Mreže računala

Rješenja zadataka s prvog kolokvija, 29. studenoga 2024. godine

RJEŠENJE PRVOG ZADATKA

UKUPNO BODOVA: 8

```
while(1){
    struct sockaddr_in klijentAdresa;
    int lenAddr = sizeof( klijentAdresa );
    int commSocket = accept(listenerSocket,
                    (struct sockaddr *)&klijentAdresa, &lenAddr);

    if( commSocket == -1 ){
        perror( "accept" );
        continue;
    }

    int n_n;
    if(recv(commSocket, &n_n, sizeof(n_n), 0) != sizeof(n_n)){
        perror("receive"); close(commSocket);
        continue;
    }
    n = ntohl(n_n);
    char *s1 = (char*)malloc(n*sizeof(char));
    if(recv(commSocket, s1, sizeof(s1), 0) != sizeof(s1)){
        perror("receive"); close(commSocket);
        continue;
    }

    char s2[16];
    if(recv(commSocket, s2, sizeof(s2), 0) != sizeof(s2)){
        perror("receive"); close(commSocket);
        continue;
    }

    struct ind_addr binarniIP;
    if(!inet_aton(s2, &binarniIP)){
        herror("error"); close(commSocket);
        continue;
    }
    struct hostent *info = gethostbyaddr((const char*)&binarniIP,
                sizeof(binarniIP), AF_INET);

    if(!strcmp(s1, info->h_name)){
```

```
        for(int i = 0; info->h_aliases[i] != NULL; ++i){
            if(send(commSocket, h_aliases[i],
                sizeof(h_aliases[i]),0) != strlen(info->h_aliases[i])){
                    herror("error"); close(commSocket);
                    continue;
                }
        }
    }

    else{
        char odg[] = "Poslani hostname i IP adresa se ne poklapaju";
        send(commSocket, odg, sizeof(odg), 0);
    }

    close(commSocket);
}
```

RJEŠENJE DRUGOG ZADATKA

UKUPNO BODOVA: 12

Format poruka 1 bod

- Svaka poruka sastoji se od dva dijela: zaglavlja i tijela poruke. Zaglavlje se sastoji od dva integera - prvi predstavlja duljinu poruke (ne uključujući zaglavlje), a drugi predstavlja kod kojim je jednoznačno određena jedna od dolje navedenih vrsta poruke.

Jedan mogući dizajn sustava komunikacije 5 bodova

1. *INFODATUM datum* - klijent želi dobiti popis svih obilazaka na dani datum zajedno s brojem slobodnih mjesta

2. *INFODATUM_R popis* - server vraća niz uređenih trojki oblika *(vrijeme_početka,naziv_zbirke, broj_slobodnih_mjesta)* odvojenih zarezom

3. *INFOZBIRKA naziv_zbirke* - dobiti popis svih datuma i vremena početaka na te datume kada se održavaju obilasci zbirke danog naziva

   Napomena 1: U tekstu zadatka svi nazivi zbiraka koje su dostupne za obilazak sastoje se od točno dvije riječi pa ovdje i u nastavku rješenja ne predstavlja problem što pišemo *naziv_zbirke*, a ne *"naziv_zbirke"*, *(naziv_zbirke)* ili sl. (tj. ne treba odvajati *naziv_zbirke* od ostalih dijelova tijela poruke jer znamo da se on sastoji od točno dvije riječi).

4. *INFOZBIRKA_R popis* - server vraća niz uređenih parova oblika *(datum,vrijeme_početka)* odvojenih zarezom

5. *REZERVIRAJ datum vrijeme_početka naziv_zbirke br_mjesta popis* - klijent želi rezervirati određen broj slobodnih mjesta te za svako slobodno mjesto navodi ime(na) i prezime(na) osobe za koju rezervira to mjesto. Pritom je *popis* niz od točno *n* uređenih parova (međusobno odvojenih zarezima) *(imena,prezimena)* pri čemu svaki par odgovara jednoj osobi za koju je napravljena rezervacija.

   Napomena 2: Prema zadatku, svaki obilazak određen je datumom, vremenom početka i zbirkom koja se obilazi pa klijent za odabir obilaska treba navesti sva ta tri podatka kako bi odredio obilazak za koji želi napraviti rezervaciju.

   Napomena 3: Alternativno, može se izostaviti *br_mjesta* pa bi server sam trebao prebrojati za koliko osoba je dobio podatke. Također, može se ostaviti *br_mjesta*, a izbaciti dio *popis*, u kom slučaju treba uvesti još jedna vrsta poruke, primjerice, *PODACIOSOBE imena;prezimena* te se od klijenta očekuje da nakon slanja zahtjeva za rezervaciju *n* slobodnih mjesta pošalje *n* takvih poruka.

6. *REZERVIRAJ_R kod* - ako na traženom obilasku ima dovoljno slobodnih mjesta (te je klijent poslao podatke za sve osobe za koje traži rezervaciju), server klijentu vraća jedinstveni kod koji određuje tu rezervaciju. Inače, server šalje poruku tipa *ODGOVOR* da rezervacija nije moguća (te razlog zašto je tome tako).

7. *INFO kod* - klijent želi dobiti informacije o rezervaciji koja je određena danim kodom

8. *INFO_R datum vrijeme naziv_zbirke popis* - server šalje klijentu datum i vrijeme početka obilaska, naziv zbirke koja se obilazi i popis uređenih parova (međusobno odvojenih zarezima) *(imena,prezimena)* pri čemu svaki par odgovara jednoj osobi za koju je napravljena rezervacija (određena kodom koji je poslao klijent)

9. *OTKAŽI kod* - klijent želi otkazati rezervaciju određenu danim kodom

10. *BOK* - klijent šalje serveru znak za prekid komunikacije

11. *ODGOVOR poruka* - poruka je string koji je jednak *OK* ako je zahtjev uspješno obrađen, inače sadrži opis greške (greške su opisane u nastavku ovog rješenja!)

Napomena 4: Nije bilo potrebno pisati gornja objašnjenja za svaku vrstu poruke ukoliko je iz naziva dijelova tijela poruke bilo jasno što oni predstavljaju. Primjer: *datum*, *vrijeme_početka*, *kod* i slično - ti nazivi dijelova tijela poruke ne zahtijeva dodatno objašnjenje, no nazivi dijelova tijela poruke poput *popis*, *poruka* i slično zahtijevaju detaljnije objašnjenje. Također, nije bilo potrebno uz svaku vrstu kao gore pisati neki broj koji ju predstavlja (npr. uz *ODGOVOR* imamo broj 11), no u tom slučaju je u donjim primjerima poruka trebalo negdje napomenuti koji broj korišten u tim primjerima predstavlja koju vrstu poruke (osim ako je umjesto 11 navedno *ODGOVOR* i sl.).

Moguće greške 3 boda

- neispravan format vremena, datuma, broja mjesta koja se rezerviraju, koda (kod se mora sastojati od 20 znakova)
- navedeno vrijeme i datum u prošlosti ili ne postoji obilazak na navedeni datum i/ili vrijeme i/ili za navedenu zbirku (s time da navedeno ime zbirke mora biti jedno od onih koja su navedena u zadatku)
- obilazak na kojem se želi rezervirati *n* slobodnih mjesta nema toliko slobodnih mjesta
- ne postoji rezervacija za navedeni kod
- popisi nisu u ispravnom formatu (ispravan format naveden je ranije uz svaku vrstu poruke koja koristi neki popis)
- pri rezervaciji *n* slobodnih mjesta, nisu navedeni podaci za točno *n* osoba (ili su za neku osobu navedeni nepotpuni podaci, primjerice, samo ime bez prezimena)

Primjeri klijentskih poruka i pripadni odgovori servera 3 boda

Napomena 5: Ispod je prikazana po jedna klijentska poruka i pripadni odgovor servera za svaku od prvih 5 funkcionalnosti iz teksta zadatka. Izostavljene su poruke servera *[2,ODGOVOR] OK* ako je zahtjev klijenta uspješno obrađen. Naravno, u slučaju greške potrebno je prikazati odgovarajući odgovor servera (ovdje su odabrani primjeri koji ilustriraju uspješno obrađene zahtjeve klijenta).

