FAKULTET TEHNIČKIH NAUKA
UNIVERZITET U NOVOM SADU

Dokumentacija projekta
Cub Chase

- Distribuirani računarski sistemi u elektroenergetici -

Jovan Jovkić PR 31/2016

Jelena Pauković PR 90/2016

DIjana Atanasković PR 114/2016

Biljana Bajić PR 136/2015

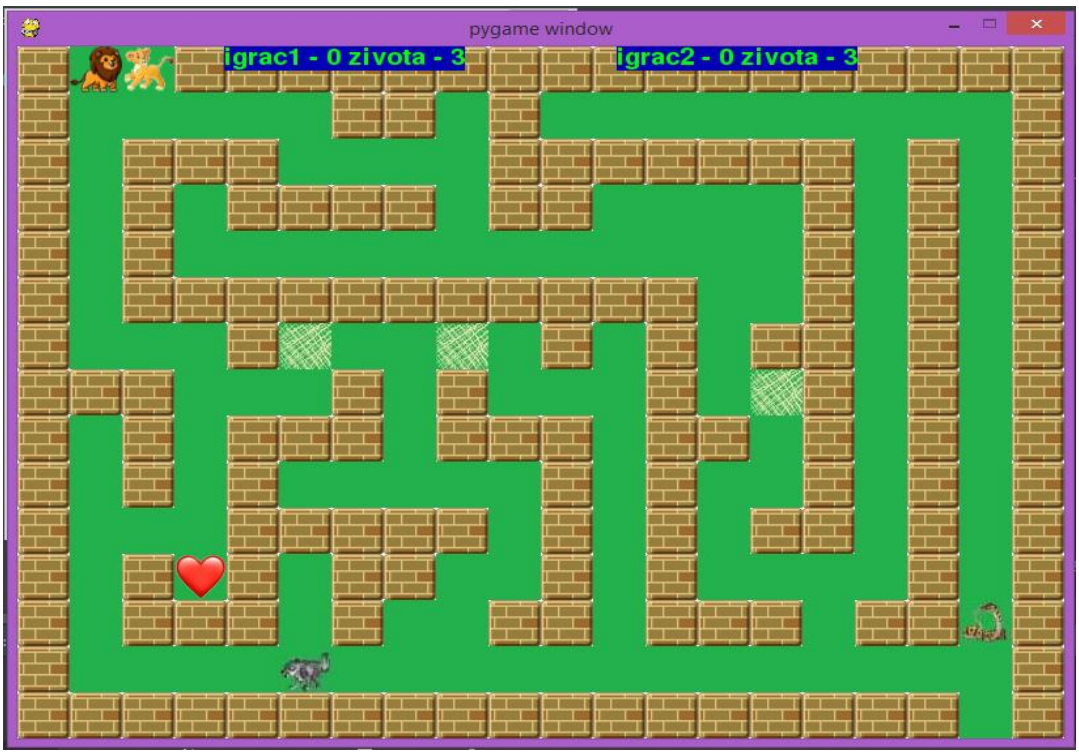
Uvod

Projekat CubChase predstavlja igricu napravljenu po uzoru na CubChase igru u programskom jeziku Python. Igricu je moguće igrati pojedinačno, a postoji mogućnost igranja sa 2 igrača. Originalna igrica takođe podržava igru jednog ili više igrača. Igrači se kreću po predefinisanoj mapi, izbegavaju čudovišta i aktiviraju zamke u cilju eliminisanja čudovišta. Nakon što igrač prođe svakom stazom u lavirintu prelazi se na sledeći nivo, gde se sva čudovišta ponovo generišu i povećava se težina igrice tako što se ubrza kretanje čudovišta. Igrica se završava kada jedan od igrača izgubi sva tri života. Pobednik je onaj koji prvi prođe kroz sve staze i ostavi svoje tragove. Životi se gube prilikom dodira sa čudovištem. S vremena na vreme u lavirintu se pojavljuje srce (dodatna sila) koja se random postavi na neko mesto u lavirintu, igrač kada pređe preko nje dobija dodatni život.

