JMBAG IME & PREZIME

# Mreže računala

Ispit, 15. veljače 2024. godine

Na ispitu je dozvoljeno korištenje samo pribora za pisanje i službenog šalabahtera. Predajete samo papire koje ste dobili. Ispit se sastoji od šest zadataka na kojima je moguće ostvariti najviše 80 bodova. Ispit se piše 2 sata.

## PRVI ZADATAK

OSTVARENI BODOVI | | 20 |

Zadana je sljedeća struktura podataka koja sadrži cijeli broj `n` te niz stringova `text`. Broj `n` je duljina niza `text`.

```
struct Strings{
  int n;
  char** text;
};
```

Koristeći pretpostavku da se svaki broj i svaki string može poslati jednom naredbom `send` i primiti jednom naredbom `recv`, implementirajte funkcije:

- `int sendStrings(int sock, struct Strings S)`
- `int receiveStrings(int sock, struct Strings* T)`

Funkcija `sendStrings` uzima jedan element `S` tipa `Strings`. Ako se u nizu `text` nalazi string koji je validan *hostname* nekog računala, funkcija šalje broj IP adresa tog računala te sve IP adrese tog računala u dekadskom obliku (kao string). Ako se u nizu ne nalazi validan *hostname* nekog računala, funkcija šalje 0. Ako se u nizu nalazi više validnih *hostnameova*, gleda se samo prvi u nizu.

Funkcija `receiveStrings` mora te podatke primiti i pobrinuti se da oni završe na adresi `T`.

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

JMBAG IME & PREZIME

## DRUGI ZADATAK

OSTVARENI BODOVI 20

Restoran *Gospodar Pereca* ima na raspolaganju ukupno 50 stolova. Od tih 50 stolova (koji su identificirani prirodnim brojevima od 1 do 50), za 30 stolova (s brojevima od 1 do 30) mogu sjesti po 4 osobe, dok za preostalih 20 stolova može sjesti po 6 osoba. Svaki stol može se rezervirati samo na 1 sat i to samo unutar trenutnog tjedna (primjerice, stol 20 za petak u 19 sati - istaknimo da se rezervira od punog sata, tj. od, primjerice, 19 sati, a ne od, primjerice, 19:15h, 19:24h i sl.). Restoran radi svakodnevno od 14:00h do 23:00h.

(a) *(8 bodova)* Potrebno je dizajnirati protokol za mrežnu aplikaciju koja služi za rezervaciju stolova u tom restoranu. Istaknimo da se u ovome podzadatku **ne očekuje** pisanje programskog koda! Klijent mora moći:

(i) Za dan i vrijeme (primjerice, za petak u 19 sati) dobiti informaciju koliko je kojih tipova stolova dostupno (tj. koliko je stolova za 4 osobe dostupno i koliko je stolova za 6 osoba tada dostupno). Ako korisnik navede samo dan (primjerice, samo navede petak), potrebno mu je poslati informacije koliko je kojih tipova stolova dostupno za svaki sat u kojem restoran taj dan radi.

(ii) Rezervirati jedan stol za dani dan i sat. Prilikom rezervacije obavezno treba navesti OIB osobe koja vrši rezervaciju te tip stola koji rezervira (za 4 ili 6 osoba). U slučaju uspješne rezervacije, potrebno je javiti broj koji identificira rezervirani stol.

Osmislite vrste poruka i njihov format koje razmjenjuju klijent i server. Navedite kakvi se sve tipovi grešaka mogu dogoditi u komunikaciji. Odaberite primjer po jedne klijentske poruke za svaku od točaka (i) i (ii) (u skladu s vašim definicijama mogućih vrsta i formata poruka) te pripadne odgovore servera.

(b) *(5 + 7 = 12 bodova)* 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). Sami odredite kako su odijeljeni zaglavlje i tijelo, kao i njihove komponente. Funkcije vraćaju 1 ako je došlo do greške (inače 0).

(i) Implementirajte **sve** strukture/polja/varijable koje trebaju serveru za pamćenje svih navedenih podataka o stolovima i njihovim rezervacijama u trenutnom tjednu.

(ii) Implementirajte serversku funkciju `void rezerviraj(int sock, char *poruka)` (pri čemu `poruka` kao i gore ima zaglavlje + tijelo) koja ažurira odgovarajuće strukture, polja i varijable na serveru te vraća poruku klijentu o uspješnoj rezervaciji ili neuspješnoj rezervaciji (u kom slučaju poruka sadrži opis greške).

