PROGRAMSKI JEZICI 2 (2246) – PISMENI ISPIT (14.06.2018.)

1. **(30)** Napisati aplikaciju simulacije usluga u hotelu. Gosti mogu da se podjele na goste koji borave o svom trošku i goste koji borave o trošku firme. Gosti koji borave o svom trošku mogu da imaju pristup bazenu i restoranu. Gosti koji borave o trošku firme dodatno imaju ekskluzivno pravo pristupa kongresnim salama u kojima su brojni dodatni sadržaji za poslovanje. Obični gosti mogu da koriste kongresne sale sa 50% većom cijenom. Gosti koji borave o trošku firme imaju i dodatne podatke o firmi u čije ime gostuju kao i ukupan novac koji im je na raspolaganju za potrošnju(Double). Gosti koji borave o svom trošku imaju dodatni atribut novac(Double). Usluga se predstavlja kao HashMap, gdje je *key* tipa String i može da bude: „Dodatni obrok“ , „Koktel“, „Korištenje projektora“, „Korištenje računara“ i „Korištenje džakuzi bazena“ , dok *value* predstavlja slučajno generisan cijeli broj u opsegu od 10 do 100. Gostima se jedna od gore navedenih dodatnih usluga podešava na slučajno odabranu vrijednost usluge prilikom kreiranja.

   Simulacija rada hotela sastoji se od sljedećeg:

   - Kreira se automatski po 10 objekta gostiju sa koji borave o svom trošku i o trošku firme (automatsko kreiranje podrazumijeva automatsko generisanje imena, prezimena i godine rođenja u proizvoljnom obliku i slučajnog odabira dodatne usluge).
   - Nakon kreiranja, slučajno se bira 5 osoba koje će biti primljene u restoran, 5 osoba koje će biti primljene u kongresnu salu i ostale osobe se primaju na bazen.
   - Osobe koje se primaju u kongresnu salu smještaju se u prioritetni red u kom se sortiraju po tipu gosta, prvo gosti iz firmi a onda gosti o svom trošku. Osobe koje su primljene u restoran i bazen smještaju se u proizvoljno odabranu kolekciju.
   - Prioritetni red osoba se serijalizuje u fajl ulazakUSalu.*ser*.
   - Korisničkim unosom „START“ sa tastature hotel počinje sa radom. Jedna nit predstavlja „pregled“ osoba koje čekaju na ulazak u salu, dok druga nit upravlja pregledom ostalih osoba, po principu da se osobe u restoranu zadržavaju 5 sekundi, a na bazenu 10. Pod ulaskom u salu podrazumijeva se uklanjanje osobe iz reda uz pauzu između ulazaka od 3 sekunde. Prilikom obrade osobe na konzolu se ispisuju njihovi podaci, razlog zbog kog su došli i koliku cijenu usluge su platili. Prilikom plaćanja usluge umanjuje se novac koji osoba posjeduje.
   - Na kraju simulacije ispisuju se osobe kojima je stanje novca na računu ostalo pozitivno.

2. **(20)** Napisati klasu *Lista* koja predstavlja implementaciju liste koji radi sa generičkim elementima. Ova klasa ima dva atributa: *vrijednost* i *referenceNext*, gdje *vrijednost* predstavlja generički element, a *referenceNext* predstavlja referencu na naredni element liste (vrijednost je *null* za poslednji član liste). Klasa posjeduje dvije metode *dodaj* i *obriši*, za smještanje i skidanje elemenata sa kraja liste, respektivno. Elementi liste se mogu brisati ako ista nije prazna. U slučaj prazne liste potrebno je baciti korisnički-implementiran *ListException*. Neophodno je obezbijediti sinhronizovan pristup listi. *AddThread* i *RemoveThread* su niti zadužene za stavljanje i skidanje elemenata sa liste, respektivno. Ove klase imaju po 2 atributa: lista (tipa Lista) i number (tipa int), gdje *lista* predstavlja listu l sa kojim niti rade, a *number* broj elemenata koje *AddThread* i *RemoveThread* niti trebaju

dodati u listu, odnosno obrisati iz liste, respektivno. Napisati klasu *ListMain* u čijoj će main metodi biti instancirana lista, te niti *AddThread* i *RemoveThread* koje će u listu dodavati i iz liste brisati zadati broj elemenata. Elementi koji se smještaju u listu generišu se proizvoljno. Simulacija punjenja/pražnjenja liste traje dva minuta, pri čemu se punjenje i pražnjenje pokreću naizmjenično. U slučaju da po završetku simulacije lista nije prazana, objekat liste je potrebno serijalizovati i smjestiti u fajl *lista.ser*. Na početku svake simulacije potrebno je provjeriti da li postoji navedeni fajl i u slučaju da postoji, deserijalizovati listu i nju koristiti u simulaciji.

3. **(20)** Napisati aplikaciju pretraga direktorijuma koja za datu riječ vrši pretragu iste u tekstualnim datotekama u datom direktorijumu i njegovim poddirektorijumima. Aplikacija bi nakon pretrage trebala da ispiše listu sa putanjom do fajlova u kojima je navedena data riječ kao i broj pojavljivanja te riječi za svaki od fajlova. Korisnik putem argumenata komandne linije unosi putanju od koje počinje pretraga kao i ključnu riječ za pretragu. Rezultati pretrage se ispisuju na konzolu.

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***.