada se svi crni krugovi postanu boje žetona.

Na kraju igre program će iznad ploče ispisati tko je pobijedio ili ako nema spojenih 4 pisati će da je izjednačeno. Naravno, i tekst će biti u boji pobjedničkih žetona ili bijeli ukoliko pobjednika nema.

Mogući ishodi igre - 'SINGLEPLAYER' mode (lijevo) i 'MULTIPLAYER' mode (desno):



Nakon kraja igre ploča sa pobjedničkim stanjem biti će prikazana još 4 sekunde te nakon toga će vam ponovno biti prikazan početni ekran.

2.25 IZLAZAK IZ IGRE

Ukoliko želite izaći iz igre jednostavno kliknite na x u gornjem desnom kutu prozora – ako se u tom trenutku igrate to će vas vratiti na početni ekran i resetirati trenutni rezultat, a kliknete li x dok ste na početnom ekranu zatvoriti ćete cijelu igru tj. Izaći ćete iz programa.

3. TEHNIČKE INFORMACIJE

3.1 TEHNOLOGIJE

Backend i frontend programa su pisani u Python programskom jeziku te se nalaze u python dokumentu ‘spoji 4.py’.

Uz kod su priložene i još 4 slike zbog estetskih razloga; ‘upitnik.png’, ‘pehar.png’, ‘postavke.png’ i ‘ikona.png’. Ikone korištene u programu skinute su sa stranice https://www.flaticon.com/

Svi se dijelovi moraju nalaziti u istoj mapi kako bi igrica funkcionirala.

3.2 IZVEDBA RJEŠENJA

Dok ne izađete iz programa aktivna je while petlja u kojoj je sadržan početni ekran i još jedna while petlja koja je aktivna dok igrate igru tj. od pritiska na gumb ‘IGRAJ’ pa sve do kraja igre. Uz glavni dio postoji puno definicija te one ustvari rade većinu ‘posla’.

3.21 BIBLIOTEKE

Projekt je napravljen u Python-u uz importanje različitih biblioteka;

- numpy

- pygame
- sys
- random
- time
- math

3.22 DEFINICIJE ZA OSNOVNE NAREDBE (+ prototip programa)

Igra bi funkcionirala i sa samo ovom skupinom naredbi te je tako izgledala prva verzija igrice ‘SPOJI 4’. Prototip projekta je bila igra bez grafičkog prikaza u ‘MULTIPLAYER’ načinu igranja koja se igra u Python shellu upisivanjem broja za odabir kolone.

Uz pomoć biblioteke numpy program napravi ploču koja funkcionira kao lista listi koje na početku sadrže samo nule. Moglo bi se reći da je ploča ustvari lista redova, a kolone pozicije u redovima. Ako isprintamo takvu ploču, red sa indexom 0 bi se nalazio na vrhu i nule sa indexom 0 bi se nalazile u koloni koja je najviše lijevo. U stvarnosti žetoni koje ubacujemo uvijek padaju dolje tj. prirodno je reći da je nulti (u svakodnevnom govoru prvi ali tu sve kreće od nule) red onaj koji se nalazi najviše dolje pa je zato ploča okrenuta naredbom iz biblioteke numpy – sada je nulti red dolje, nulta kolona je ostala lijevo.

Igra funkcionira tako da nule simboliziraju prazna mjesta, jedinice žetone jednog igrača te dvojke žetone drugog igrača. Odabirom kolone 1 ili 2 zamjeni 0 koja se nalazi u prvom slobodnom redu te ima indeks kolone koju je igrač odabrao.

Nakon svakog poteza program provjerava postoje li spojene četiri jedinice ili četiri dvojke u svim smjerovima, ako postoje igra je gotova te je taj igrač pobijedio.

