JMBAG

IME & PREZIME

Mreže računala

Prvi kolokvij, 29. studenoga 2024. 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 prirodan broj $n > 0$, string s_1 duljine n te string s_2 koji je IP adresa nekog računala u dekadskom obliku,
- ako je s_1 *hostname* računala čija IP adresa u dekadskom obliku je jednaka s_2, klijentu šalje sve alternativne *hostnameove* tog računala, a u suprotnom odgovarajuću obavijest.

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.

Arheološki muzej organizira obilaske pojedinih dijelova svog stalnog postava. Obilasci počinju svaki puni sat u radno vrijeme muzeja (preciznije, od utorka do subote, s prvim obilaskom u 08:00h, a posljednjim u 17:00h). Svaki obilazak pokriva točno jedan dio stalnog postava: Pretpovijesna zbirka, Antička zbirka, Srednjovjekovna zbirka, Egipatska zbirka i Numizmatička zbirka. Svaki obilazak određen je datumom, vremenom početka i zbirkom koja se obilazi (primjerice, na dan 29.11.2024 u 14:00h postoji najviše jedan obilazak Egipatske zbirke, najviše jedan obilazak Numizmatičke zbirke, itd.). Za svaki obilazak propisan je najveći broj sudionika (primjerice, 20 sudionika).

Potrebno je dizajnirati protokol za mrežnu aplikaciju koja služi za prijavu jedne ili više osoba za određeni obilazak. Klijent mora moći:

1. Za dani datum, dobiti popis svih obilazaka na taj datum zajedno s brojem slobodnih mjesta (ovo uključuje i obilaske s nula slobodnih mjesta).
2. Za danu zbirku, dobiti popis svih datuma i vremena početaka na te datume kada se održavaju obilasci te zbirke.
3. Za dani obilazak, rezervirati određen broj slobodnih mjesta. Pritom ako korisnik rezervira n mjesta, treba serveru poslati i imena te prezimena za svaku od n osoba za koju rezervira ta mjesta (ime i prezime svake osobe može se naravno sastojati od više riječi).
4. Svaka uspješno napravljena rezervacija određena je jedinstvenim kodom (koji se sastoji od 20 znakova). Klijent mora moći dobiti informacije o toj rezervaciji: datum i vrijeme početka obilaska, naziv zbirke koja se obilazi i imena te prezimena svih osoba za koje je rezervirano mjesto u tom obilasku pomoću te rezervacije.
5. Otkazati postojeću rezervaciju.
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 (primjerice, klijent želi rezervirati mjesta na nepostojećem obilasku). 3 boda

Odaberite primjer po jedne klijentske poruke za svaku od prvih pet 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 *Picassomon* posjeduje mrežnu aplikaciju za prijave na radionice slikanja Pokémona. Za svaki datum imaju organiziranu najviše jednu radionicu slikanja o kojoj se pamti: sat početka (tipa `int`), tema (kojeg Pokémona će polaznici slikati), cijena u eurima (tipa `float`) te maksimalan broj sudionika. Primjerice, za `29.11.2024` u `17` sati organizirana je radionica slikanja Pókemona `Mr. Mime` s cijenom sudjelovanja `25.99` eura i maksimalnim brojem sudionika `24`. Naravno, potrebno je za svaku radionicu pamtiti i sve prijavljene sudionike. Pritom se za svakog sudionika pamti samo njegov OIB (podsjetnik: OIB se sastoji od točno 11 znamenaka).

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

- Zaglavlje poruke sastoji se od koda (koji određuje vrstu poruke) te OIB-a korisnika.
- Vrste poruka uključuju sljedeće (u drugom stupcu je naveden kod):

INFO datum	1	Korisnik želi dobiti informacije o radionici na dani datum. Informacije o radionici uključuju sat početka, temu, cijenu i broj slobodnih mjesta.
INFO_R informacije	2	Server šalje odgovor s informacijama za gornji upit.
PRIJAVI datum	3	Korisnik se želi prijaviti na radionicu koja se održava na dani datum.
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 0 ako je došlo do greške (inače 1).

1. Implementirajte **sve** strukture/polja/varijable koje trebaju serveru za pamćenje svih navedenih podataka o radionicama. 4 boda

2. Implementirajte serversku funkciju `void prijava(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 ili neuspješnoj (u kom slučaju poruka sadrži opis greške) prijavu na radionicu. Prijava nije uspjela ako: OIB iz zaglavlja nije ispravan (točno 11 znamenaka!), ne postoji radionica na datum iz tijela poruke ili nema slobodnih mjesta na toj radionici. 7 bodova

3. Implementirajte klijentsku funkciju `void upit(int sock)` koja od korisnika učita njegov OIB te datum (možete koristiti tri podatka tipa `int` za dan, mjesec i godinu). Zatim funkcija pita server za informacije o radionici koja se održava na taj datum. Iz pristigle poruke treba izvaditi sve dobivene informacije o toj radionici i ispisati ih klijentu (ili u slučaju greške ispisati poruku o grešci). Ispis mora biti u sljedećem obliku (na primjeru koji je prethodno naveden u tekstu zadatka):

```
Mr. Mime, 29.11.2024, 17h, 14.5eur, 5 mjesta slobodno
```

6 bodova

JMBAG IME & PREZIME

ČETVRTI ZADATAK

OSTVARENI BODOVI 13

Zadana je sljedeća struktura podataka koja reprezentira binarnu relaciju $R \subseteq \mathbb{Z} \times \mathbb{Z}$. Broj `n` označava broj parova u relaciji, a dvodimenzionalno polje `pairs` sadrži parove relacije. Polje `pairs` je dimenzija $\texttt{n} \times 2$.

```
struct Relation {
  int n;
  int** pairs;
};
```

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

- `int sendInverse(int sock, struct Relation R)`
- `int receiveInverse(int sock, struct Relation* S)`

Funkcija `sendInverse` uzima element `R` tipa `Relation` i mora poslati inverz relacije R.

Funkcija `receiveInverse` mora taj inverz primiti i pobrinuti se da on završi na adresi `S`.

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 `Relation` koje koristite budu adekvatne veličine (alocirajte potrebne podatke). Strukture možete slati element po element, ali ne cijele odjednom.