JMBAG

IME & PREZIME

# Mreže računala

Prvi kolokvij, 24. studenoga 2023. godine

Na kolokviju je dozvoljeno koristiti samo pribor za pisanje i službeni šalabahter. Predajete samo papire koje ste dobili. Kolokvij ima ukupno **50 bodova**, međutim konačni broj bodova se računa kao `min(40, #bodova)`.

## PRVI ZADATAK

OSTVARENI BODOVI 8

Napišite dio serverskog koda od trenutka prihvaćanja konekcije do trenutka prekidanja konekcije (uključivo) koji

- od klijenta prima string koji je IP adresa nekog računala u dekadskom obliku,
- šalje klijentu službeni *hostname* tog računala te broj alternativnih *hostnameova* tog računala.

Nakon toga server prekida konekciju. Možete pretpostaviti da se svaki broj i svaki string mogu poslati/primiti jednom *send*/*recv* naredbom.

JMBAG

IME & PREZIME

## DRUGI ZADATAK

OSTVARENI BODOVI | | 12 |

U ovom zadatku se **ne očekuje** pisanje programskog koda.

Tvrtka *Mijo Bus* bavi se iznajmljivanjem autobusa (naravno, uz svaki autobus dolazi i kvalificirani vozač). Za svaki autobus imamo naziv modela (npr. *Setra 515*), registracijsku oznaku (npr. `VŽ314LL`), broj sjedećih i broj stajaćih mjesta te dodatne sadržaje (npr. *TV,WC,klima*). Svaki je autobus jedinstveno određen svojom registracijskom oznakom. Autobus se rezervira počevši od navedenog vremena određenog datuma na određen broj sati (npr. od 15:00h 28.12.2023 na 20 sati) te se temeljem tog broja sati određuje cijena najma (u eurima, npr. 756,00 eura - naravno ta cijena ovisi i o modelu autobusa).

Potrebno je dizajnirati protokol za mrežnu aplikaciju koja služi za najam autobusa. Klijent mora moći:

1. Za dano vrijeme i datum početka rezervacije te broj sati na koji se autobus unajmljuje, dobiti popis modela autobusa koji su raspoloživi u tom terminu zajedno s ukupnom cijenom najma pojedinog raspoloživog modela autobusa za to vrijeme.
2. Kao u 1. ali s time da klijent još navodi i broj putnika za koje želi unajmiti autobus (pritom ako je broj putnika, primjerice, 20 tada u obzir dolaze svi raspoloživi autobusi u tom vremenu koji imaju 20 ili više mjesta ukupno (stajaća + sjedeća mjesta)).
3. Za svaki od modela autobusa koji tvrtka posjeduje dobiti više informacija, tj. klijent mora moći saznati sljedeće (ne sve istovremeno!):
   - popis svih registracijskih oznaka autobusa tog modela koje tvrtka posjeduje,
   - cijena jednog sata najma autobusa tog modela,
   - broj sjedećih i broj stajaćih mjesta autobusa tog modela,
   - popis dodatnih sadržaja autobusa tog modela.
4. Rezervirati autobus dane registracije od danog vremena dani datum na određen broj sati. Klijentu treba javiti sljedeće povratne informacije: je li taj autobus raspoloživ za najam, ukupnu cijenu tog najma te poziv na broj na koji klijent treba uplatiti dani iznos u roku 24 sata od kreiranja ove rezervacije (poziv na broj: 604060-404-registracijska oznaka odabranog autobusa).
5. Svaka uspješno napravljena rezervacija određena je jedinstvenim kodom (koji se sastoji od 10 znakova). Klijent mora moći provjeriti stanje te rezervacije (stanja su: potreban iznos nije još plaćen, autobus uspješno unajmljen i rezervacija istekla zbog neplaćanja) i saznati koliko je vremena (u minutama) preostalo za plaćanje ove rezervacije kako bi najam bio uspješan.
6. Završiti komunikaciju.

Osmislite vrste poruka i njihov format (zaglavlje/*header* i tijelo/*payload*) koje razmjenjuju klijent i server. 5 + 1 = 6 bodova

Navedite kakvi se sve tipovi grešaka mogu dogoditi u komunikaciji (npr. klijent želi rezervirati autobus s nepostojećom registracijom). 3 boda

Odaberite primjer po jedne klijentske poruke za svaku točku pod 3. i svaku od preostalih 5 funkcionalnosti (primjeri u skladu s vašim definicijama mogućih vrsta i formata poruka) te pripadne odgovore od strane servera. 3 boda