Prikaz ploče kakvu ju računalo vidi (također prototip igre Spoji 4):

```
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]]
Player1 izaberite stupac (0-6):3
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0.]]
Player2 izaberite stupac (0-6):4
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 2. 0. 0.]]
```

3.23 DEFINICIJE ZA GRAFIČKI PRIKAZ

Zahvaljujući biblioteci pygame moguće je napraviti grafičku igru u Pythonu. Pygame dolazi sa svojim setom naredbi; uz pomoć njih možemo definirati veličinu igraćeg prozora te na određenim koordinatama nacrtati osnovne oblike na tu površinu, učitati sliku, ispisati tekst itd.

Definirala sam tri varijable (gam_over = True, game_exit = False i postavke = False) koje mi omogućavaju mijenjanje prikaza kojeg korisnik vidi. Program je aktivan sve dok je game_exit = False te ukoliko je gam_over = True korisnik će vidjeti početni ekran. Klik na ikonu ‘postavke’ pretvara postavke u ‘True’ te tada korisnik vidi izbornik za postavke, a klik na gumb ‘IGRAJ’ pokreće ‘ekran igre’ (iz 2.2) tj. pretvara gam_over u ‘False’.

Igraća ploča još uvijek funkcionira na isti način, ali je prikazana kao pravokutnik krugovima čija boja ovisi je li na određenom mjestu nula, jedan ili dva.

Dakle, program sve radi na ploči u obliku liste listi (iz 3.22), a ova skupina definicija samo poboljšava frontend programa.

3.24 DEFINICIJE ZA SINGLEPLAYER MODE

Eh, ovdje postaje kompliciranо. Ova skupina definicija isključivo radi na backendu programa tj. program sa njima odlučuje koji će potez napraviti s obzirom na poteze korisnika (u singleplayeru, naravno).

Program napravi kopiju ploče te u nju odigra potez u svaku moguću kolonu (tj. svaku koja ima još praznih mjesta) i svaki put izračuna 'vrijednost ploče' koja mu govori koliko je taj potez dobar. Na kraju će odigrati kolonu u kojoj je njegov žeton postigao najveću 'vrijednost ploče'.

3.241 VREDNOVANJE PLOČE

Ploča se podijeli na dijelove koji sadrže po 4 pozicije tj. na sve moguće načine u kojima je moguće 'spojiti 4'. Ti dijelovi se u ploči nalaze u svim smjerovima, no program ih bez obzira na njihov položaj pretvori u liste od 4 člana. Svaki od tih dijelova se zatim boduje tako da se metodom .count() pobroji koliko ima igračevih žetona i praznih mjesta u tom dijelu te s obzirom na to dobije određeni broj bodova. Bodovi svih dijelova se zbroje i zajedno čine vrijednost ploče za taj potez (iz 3.24).

3.242 MINIMAX ALGORITAM

Minimax (ponekad MinMax, MM) je pravilo odlučivanja koje se koristi u umjetnoj inteligenciji, teoriji odluka, teoriji igara, statistici i filozofiji za minimiziranje mogućeg gubitka za najgori slučaj. Kad se radi o dobitku, naziva se "maximin" - za maksimiziranje minimalnog dobitka. Više o ovom algoritmu možete saznati na https://en.wikipedia.org/wiki/Minimax.

Opći minimax algoritam bi izgledao ovako:

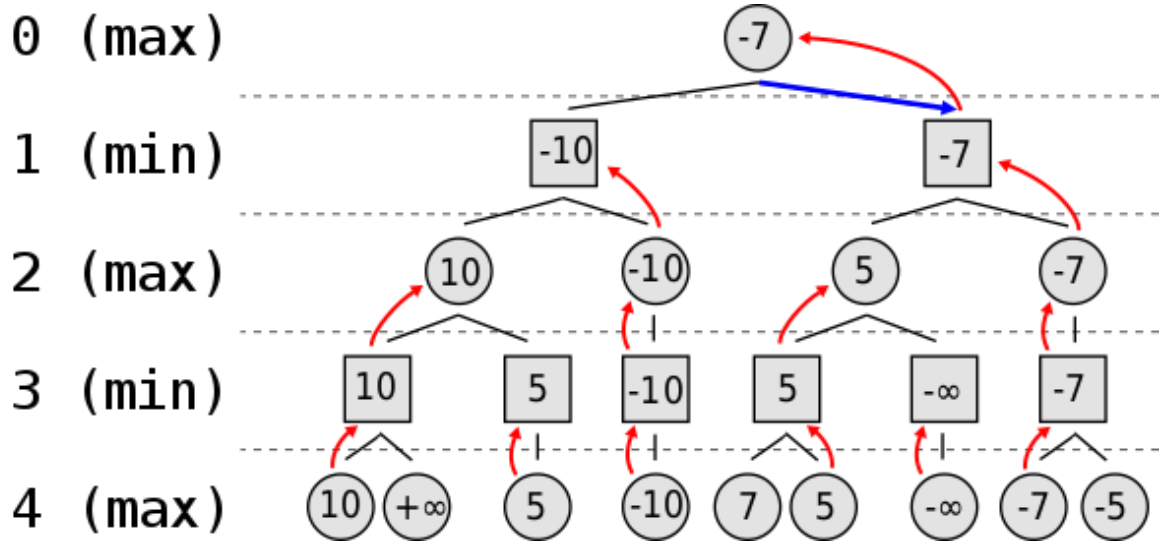
```
minimax(čvor, dubina, maksimiziranje_igrača):
    if dubina == 0 or čvor == terminalni čvor:
        return vrijednost čvora
    if maksimiziranje_igrača:
        vrijednost: = -∞
        for dijete in čvor:
            vrijednost: = max (vrijednost, minimax (dijete, dubina - 1, False))
        return vrijednost
    else (* minimiziranje igrača *):
        vrijednost: = + ∞
        for dijete in čvor:
            vrijednost: = min (vrijednost, minimax (dijete, dubina - 1, True))
        return vrijednost
```

Minimax funkcija ustvari poziva sama sebe dok ne dođe do kraja 'dubine' (tada je dubina == 0, ali uz to piše da čvor može biti 'terminalni čvor' što je ustvari kraj igre) tj. dok ne predvidi zadani broj poteza te uz pomoć uključivanja i isključivanja 'maksimiziranje_igrača' tj. maksimizacije vraća naizmjence najveće ili najmanje vrijednosti kako bi program odlučio koji potez će mu u budućnosti donijeti najbolji ishod.

Maksimizacija je uključena kada program predviđa svoj potez (jer kao što znamo najbolji potez nosi najviše bodova – iz 3.241), ali kada predviđa potez igrača maksimizacija je isključena tj. sada radi minimizaciju zato što igrač također želi igrati potez koji je najbolji za njega te istovremeno najlošiji za program što bi značilo da program može predvidjeti igračev potez

istim principom kao što bira svoj samo što neće uzeti onaj sa najvećom vrijednosti ploče već onaj sa najmanjom.

Pojednostavljeni prikaz rada minimax-a:



U igrici ‘Spoji 4’ primjenjuje se ovaj algoritam na način da ‘čvor’ (čvor je jako čudan prijevod ali nisam našla bolje, na engleskom je ‘node’) ustvari predstavlja cijelu ploču - pa su ‘djeca čvora’ onda svi mogući potezi tj. sve kolone u koje se potez može odigrati. U općem prikazu ova funkcija vraća samo ‘vrijednost’ što bi u našem slučaju bila samo vrijednost ploče koju želimo odabrati - ali ovom je programu potrebna informacija za koju kolonu će on postići tu krajnju vrijednost te zato moj primijenjen minimax algoritam vraća i kolonu za koju postiže tu vrijednost.

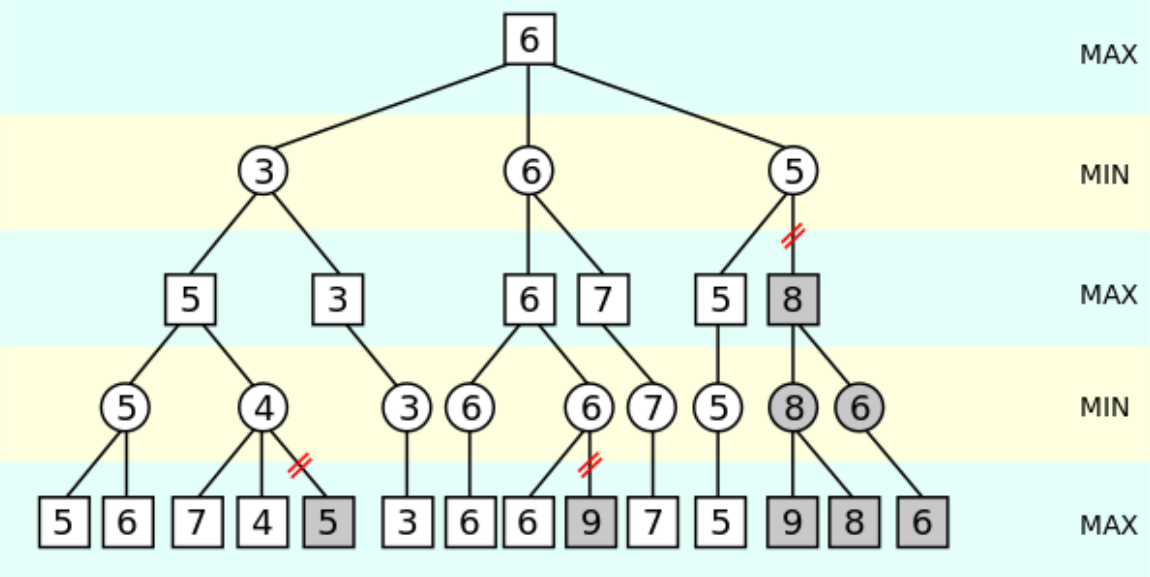
Kvaliteta ovakvog algoritma ovisi o vrijednostima koje se postave za vrijednost svakog dijela (iz 3.241), ali i o brzini donošenja odluka. Naime, kako postoji 7 mogućnosti program ima jako puno petlji kroz koje mora proći da bi donesao odluku pa sam pokušala sam ga ubrzati alpha-beta pruningom (više u 3.243).

Ovaj algoritam je daleko od savršenog te bi se još puno trebalo raditi na njemu da to postane. Također, postoji mogućnost da vam se program zamrzne ukoliko imate slabiji procesor ili ako u kodu promijenite dubinu minimax-a na veći broj jer je to naprosto previše podataka koje računalo treba obraditi.

3.243 ALPHA – BETA PRUNING

Alpha-beta pruning je algoritam pretraživanja koji nastoji smanjiti broj čvorova čija je vrijednost uzeta u obzir u minimax-ovom stablu pretraživanja. Ovo je ustvari dio minimax-a koji mu govori da nije potrebno vrednovati posljedice neke mogućnosti ako postoji neka

druga, ranije vrednovana mogućnost čija je vrijednost povoljnija od ove i time dokazuje da ta mogućnost ionako neće biti izabrana.



3.3 SISTEMATSKA KONFIGURACIJA

Naravno, potrebno je imati instaliran **Python** na računalu.

Potrebno je instalirati i biblioteke '**pygame**' i '**numpy**' jer se one ne nalaze u standardnoj biblioteci Pythona koja se instalira zajedno sa Pythonom. Najjednostavnije ih je instalirati tako da u vaš command prompt (cmd) i napišete 'pip install pygame' te 'pip install numpy'.

Iako novije verzije Pythona dolaze sa već instaliranim PIP-om (to je alat za upravljanje Python paketima), moguće je da nemate PIP te onda prvo njega trebate instalirati te vam u tome može pomoći ova stranica https://www.liquidweb.com/kb/install-pip-windows/.

Kako se ova igrica sastoji od Python koda uz koji su priložene neke slike (pogledaj 3.1) bitno je da se svi dijelovi projekta nalaze u istoj mapi.

Za pokretanje igrice dovoljno je ući u već spomenuti Python kod.