Pokretanjem same igrice otvara se startni prozor i pruža se mogućnost izbora između Start, Turnir i Plaz online. Startni prozor je urađen u skripti pr.py

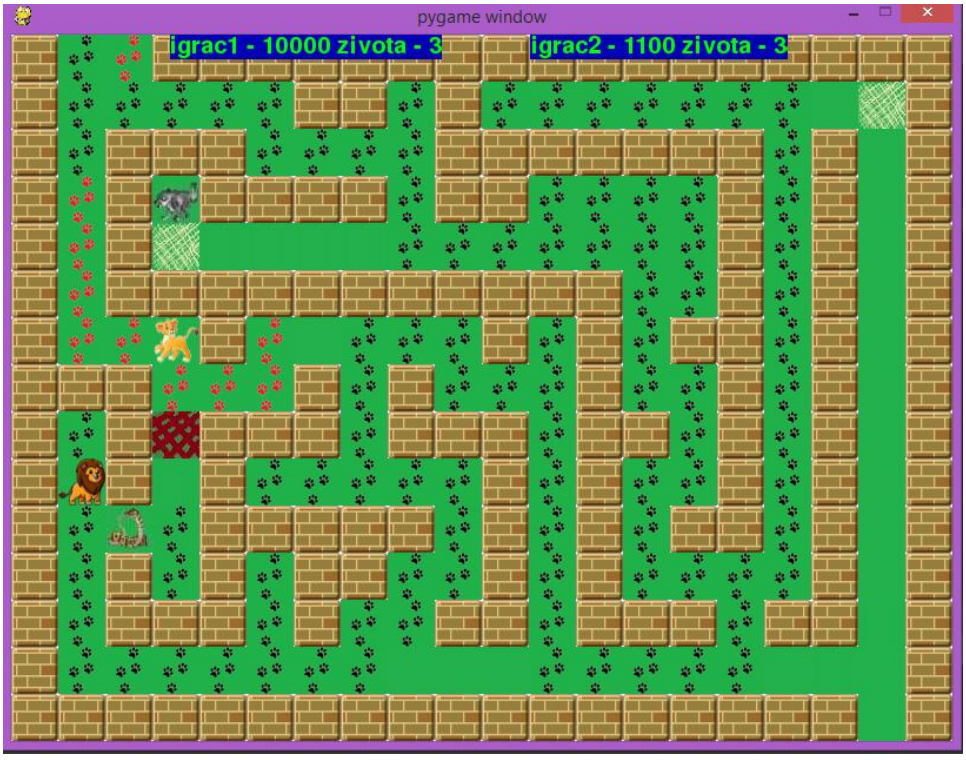
Startni prozor

Na sledećoj slici se vidi režim sa 2 igrača koji su trenutno na nivou 1 i dodatna sila (srce). Sila je implementirana u klasi App, koja pored inicijalizacije sadrzi metode draw_force,random_setup_force (na random nacin povecava ili smanjuje zivote), da_li_je_stao_na_srce, da_li_moze_da_iscrta_srce. Igrači i čudovista su implementirani u klasama IgracApp i Neprijatelj. Klasa IgracApp sadrzi inicijalizaciju (koordinate gde ce se pojaviti igrač) I sadezi metodu igrac_proces. Klasa Neprijatelj sadrzi listu slobodnih pozicija u matrici iz koje uzimamo random kordinate na koje se čudovista postavljaju svaki put pri pokretanju igrice Ili prelaskom na sledeći nivo. Sadrži metodu move_enemy (proverava po kojim koordinatama čudoviste moze da se krece).



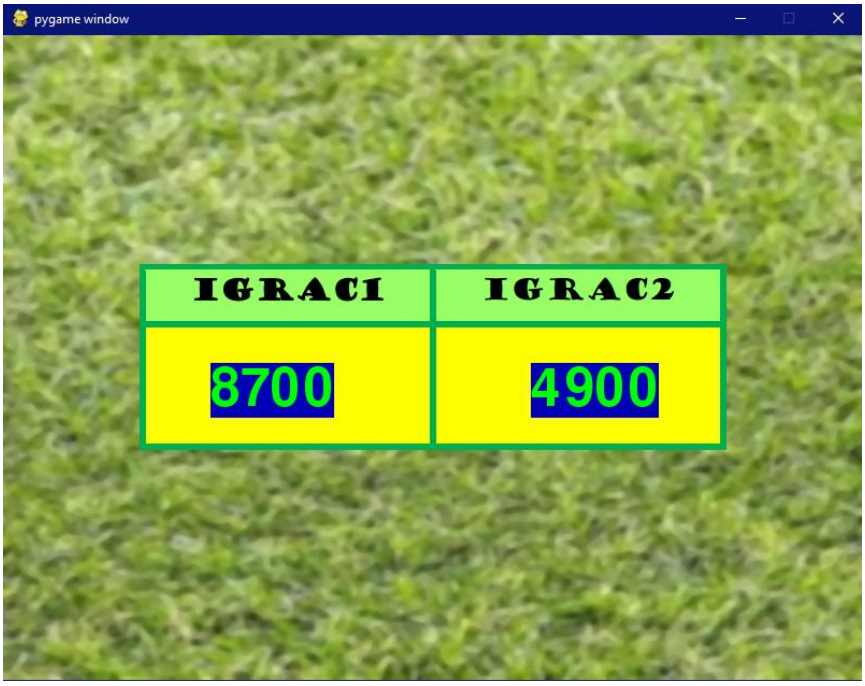
Multiplay režim, prvi nivo, prikaz dodatne sile

