

PROGRAMIRANJE II (21.07.2020.)

1 (15 bodova)

Napisati funkciju čiji je prototip:

```
int char_count(const char *str1,  
               const char *str2);
```

Funkcija u datom stringu *str1* pronalazi broj znakova koji se nalaze u stringu *str2* te vraća ukupan broj detektovanih znakova kao rezultat. Svaki jedinstveni znak broji se samo jednom tj. duplikati se ne broje. Dozvoljeno je korišćenje funkcija iz *<string.h>*.

Napisati funkciju čiji je prototip:

```
char* max_count(const char *str1,  
                int (*t)(const char *, const char *), int n, ...);
```

Navedena funkcija nad svakim stringom koji je proslijeđen kao neobavezan argument poziva funkciju koja je proslijeđena kao argument, pri čemu je neobavezni argument prvi argument, a argument *str1* drugi argument funkcije. Funkcija kao rezultat vraća string za koji je vraćen najveći broj nakon poziva funkcije koja je argument (posljednji pronađen ako ih je više sa istim brojem).

U glavnom programu ilustrovati upotrebu funkcije sa neobavezanim argumentima. Funkcija koja je proslijeđena kao argument je *char_count*. String po kojem se traže znakovi je proslijeđen putem argumenta komandne linije.

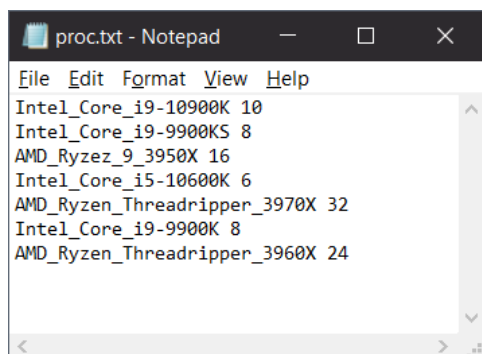
2 (15 bodova)

U tekstualnoj datoteci, čiji se naziv proslijeđuje kao prvi argument komandne linije, smješteni su podaci o nepoznatom broju modela procesora. Za svaki model procesora čuvaju se naziv i broj jezgara procesora.

Potrebno je:

- učitati podatke iz tekstualne datoteke i kreirati dinamički niz,
- primjenom *selection-sort* algoritma sortirati modele procesora po broju jezgara u opadajućem redoslijedu,
- napisati rekurzivnu funkciju koja računa prosječan broj jezgara procesora (u svakoj iteraciji jedan korak računanja),
- na standardni izlaz ispisati sve modele procesora koji imaju više jezgara od prosječnog broja, koji se dobije pozivom funkcije iz prethodne tačke.

Primjer tekstualne datoteke:



3 (15 bodova)

Neka je dat tip:

```
typedef struct node_t {  
    int info; // informacioni sadrzaj  
    struct node_t *left, *right;  
} NODE_T;
```

kojim se reprezentuje čvor stabla binarne pretrage, pri čemu je informacioni sadržaj čvora cjelobrojni podatak (*info*). Neka je dat tip:

```
typedef struct node {  
    NODE_T *root; // informacioni sadrzaj  
    struct node *next;  
} NODE;
```

kojim se reprezentuje čvor jednostruko ulančane liste binarnih stabala pretrage.

Napisati funkciju koja dodaje čvor na kraj liste, pri čemu korisnik u sklopu funkcije treba da unese informacioni sadržaj svakog čvora binarnog stabla pretrage dok god ne prekine unos. Prototip funkcije je:

```
NODE* add(NODE **head);
```

Napisati funkciju koja računa i kao rezultat vraća aritmetičku sredinu za čvor jednostruko povezane liste. Aritmetička sredina predstavlja sumu informacionih sadržaja podijeljenu sa brojem čvorova binarnog stabla. Prototip funkcije je:

```
int arithmetic(NODE *node);
```

Napisati funkciju koja kreira novu listu, a čiji su članovi oni čvorovi polazne liste čija je aritmetička sredina veća od *n* (parametar funkcije). Prototip funkcije je:

```
NODE* new_list(NODE *head, int n);
```

4 (15 bodova)

Korištenjem netežinskog, neusmjerenog grafa vizuelno se prikazuje međusobna komunikacija servera u jednoj server sali. Čvor grafa predstavlja server, pri čemu je informacioni sadržaj čvora oznaka servera (cijeli broj). Grana koja spaja dva čvora/servera označava da postoji komunikacija/veza između njih.

Potrebno je:

- koristiti matricnu reprezentaciju grafa,
- iz ulazne tekstualne datoteke, čiji se naziv proslijeđuje putem standardnog ulaza, učitati dimenzije matrice susjednosti, a potom i matricu susjednosti grafa,
- napisati funkciju za obilazak grafa po dubini (DFS algoritam),
- napisati funkciju koja računa broj server klastera (povezanih komponenata grafa) korištenjem DFS algoritma i
- ispisati broj server klastera na standardni izlaz u glavnoj (main) funkciji.

Primjer tekstualne datoteke:

