ELEKTROTEHNIČKI FAKULTET
BANJA LUKA

PROGRAMIRANJE II (25.06.2019.)

❶ (**10 bodova**) Neka je dat prototip funkcije:

```
void procesiraj(FILE *f1, FILE *f2, int n, …);
```

Funkcija iz tekstualne datoteke *f1* (datoteka je otvorena i ne treba je zatvarati) učitava stringove i kreira niz stringova. U datoteci su svi karakteri u jednom redu. Prilikom svakog učitavanja stringa iz datoteke *f1* učitava se tačno onoliko karaktera kolika je naredna vrijednost neobaveznog cjelobrojnog argumenta (*n* je broj neobaveznih argumenata). Nakon toga sortirati niz stringova *shell-sort* algoritmom i upisati niz stringova u tekstualnu datoteku *f2* na način da je svaki string upisan u novi red (datoteka je takođe otvorena i ne treba je zatvarati). Ukoliko u datoteci ima manje karaktera od onog što je navedeno narednim neobaveznim argumentom, potrebno je ispisati na standardni izlaz poruku da u tekstualnoj datoteci nema dovoljno karaktera, učitati preostale karaktere u niz stringova kao poseban string, koji je posljednji član niza stringova, prekinuti dalje učitavanje stringova iz datoteke preko neobaveznih argumenata te nastaviti sa sortiranjem niza i upisom u tekstualnu datoteku *f2*. Ukoliko nakon preuzimanja svih neobaveznih argumenata postoji još karaktera u datoteci, preostale karaktere smjestiti u jedan string, koji je posljednji član niza stringova, te nastaviti sa sortiranjem niza i upisom u tekstualnu datoteku *f2*.

❷ (**10 bodova**) U ulaznoj binarnoj datoteci upisani su podaci o nenultim elementima niza četvorobajtnih označenih cijelih brojeva. U prva dva bajta datoteke upisan je ukupan broj elemenata datog niza (cijeli neoznačeni broj), a u naredna dva bajta upisan je ukupan broj nenultih elemenata datog niza (cijeli neoznačeni broj). Nakon toga upisani su podaci o nenultim elementima niza, pri čemu je za svaki nenulti element niza upisana pozicija (cijeli neoznačeni broj veličine dva bajta) i vrijednost elementa na toj poziciji (cijeli označeni broj veličine četiri bajta). Prilikom upisa podataka korištena je LE konvencija.

Napisati funkciju u kojoj treba iz ulazne binarne datoteke čiji je naziv prvi parametar funkcije, pročitati podatke o nenultnim elementima niza te rekonstruisati i vratiti originalni (dinamički) niz. Ukupan broj elemenata niza vraća se preko parametra *br*. Prototip funkcije je:

```
int* rekonstruisi(const char *naziv,
                  unsigned short *br);
```

Napisati funkciju koja u nizu cjelobrojnih podataka pronalazi i vraća poziciju (prvo pojavljivanje) zadatog elementa. Koristiti metodu sekvencijalnog pretraživanja sa stražom (pretpostaviti da je niz prethodno pripremljen za pretraživanje sa stražom). Prototip funkcije je:

```
int pretrazi(int *niz, int kljuc);
```

Na primjer, neka je definisan niz na sljedeći način:

```
int niz[] = { 10, 0, 0, 11, 5 };
```

Sadržaj ulazne binarne datoteke za prethodni niz je:

```
05 00 03 00 00 00 0A 00 00 00 03 00 0B 00 00 00 04
00 05 00 00 00
```

❸ (**10 bodova**) Neka je dat sljedeći tip podataka:

```
typedef struct cvor {
  char str[100]; // inf. sadrzaj
  struct cvor *sljedeci;
} CVOR;
```

kojim se reprezentuje čvor jednostruko povezane liste, pri čemu je informacioni sadržaj čvora string *str*.

Napisati funkciju koja dodaje novi podatak na početak liste. Prototip funkcije je:

```
void dodaj_pocetak(CVOR **pglava,
                   const char *podatak);
```

Napisati funkciju koja formira stek (ulančana reprezentacija) tako što sve elemente jednostruko ulančane liste (parametar *glava* pokazuje na prvi čvor liste) koji zadovoljavaju neki uslov, dodaje na stek (koristeći funkciju *dodaj_pocetak*). Funkcija treba da vrati pokazivač na prvi čvor steka ili NULL ako je stek prazan. Parametar *provjeri* predstavlja pokazivač na funkciju koja za zadati string provjerava da li je ispunjen neki uslov. Funkcija za provjeru uslova vraća vrijednost 0 ako uslov nije ispunjen, a vrijednost 1 ako je uslov ispunjen. Prototip funkcije je:

```
CVOR* formiraj(CVOR *glava,
               int (*provjeri)(const char *));
```

❹ (**10 bodova**) Dat je sljedeći tip podataka kojim se reprezentuje čvor stabla binarnog pretraživanja:

```
typedef struct cvor {
    MOB_TEL mobilni_tel;
    struct cvor *lijevi, *desni;
} CVOR;
```

Informacioni sadržaj čvora je podatak tipa MOB_TEL:

```
typedef struct {
    char naziv[21];
    int kolicina;
    double cijena;
} MOB_TEL;
```

Pretpostaviti da su implementirane sljedeće funkcije:

```
// ucitavanje podataka putem standardnog ulaza
void ucitaj_podatke(MOB_TEL *mobilni_tel);

// formiranje novog cvora – pomocna funkcija
CVOR* novi_cvor(MOB_TEL *mobilni_tel);

// brisanje stabla – oslobadjanje zauzete memorije
void brisi_stablo(CVOR *korijen);
```

Potrebno je:

- napisati funkciju koja dodaje podatke o novom mobilnom telefonu u stablo, tako da je kriterijum pri dodavanju novog čvora u stablo naziv telefona:

  ```
  CVOR* dodaj_cvor(CVOR *korijen, MOB_TEL *m_tel);
  ```

  Ukoliko u stablu već postoji mobilni telefon sa datim nazivom potrebno je ažurirati njegove podatke.
- napisati funkciju koja pretražuje stablo po nazivu telefona (rekurzivno):

  ```
  CVOR* trazi(CVOR *korijen, char *naziv);
  ```

  U slučaju neuspješne pretrage funkcija vraća vrijednost 0.
- napisati glavnu funkciju (**main**) u kojoj je potrebno:
  - unijeti podatke za *n* telefona i formirati odgovarajuće stablo,
  - pronaći telefon po nazivu (unosi se putem standardnog ulaza) i ispisati njegove podatke na standardni izlaz u formatu:

    ```
    naziv | kolicina | cijena
    ```