Univerzitet u Beogradu
Elektrotehnički fakultet

Funkcionalno programiranje
13M111FP
2022/2023.

Projekat

Napisati program na programskom jeziku Scala za realizaciju jedne varijante igre Sokoban. Radi lakšeg razumevanja, implementacija igre može se pogledati na sokoban.info[1]. Igra se zasniva na kretanju junaka po skladištu (na japanskom „sokoban“ znači „čuvar skladišta“), sa ciljem da se sve kutije guranjem smeste na neko od završnih polja (tačaka). Teren za igru se sastoji iz ploča po kojima se junak kreće pomoću četiri komande (gore, dole, levo, desno) i zidova kroz koje nije moguće kretanje ni guranje kutija. Igra se završava kada su sve kutije na ciljnim pozicijama.

Za maksimalan broj poena, program treba da ispuni sve uslove projektnog zadatka i da radi pouzdano.

0. [10p] Korisnički interfejs treba da bude intuitivan i dovoljno detaljan. Korisnik sa programom intereaguje putem jednostavnog menija. Program treba da ispiše sadržaj menija, a zatim čeka da korisnik izabere redni broj neke od ponuđenih stavki, nakon čega, pre izvršenja, od korisnika očekuje da po potrebi unese dodatne parametre. Program treba da korisniku omogući sledeće manipulacije:

1. [20p] Osnovne funkcionalnosti – učitavanje, odigravanje i čuvanje igre
   - Učitavanje mape terena iz fajla. Format fajla definisati prema primeru fajla sa zapisom mape terena iz Dodatka 1. Mapa terena je matrica koja može da sadrži sledeće simbole:
     - simbol “-” koji predstavlja običnu ploču;
     - simbol “#” koji označava zid kroz koji nije moguće kretanje ni guranje kutija;
     - simbol “S” koji označava poziciju junaka;
     - simbol “.” koji označava jednu od krajnjih pozicija;
     - simbol “X” koji označava kutiju koja nije pozicionirana na završnu poziciju, već stoji na običnoj ploči.
     - simbol “O” koji označava kutiju koja je pozicionira na završnu poziciju.
   - Započinjanje nove igre izborom neke od raspoloživih mapa.
   - Odigravanje poteza. Odigravanje predstavlja pomeranje junaka u zavisnosti od poteza koji korisnik zada (dole, gore, levo, desno), uz prikaz stanja igre nakon odigravanja poteza. Ukoliko je nakon odigravanja svaka od kutija postavljena na završnu poziciju, igra se završava pobedom.
   - Poništavanje jednog ili više prethodnih poteza („undo“ funkcionalnost proizvoljne dubine).
   - Odigravanje sekvence poteza učitavanjem iz fajla. Sekvenca poteza sadrži redom oznake poteza, svaki u zasebnom redu. Oznake poteza su:
     - dole: “D”
     - gore: “U”
     - levo: “L”
     - desno: “R”

[1] Pristupano 10.5.2023. godine.

2. [35p] Kreiranje novih mapa na osnovu postojećih. Operacije za kreiranje mapa su sledeće:
   - Proširivanje mape redom ili kolonom (dodaju se kao prvi ili kao poslednji red/kolona).
   - Uklanjanje prve ili poslednje kolone ili reda iz mape.
   - Zamena jednog tipa polja na zadatoj poziciji drugim tipom polja (postojeće tipove polja videti u jednoj od prethodnih stavki).
   - Predvideti sledeće predefinisane operacije:
     - Inverzija: rezultujuća mapa dobija se tako što se na originalnoj mapi zamene pozicije kutija i završnih tačaka.
     - Minimizacija zida: rezultujuća mapa dobija se tako što se na originalnoj mapi brišu zidovi koji nisu deo spoljnog omotača skladišta, tako da na novodobijenoj mapi skladište okružuje zid debljine 1 i na njoj nema drugih zidova.
     - Filtriranje: ukoliko je makar jedan od suseda date ploče na vertikalnoj ili horizontalnoj razdaljini manjoj ili jednakoj N zid, zameniti datu ploču običnom pločom.
     - Fraktalizacija: izabrano polje zida se menja običnom pločom, pa se zatim dodaju polja zida tako da on bude minimalan zatvoren, a da je ciljano polje sada unutar njega. Ovo potencijalno zahteva proširivanje mape redom ili kolonom.
   - Pravljenje proizvoljne imenovane kompozitne operacije ulančavanjem osnovnih (prostih ili kompozitnih). Rezultat je funkcija, dakle voditi računa da se eventualne provere domena i ulaznih podataka svih koraka koji se komponuju moraju obaviti na početku tela kompozicije.
   - Formiranje imenovane sekvence operacija, koja se definiše kao operacija koja se sastoji od liste operacija. Izvršavanje ovakve operacije se razlikuje od izvršavanja kompozicije liste operacija po tome što je eventualne greške pri izvršavanju nekog od koraka potrebno obraditi tek kada se dođe do izvršavanja tog koraka. Npr. ako se imenovana operacija f sastoji od koraka a, b i c, može se desiti da su ulazni parametri za korake a i b korektni, a za korak c nisu. U tom slučaju, izvršiće se koraci a i b, pa se pri izvršavanju koraka c prijaviti greška.
   - Provera validnosti mape, koja podrazumeva proveru da li je spoljašnji zid skladišta kompletan (da li formira zatvoren oblik, tako da ne može da se izađe iz skladišta), da li je broj kutija i završnih tačaka jednak i sl. Ova funkcionalnost ne proverava da li je nivo rešiv.

3. [15p] Ispis rešenja igre u vidu sekvence poteza koji dovode sve kutije iz početnih u krajnje pozicije. Sekvencu rezultujućih poteza upisati u fajl. Sekvenca poteza sadrži redom oznake poteza, svaki u zasebnom redu, po istom formatu kao u okviru jedne od prethodnih stavki. Dovoljno je prikazati jedno od mogućih rešenja. Za maksimalni broj poena, implementirani rešavač mora poštovati principe funkcionalnog programiranja, dakle biti implementiran kao terminalna rekurzija bez *var* promenljivih. Rešavač treba da radi za mape dimenzija do 15x15.

Za preostalih 20 poena potrebno je implementirati jedan od zahteva 4 i 5

4. [20p] Realizovati grafički korisnički interfejs za igru. Studentu je dozvoljeno korišćenje biblioteka za pravljenje grafičkog korisničkog interfejsa po izboru.
5. [20p] Sprovesti testiranje korišćenjem biblioteke ScalaTest. Potrebno je implementirati svite testova koje ispituju i demonstriraju ispravnost rada svake od implementiranih funkcionalnosti. Svaka svita testova treba da sadrži barem 3 jedinična (Unit) testa.

Program napisati tako da se koriste programski konstrukti svojstveni funkcionalnim jezicima, koji su obrađeni tokom nastave na predmetu. Gde je primereno, koristiti funkcije višeg reda, parcijalno primenjene funkcije, odnosno funkcije u kerifikovanom obliku.

Za sve nedovoljno precizne zahteve od studenata se očekuje da usvoje razumne pretpostavke i da ih dosledno primenjuju u rešenju.

Dodatak 1: primer mape terena i slike odgovarajućeg nivoa (nivo Boxxle 1 sa priložene veb stranice)

```
-#####---
-#---####
-#---#--#
-##----.#
###-###.#
#-x-#-#.#
#-xx#-###
#S--#----
#####----
```