JMBAG IME & PREZIME

## TREĆI ZADATAK

OSTVARENI BODOVI 17

Tvrtka *Računala i dr.* posjeduje mrežnu aplikaciju pomoću koje djelatnici u nekoj od poslovnica te tvrtke mogu pratiti raspoloživost pojedinog artikla po ostalim poslovnicama te tražiti rezervaciju određene količine artikala u nekoj poslovnici (kako bi se kasnije ta količina artikla preuzela od nekog kupca u toj poslovnici). Svaki djelatnik ima svoje jedinstveno korisničko ime (točno 8 znakova) i šifru (točno 8 znakova). Djelatnika u toj tvrtci je najviše 100. Poslovnica ima točno 10, a svaka je jedinstveno određena svojim nazivom (od najviše 100 znakova). Svaki artikl ima svoj naziv (najviše 50 znakova) i jedinstveni 10-znamenkasti kod. Za svaku poslovnicu i artikl pamtimo koliko tog artikla ima u toj poslovnici i to: koliko je već rezervirano i koliko je tog artikla još raspoloživo za rezervaciju. Pretpostavite da niti jedan od naziva u ovome zadatku ne sadrži znakove `','`, `'-'` i `'.'`.

Protokol za komunikaciju serverskog i klijentskog programa definiran je na sljedeći način:

- Zaglavlje poruke sastoji se od koda (koji određuje vrstu poruke), korisničkog imena i lozinke.
- Vrste poruka uključuju sljedeće (u drugom stupcu je naveden kod):

| | | |
|---|---|---|
| *UPIT naziv* | 1 | djelatnik želi sve kodove artikala s nazivom `naziv` zajedno s popisom poslovnica i količinom tog artikla u tim poslovnicama koji su raspoloživi za rezervaciju |
| *UPIT_R informacije* | 2 | server šalje odgovor s informacijama za gornji upit |
| *REZERVIRAJ br kod poslovnica* | 3 | djelatnik želi rezervirati `br` komada artikla s kodom `kod` u poslovnici `poslovnica` |
| *ODGOVOR poruka* | 4 | server šalje string koji je jednak "OK" za uspješno obrađen zahtjev ili opis greške |

- Pretpostavimo da su implementirane funkcije za slanje i primanje poruke (s utičnice `sock`):
  - `int posalji(int sock, const char *poruka)`
  - `int primi(int sock, char **poruka)`

  pri čemu u tim funkcijama `poruka` predstavlja cijelu poruku (zaglavlje + tijelo) gdje su zaglavlje i tijelo, kao i komponente zaglavlja, odijeljeni znakom zareza (','). Sami odredite kako su odijeljene komponente tijela poruke. Funkcije vraćaju 1 ako je došlo do greške (inače 0).

(1.) Implementirajte **sve** strukture/polja/varijable koje trebaju serveru za pamćenje svih navedenih podataka o djelatnicima, poslovnicama, artiklima kao i njihovoj dostupnosti. 4 boda

(2.) Implementirajte serversku funkciju `void rezerviraj(int sock, char *poruka)` (oprez: `poruka` kao i gore ima zaglavlje + tijelo) koja ažurira odgovarajuća polja i strukture na serveru te vraća poruku klijentu o uspješnoj rezervaciji ili neuspješnoj rezervaciji (u kom slučaju poruka sadrži opis greške). Rezervacija nije uspjela ako: ne postoji korisničko ime iz zaglavlja ili lozinka ne odgovara tom korisničkom imenu, ne postoji taj artikl ili ga nema u toj poslovnici u dovoljnoj količini. 7 bodova

(3.) Implementirajte klijentsku funkciju `void upit(int sock)` koja od djelatnika učita njegovo korisničko ime, lozinku i naziv nekog artikla. Zatim funkcija pita server za sve kodove tog artikla zajedno s popisom poslovnica i količinom tog artikla u tim poslovnicama koji su raspoloživi za rezervaciju. Iz pristigle poruke treba izvaditi sve dobivene informacije o tom artiklu i ispisati ih klijentu (ili u slučaju greške ispisati poruku o grešci). 6 bodova

JMBAG IME & PREZIME

## ČETVRTI ZADATAK

OSTVARENI BODOVI 13

Zadana je sljedeća struktura podataka koja reprezentira konačan skup cijelih brojeva. Broj `n` označava broj elemenata skupa, a jednodimenzionalno polje `elements` sadrži elemente skupa.

```
struct Set {
  int n;
  int* elements;
};
```

Koristeći pretpostavku da se jedan `int` može poslati jednom naredbom `send` i primiti jednom naredbom `recv`, implementirajte funkcije:

- `int sendUnion(int sock, struct Set A, struct Set B)`
- `int receiveUnion(int sock, struct Set* C)`

Funkcija `sendUnion` uzima dva elementa `A` i `B` tipa `Set` i mora poslati njihovu uniju $A \cup B$.

Funkcija `receiveUnion` mora tu uniju primiti i pobrinuti se da ona završi na adresi `C`.

Pazite na greške koje se mogu dogoditi. Svaka funkcija mora vratiti 0 ako je sve dobro prošlo, inače 1.

**Upute**: pobrinite se da polja strukture `Set` koje koristite budu adekvatne veličine (alocirajte potrebne podatke). Strukture možete slati element po element, ali ne cijele odjednom.