JMBAG IME & PREZIME

## TREĆI ZADATAK

OSTVARENI BODOVI 8

Pretpostavite da je zadan sljedeći HTML kod i CSS pravila. Na donjoj slici skicirajte izgled odgovarajuće stranice. Nekako, primjerice križanjem, naznačite obojana područja i napišite o kojoj je boji riječ.

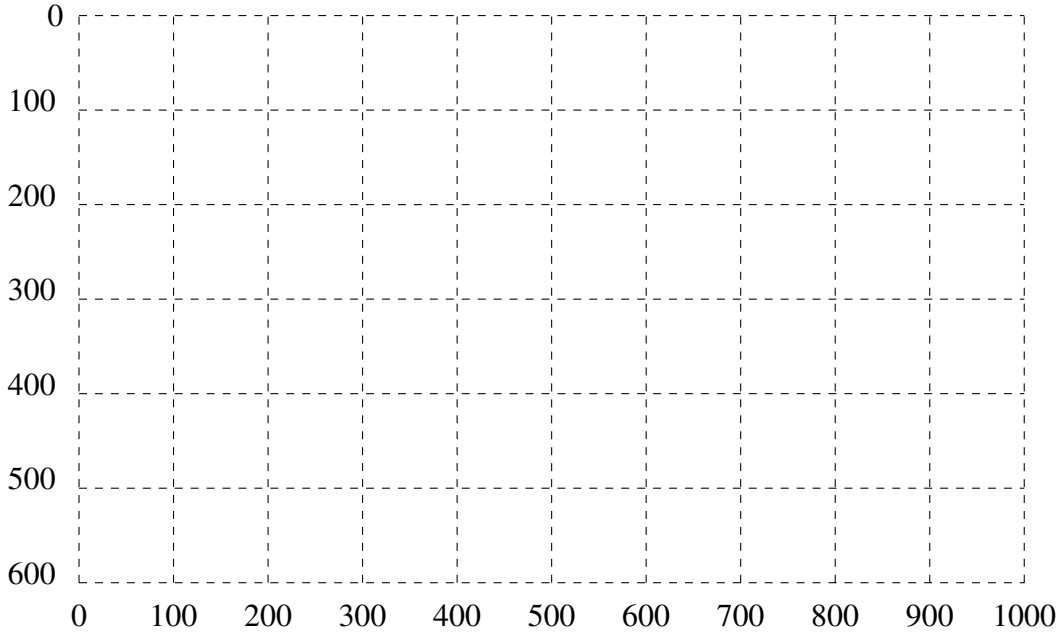
**CSS**

```
* {
    background: yellow;
    float: right;
    margin: 0;
}
#A {
    float: right;
    height: 50%;
}
.A p {
    float: left;
    width: 20%;
    height: 20%;
    background: red;
    margin: 0 50px;
    font-size: 50px;
    border: 1px solid black;
}
div.A {
    width: 1000px;
    height: 500px;
    background-color: blue;
}
div  .B {
    clear: right;
}
```

**HTML**

```
...
<div class="A">
    <p class="A">
        A
    </p>
    <p class="B">
        B
    </p>
    <p class="C" id="A">
        C
    </p>
    <p class="D" id="B">
        D
    </p>
</div>
```

Stranica dimenzija $1000 \times 600$ px:



JMBAG

IME & PREZIME

## ČETVRTI ZADATAK

OSTVARENI BODOVI | | 10 |

Što je Internet Control Message Protocol? Opišite značenje ICMP poruka *source quench*, *time exceeded* i *redirect*.

Objasnite kako Microsoft `tracert` koristi ICMP za dobivanje puta između dva čvora na Internetu.

JMBAG

IME & PREZIME

## PETI ZADATAK

OSTVARENI BODOVI 10

Objasnite razliku između paradigmi *Remote procedure Call* - RPC i *Remote method invocation* - RMI.

Što je komunikacijski umetak?

Navedite dva primjera RPC i dva primjera RMI alata.

JMBAG

IME & PREZIME

## ŠESTI ZADATAK

OSTVARENI BODOVI | 12

Navedite i opišite sva značajna softverska rješenja (od hardverskog do aplikacijskog sloja) potrebna za sigurno slanje podataka sa svog računala na udaljeni server korištenjem aplikacije tipa *FileZilla*, *WinSCP* itd. Pretpostavka je da sjedite za računalom spojenim Ethernet kabelom na usmjernik.