PROGRAMSKI JEZICI 2 (2246) – PISMENI ISPIT (05.07.2018.)

1. **(30)** Napisati aplikaciju simulacije prelaska graničnog prelaza. Granični prelaz mogu da pređu pješaci, automobili, autobusi i kamioni. Pod graničnim prelazom se smatra samo jedna strana prelaza između država. Svi koji prelaze moraju da prođu policijsku i carinsku kontrolu respektivno. Prelaz ima 2 terminala: terminal za kamione i terminal za autobuse, vozila i pješake. Svaki kamion u sebi posjeduje robu slučano odabrane ukupne vrijednosti i za istu se mora obračunati carinska stopa u iznosu od 15%. Vozila, automobili i pješaci (tačnije osobe u istim ili pješak kao osoba) mogu a i ne moraju da imaju robu koju mogu da prijave. U vozilu se nalazi od 1 do 5 osoba, u kamionu 1 ili 2 osobe, a u autobusu od 1 do 52 osobe. 60% osoba koje prelaze granicu posjeduju pasoš. One osobe koje ne posjeduju pasoš ne mogu preći granicu i moraju napustiti vozilo. Prilikom dolaska na carinski terminal osim kamiona svaka od osoba prijavljuje ili ne prijavljuje robu (ako istu posjeduje za šta je vjerovatnoća 50% da će roba biti prijavljena) te carinski terminal slučajnim odabirom (u 50% slučajeva) provjerava one koji su izjavili da nemaju robu. U slučaju da roba postoji i prijavljena je, za istu se obračunava carinska stopa u iznosu 10%. U slučaju da je pronađena roba koja nije prijavljena na istu se obračunava carinska stopa u iznosu 10% kao i kazna u izosu vrijednosti robe. Klase u vezi vozila i osoba se nalaze u jednom modulu a preostale klase u drugom modulu. Simulacija rada carinskog terminala sastoji se od sljedećeg:
    - Kreira se 20 objekta svih koji mogu preći granicu (kreiranje podrazumijeva automatsko generisanje imena vozila ili imena osobe, a za vozila slučajno generisane osobe u prethodno definisanim opsezima).
    - Nakon kreiranja, kamioni se postavljaju u red na carinskom terminalu za kamione, a ostali u prioritetni red gdje prioritet imaju pješaci u odnosu autobuse ili vozila.
    - Prioritetni red se serijalizuje u fajl prioritetniRed.*ser,* a red kamiona se serijalizuje u fajl *kamioni.ser*.
    - Korisničkim unosom „START“ sa tastature carinski prelaz počinje sa radom. Jedna nit predstavlja „pregled“ stanja na terminalu za kamione, dok druga nit upravlja pregledom terminalom na kojem su svi ostali. Zadržavanje na kontroli pasoša iznosi 1 sekundu a na carinskoj kontroli 2 sekunde po osobi. Obrađeni se tom prilikom uklanjaju iz reda. Prilikom obrade vozila na konzolu se ispisuju podaci o vozilu/osobi kao i osobama u vozilu, da li su prijavili carinu te da li je obračunata carina ili kazna.
    - Na kraju simulacije ispisuju se osobe koje nisu posjedovale pasoš ili nisu izvršile carinsku prijavu.

2. **(20)** Napisati klasu *Stablo* koja predstavlja implementaciju binarnog stabla pretrage koje radi sa cjelobrojnim vrijednostima. Ova klasa ima tri atributa: *vrijednost, nodeLeft i nodeRight*, gdje *vrijednost* predstavlja cjeli broj, a *nodeLeft i nodeRight* predstavljaju referencu na lijevo i desno dijete čvora stabla (vrijednost je *null* ako nema djeteta). Klasa posjeduje jednu metodu *dodaj*, za smještanje elemenata u binarno stablo pretrage. Ako su vrijednosti koje se dodaju iste potrebno je baciti korisnički definisan *DodavanjeException*. Neophodno je obezbijediti sinhronizovan pristup binarnom stablu. *AddThread* je nit zadužena za stavljanje elementa u binarno stablo. *ObserveThread* je nit koja prolazi kroz binarno stablo pretrage u inorder redosljedu obilaska,

ispisuje sadržaj i zadržava se na svakom čvoru 2 sekunde. Nit *AddThread* ima 2 atributa: binarno stablo (tipa Binarno stablo) i number (tipa int), gdje *binarno stablo* predstavlja binarno stablo sa kojom nit radi, a *number* broj elemenata koje ta nit treba dodati u binarno stablo pretrage. Nit *ObserveThread* posjeduje samo binarno stablo pretrage. Napisati klasu *StabloMain* u čijoj će main metodi biti instancirano binarno stablo, te niti *AddThread* i *ObserveThread*. Nit *ObserveThread je* potrebno instancirati nakon 20 sekundi iz razloga da se dodaju određeni čvorovi u stablo prije pozivanja iste. Elementi koji se smještaju u binarno stablo pretrage generišu se proizvoljno. Simulacija traje tri minuta.

3. **(20)** Napisati konzolnu aplikaciju pogađanje u metu gdje korisnik pritiskom na tipku space pogađa metu slučajnim odabirom na određenu poziciju. Izgled mete je sljedeći:

b	b	b	b	b
b	y	y	y	b
b	y	c	y	b
b	y	y	y	b
b	b	b	b	b

Crveno polje donosi 100 bodova, zuto polje 25 a plavo 10. Na standardnom izlazu ispisati matricu u kojoj će umjesto boje biti početno slovo boje, a ako je meta negdje pogođena umjesto slova boje karakter X. Korisnik ima 5 pokušaja da pogađa metu. Za svaki od pokušaja vjerovatnoća je 30% da će promašiti metu i tada se korisnku oduzima 5 bodova. Nakon završetka igre stanje mete se čuva u tekstualni fajl pri čemu se u prvom redu zapisuje broj bodova posljednjeg takmičara a u novom redu broj pogađanja u svako od polja (neko polje može biti pogođeno više puta i ako ništa nije pogođeno sve vrijednosti će biti postavljene na 0). Igra ima dva moda:

- START – započinje se nova igra
- SHOW – prikazuju se podaci iz fajla za poslednjeg igrača koji je igrao

Napomena: Vrijeme trajanja ispita je 180 minuta. Nakon završenog ispita, zadatke je potrebno *upload*-ovati na *Moodle*, arhivirane u formatu ***broj_indeksa_ime_i_prezime***.