Na sledećoj slici prikazan je izgled zamke kada je aktivirana. Aktivirana zamka je crvene boje i implementirana je u klasi App.py.



Aktivna zamka

U slučaju da igrač oba igrača izgube sve zivote prikazaće se prozor na kome će biti ispisan broj osvojenih poena oba igrača.



Kraj igre

2. Razvoj aplikacije

Za razvoj ove igrice korišćen je Python programski jezik, Pygame framework i time, os, random i multipocess biblioteke za paralelizaciju rada.

Python programski jezik je interpretativni „visoki“ programski jezik koji podržava više tipova paradigmi, kao što su: objektno-orijentisana, imperativna, funkcionalana, proceduralna. Korišćenje Pythona nam je mnogo olakšao sam razvoj igrice jer automatski vodi računa o stvarima kao što su memory managment, o tipovima podataka, lako skaliranje aplikacija, podržava cross-platform programiranje, itd.

Loše strane Python-a su pre svega što je Python spor, pored toga nije baš najidealnije rešenje kod programiranja multi-processor/multi-core aplikacija, ima svoje limitacije kada je u pitanju pristup bazama podataka, gotovo je nemoguće napraviti kvalitetnu 3D grafiku, i nije najsjajnije rešenje prilikom rešavanja zadataka koji zahtevaju optimizaciju korišćenja memorije.

Framework koji je korišćen za izradu ove igrice je Pygame. Zamenio je PySDL i danas se koristi za izradu multimedijalnih aplikacija kao što su igre. Koristi SDL biblioteku koja omogućava razvoj igara u realnom vremenskom periodu bez korišćenja programskog jezika C. Aplikacije koje koriste pygame su vrlo prenosive i mogu raditi i na android telefonima i tabletima uz upotrebu pgsp4a (pygame subset for android).

Jedan od najvećih problema SDL-a je što ne podržava vektorsku matematiku,detekciju kolizije i upravljanje 2D i 3D grafičkim prikazom, MIDI suport ili manipulaciju nizom piksela.

Multiprocessing biblioteka, koja nam pruža kako lokalnu tako i distribuiranu konkurenciju. Zamenujući Global Interpreter Lock (mehanizam u Python-u koji

garantuje da samo jedan thread može istovremeno da izvršava bytecode) koristeći podprocese umesto thread-ova. Zahvaljujući ovom modulu (modul je fajl u Python-u sa ekstenzijom .py koji sadrži definicije i izraze) koji dozvoljava programeru da u potpunosti iskoristi procesore na datoj mašini i samim tim poralelizuje izvršavanje zadataka. Biblioteka se koristi i na Windows i na Unix platformama.

Problem koji se javlja prilikom korišćenja multiprocess biblioteke umesto thread-ova je što se koriste procesi koji svaki ima zasebni memorijski prostor što otežava deljenje objekata i resursa između procesa, dok korištenjem thread-ova koristimo deljeni memorijski prostor i ovaj problem ne postoji. Kod thread-ova moramo zato koristiti neke od mehanizama zaključavanja kako ne bi više niti pisalo na istu memorijsku lokaciju, što bi dovelo do nekonzistentnog stanja prilikom izvršavanja aplikacije.

Zaključak

Razvijajući ovu igricu imali samo priliku da se susretnemo sa nekim novim konceptima koje Python omogućava. Korištenje programskog jezika koji do sada nije viđen, kao i potpuno novi framework i biblioteke doprinele su da ovaj zadatak bude izazovan. Korišćeno je paralelno programiranje i projekat je razdvojen na više komponenti. Paralelizacija je demonstrirana preko thread-ova i proces-a koji izvršavaju određene module, a pored toga Pygame već ima ugrađene mehanizme za paralelnu obradu modula, te je kroz ovaj projekat programski upotreba niti i procesa demonstrirana pri izvršavanju samo određenih delova aplikacije.