klijent:	[10,INFODATUM] 28.12.2024
server:	[53,INFODATUM_R] (09:00h,Antička zbirka,4),(14:00h,Egipatska zbirka,0)
klijent:	[19,INFOZBIRKA] Numizmatička zbirka
server:	[59,INFOZBIRKA_R] (16.12.2024,08:00h),(17.12.2024,12:00h),(20.12.2024,17:00h)
klijent:	[70,REZERVIRAJ] 17.12.2024 12:00h Numizmatička zbirka 2 (Iva Marija,Ivić),(Maja,Majić)
server:	[20,REZERVIRAJ_R] Ani9-43dfrx0+2djfZnM
klijent:	[20,INFO] Ani9-43dfrx0+2djfZnM
server:	[68,INFO_R] 17.12.2024 12:00h Numizmatička zbirka (Iva Marija,Ivić),(Maja,Majić)
klijent:	[20,OTKAŽI] Ani9-43dfrx0+2djfZnM
server:	(ovdje može [2,ODGOVOR] OK)
klijent:	[0,BOK]
server:	(ovdje može [2,ODGOVOR] OK)

Napomena 6: U ovome zadatku se za vrijeme može umjesto, primjerice, 09:00h koristiti 9:00h ili 9:00 ili 9h ili samo 9 (jer obilasci počinju samo u puni sat).

RJEŠENJE TREĆEG ZADATKA

UKUPNO BODOVA: 17

1. 4 boda

```
typedef struct {
        char datum[11]; /* datumi su oblika dd.mm.gggg */
        int sat;
        char* tema;
        float cijena;
        int max_br;
        char** sudionici; /* za OIB-e sudionika */
        int br_sudionika; /* koliko imamo tih OIB-a */
} radionica;

radionica* radionice;

int br_radionica;
```

Napomene:

- U nastavku ćemo radi jednostavnosti koristiti da su OIB-i, naziv teme i datum radionice stringovi (primjerice, zato smo umjesto `char datum[10]` u gornjem napisali `char datum[11]` kako bi imali mjesta za taj dodatni znak `'\0'` koji označava kraj stringa).
- U prikazanom rješenju, ako je na neku radionicu `r` prijavljeno već 5 sudionika tada je `r.sudionici[0]` OIB prvog sudionika, `r.sudionici[1]` OIB drugog sudionika i tako dalje, a `r.br_sudionika` je jednak 5.
- Nije poznat maksimalan broj radionica, maksimalna duljina stringa koji predstavlja naziv teme te ograničenje na `max_br` sudionika za sve moguće radionice (svaka radionica ima propisan svoj, proizvoljno velik `max_br`, pa će `radionice`, te `tema` i `sudionici` za pojedinu radionicu biti dinamički alocirani na serveru.

2. 7 bodova

Prvo jedna pomoćna funkcija koju ćemo koristiti u nastavku (za jednostavnije slanje poruke):

```
void saljiPoruku(int sock, char* oib, char* poruka) {
        char odgovor[100]; /* 100 je dovoljno s obzirom
                              na poruke kakve saljemo */
        sprintf(odgovor,"4,%s,%s", oib, poruka);
        posalji(sock, odgovor);
}
```

Napomene:

- Prema zadatku možemo sami odrediti kako su odijeljene komponente tijela poruke - u ovome rješenju će između svakog dijela tijela poruke biti točno jedan znak zareza (`','`).
- U sljedećoj funkciji `prijava` bismo mogli imati razne provjere, no napravit ćemo samo one koje se traže u zadatku.

```
void prijava(int sock, char* poruka) {
    /* izvadimo dijelove iz poruke klijenta */
    int vrsta, i;
    char* oib; /* ne znamo ispravnost oib-a (11 znamenaka) */
    oib = malloc (strlen(poruka)); /* dovoljne velicine */
    char datum[11];
    sscanf(poruka, "%d,%[^,],%s", &vrsta, oib, datum);

    /* provjera ispravnosti OIB-a (tocno 11 znamenaka) */
    for(i = 0; i < strlen(oib); ++i)
        if(!isdigit(oib[i]))
                break;
    if(i != strlen(oib) || strlen(oib) != 11) {
        saljiPoruku(sock,oib,"Dobiveni OIB nije ispravan!");
        free(oib);
        return;
    }

    /* provjera postoji li radionica na dani datum */
    for(i = 0; i < br_radionica; ++i)
        if(strcmp(radionice[i].datum,datum) == 0)
            break;
    if(i == br_radionica) {
        saljiPoruku(sock,oib,"Nema radionice na taj datum!");
        free(oib);
        return;
    }

    /* provjera ima li mjesta na trazenoj (i-toj!) radionici */
    if(radionice[i].br_sudionika >= radionice[i].max_br) {
        saljiPoruku(sock,oib,"Nema mjesta na toj radionici!");
        free(oib);
        return;
    }

    /* prijavimo korisnika na (i-tu!) radionicu */
    radionice[i].br_sudionika++;
    radionice[i].sudionici = realloc(radionice[i].sudionici,
        radionice[i].br_sudionika * sizeof(char*));
    radionice[i].sudionici[br_sudionika-1]
                = malloc(12 * sizeof(char));
    strcpy(radionice[i].sudionici[br_sudionika-1], oib);

    /* javimo korisniku da je sve dobro proslo */
    saljiPoruku(sock, oib, "OK");
    free(oib);
}
```

3. 6 bodova

```
void upit(int sock) {
    /* ucitavanje podataka od korisnika */
    char oib[12];
    int dan, mjesec, godina;
    scanf(" %s %d %d %d", oib, dan, mjesec, godina);

    /* pitamo server za informacije o radionici */
    sprintf(poruka, "1,%s,%s%d.%s%d.%d", oib,
        (dan < 10) ? "0" : "", dan,
        (mjesec < 10) ? "0" : "", mjesec, godina);
    posalji(sock, poruka);

    /* primimo poruku od servera */
    char* odgovor;
    primi(sock, &odgovor); /*primi alocira dovoljno memorije*/

    /* provjerimo vrstu poruke (provjera radi li se o gresci) */
    int vrsta_poruke, i;
    sscanf(odgovor,"%d", &vrsta_poruke);

    /* u slucaju greske, izvadimo i ispisemo poruku o njoj */
    if(vrsta_poruke == 4) {
        /* poruka je dio odgovora nakon drugog zareza */
        int i, br_zareza = 0;
        for(i = 0; br_zareza < 2; ++i) {
            if(odgovor[i] == ',')
                br_zareza++;
        }
        printf("Greska: %s\n", &odgovor[i]);
        free(odgovor);
        return;
    }

    /* izvadimo i prikazemo dobivene informacije o radionici */
    char *tema = malloc(strlen(odgovor)); /* dovoljna velicina */
    int sat, br_slobodnih;
    float cijena;
    sscanf(odgovor,"%*d,%d,%[^,],%f,%d",
          &sat, tema, &cijena, &br_slobodnih);
    printf("%s, %s%d.%s%d.%d, ", tema,
        (dan < 10) ? "0" : "", dan,
        (mjesec < 10) ? "0" : "", mjesec, godina);
    printf("%dh, %feur, %d mjesta slobodno", sat,
        cijena, br_slobodnih);
    free(odgovor);
    free(tema);
}
```

RJEŠENJE ČETVRTOG ZADATKA

UKUPNO BODOVA: 13

```
int sendInverse(sock, struct Relation R){

    int n_n = htonl(R.n);
    if(send(sock, &n_n, sizeof(n_n), 0) != sizeof(n_n)) return 1;

    for(i = 0; i < R.n; ++i){
        n_n = htonl(R.pairs[i][1]);
        if(send(sock, &n_n, sizeof(n_n), 0) != sizeof(n_n)) return 1;

        n_n = htonl(R.pairs[i][0]);
        if(send(sock, &n_n, sizeof(n_n), 0) != sizeof(n_n)) return 1;
    }

    return 0;
}
```

Napomene:

- Niz `pairs` ima `n` redaka i 2 stupca. Inverz također ima `n` redaka i 2 stupca.
- Prvo je trebalo poslati broj elemenata u inverzu relacije.

```
int receiveInverse(sock, struct Relation *S){
    int n_n;
    if(recv(sock, &n_n, sizeof(n_n), 0) != sizeof(n_n)) return 1;
    S->n = ntohl(n_n);
    S->pairs = (int**)malloc(S->n*sizeof(int*));

    for(int i = 0; i < S->n; ++i){
        S->pairs[i] = (int*)malloc(2*sizeof(int));

        if(recv(sock, &n_n, sizeof(n_n), 0) != sizeof(n_n)) return 1;
        S->pairs[i][0] = ntohl(n_n);

        if(recv(sock, &n_n, sizeof(n_n), 0) != sizeof(n_n)) return 1;
        S->pairs[i][1] = ntohl(n_n);
    }

    return 0;
}
```

H. Marciuš & S. Horvat, 11. prosinca 2024.