Univerzitet u Beogradu
Matematički fakultet

Datum: 28.12.2025.
Ispitni rok: Decembar PS

Računarske mreže (modul Računarstvo i informatika)
- praktični ispit -

Tehničke napomene

Na radnoj površini nalazi se šifrovana zip arhiva sa nazivom `mreze.r.dec.zip`. Unutar te arhive nalazi se direktorijum sa nazivom `rmr_dec_ImePrezime_mrGGXXX` (gde `mrGGXXX` predstavlja korisničko ime Vaše Alas mejl adrese). U ovom direktorijumu nalazi se validan IntelliJ projekat (koji predstavlja Vaš rad) sa paketima `file_reader`, `chess` i `restaurant`. Ovaj direktorijum izvući iz arhive na Desktop i preimenovati ga u skladu sa Vašim podacima. Otvoriti IntelliJ IDEA, izabrati opciju `Open project` (ne `Import project`!) i otvoriti pomenuti direktorijum. **Kodovi koji se ne prevode se neće pregledati**. Vreme za izradu ispita je **3 sata**.

Srećan rad!

Ispitni zadatak 1: URL File Reader (10 poena)

Napisati Java aplikaciju koja korišćenjem URL Java API-ja (klase `URL` i `URLConnection`) čita fajlove koristeći `file` protokol. Korisnik na standardni ulaz redom zadaje apsolutne putanje do fajlova (apsolutna putanja za tekući direktorijum se može dobiti komandom `pwd` u terminalu). Obavezno je obraditi greške nastale usled pogrešnog ili nevalidnog unosa.

```
file.txt:
Ovo je neki fajl.
Citam ga pomocu klasa URL i URLConnection

> /usr/oop/Desktop/file.txt
Otvaram fajl pomocu "file" protokola:
    Ovo je neki fajl.
    Citam ga pomocu klasa URL i URLConnection
```

Naredni zadatak se nalazi na sledećoj strani!

Ispitni zadatak 2: Chess Scoreboard (30 poena)

Implementirati TCP client-server aplikaciju za dodavanje, ažuriranje i čuvanje informacije o šahistima i njihovim Elo rejtinzima. Serverski deo aplikacije održava *in-memory* tabelu šahista i njihove trenutne rejtinge. Tabela ima kolone `id (int)`, `naziv (String)` i `elo (int)`.

U početku, klijent se povezuje na server, koji pokreće posebnu nit koja obradjuje tog klijenta, nakon čega server nastavlja da prihvata zahteve od drugih klijenata. Nakon što se klijent konektuje na server, prosledjuje server zahteve oblika:

- `select <id>` – `<id>` je tipa `int`
- `insert <naziv>` – `<naziv>` je tipa `String`
- `update <id> <elo>` – `<id>` i `<elo>` su tipa `int`

sve dok klijent ne unese i pošalje serveru `bye`. Nakon dobijana odgovora od servera, klijent odgovor ispisuje na standardni izlaz (osim kada klijent unese `bye` – tada klijent završava sa radom, nakon poslatog paketa serveru).

Serverska nit, koja obradjuje jednog klijenta, u početku na standardni izlaz (sa serverske strane) ispisuje poruku o pristiglom klijentu u formatu `client accepted <ip>:<port>`. Nakon toga, nit čeka na zahteve klijenata i odgovara na njih na sledeći način:

- `select <id>` – vraća `naziv` i `elo` rejting šahiste sa datim identifikatorom `id`
- `insert <naziv>` – ubacuje u tabelu šahistu sa datim imenom, dodeljujući mu jedinstveni identifikator i `elo` u vrednosti 1300 (što ujedno i predstavlja najmanju moguću vrednost za `elo`) i vraća poruku o uspešnosti operacije
- `update <id> <elo>` – vrši izmenu `elo` vrednosti šahiste sa identifikatorom `id` i vraća poruku o uspešnosti operacije

U slučaju da klijent pošalje nevalidan zahtev, obraditi grešku i vratiti poruku o grešci. *Potrebno je obezbediti da se u slučaju konfliktnih situacija (npr. dva klijenta žele da promene vrednost istoj osobi) zahtevi pravilno obradjuju.*

```
Primer rada aplikacije:
> insert Magnus Carlsen
insert je uspesno izvrsen
> select 1
Magnus Carlsen : 1300
> update 1 1330
update je uspesno izvrsen
> select 1
Magnus Carlsen : 1330
> insert Fabiano Caruana
insert je uspesno izvrsen
> update 2 1290
elo vrednost iznosi barem 1300
> select 2
Fabiano Caruana : 1300
> bye
```

Server pokrenuti na portu `5000`. Server prestaje sa radom prosledjivanjem `SIGINT` (`[CTRL+C]`) signala (odnosno, nije potrebno posebno implementirati zaustavljanje serverskog dela aplikacije). **Obavezno je pravilno zatvoriti i osloboditi sve korišćene resurse**.

Naredni zadatak se nalazi na sledećoj strani!

Ispitni zadatak 3: Restaurant MATF (20 poena)

Implementirati UDP *client-server* aplikaciju koja omogućava klijentima da rezervišu obrok u restoranu MATF. Koristiti **Java Datagram API** za implementaciju klijentskog i serverskog dela aplikacije.

Nakon pokretanja servera, on odgovara na zahteve klijenata oblika `<ime> <vreme>`, gde `<ime>` predstavlja ime klijenta na kojeg se vodi rezervacija, kao i vreme koje bi osoba htela da rezerviše. Server vodi evidenciju svih uspešnih rezervacija i na osnovu te evidencije odgovara klijentu da li je moguće rezervisati mesto u restoranu ili ne. Ako u restoranu ne postoji rezervacija za dato vreme, server odgovara klijentu da je uspešno rezervisao obrok u restoranu i dodaje tu informaciju u svoju evidenciju, dok se u suprotnom klijentu javlja da rezervaciju nije moguće izvršiti. Nakon dobijenog od odgovora od servera, klijent završava sa radom, dok server nastavlja da odgovara na zahteve drugih klijenata.

```
Primer rada aplikacije:
    client 1: Stefan 18:30
    server: Uspesno ste rezervisali mesto za "Stefan" u 18:30
    client 2: Boban 20:30
    server: Uspesno ste rezervisali mesto za "Boban" u 22:30
    client 3: Nikola 18:30
    server: Vec postoji rezervacija od strane "Stefan" u to vreme, molimo pokusajte kasnije!
```

Server pokrenuti na portu `6000`. Server prestaje sa radom prosledjivanjem `SIGINT` (`[CTRL+C]`) signala (odnosno, nije potrebno posebno implementirati zaustavljanje serverskog dela aplikacije). **Postarati se da se sve greške pravilno obradjuju**. **Obavezno je pravilno zatvoriti i osloboditi sve korišćene resurse**.