ELEKTROTEHNIČKI FAKULTET
BANJA LUKA

PROGRAMIRANJE II
(24.01.2023)

❶ **(20 bodova**) U ulaznoj binarnoj datoteci, u prva četiri bajta upisana je informacija (podatak tipa int) o broju podataka koji su upisani u datu datoteku, dok je u sljedećem bajtu upisana informacija (podatak tipa unsigned char) o veličini jednog upisanog podatka. Nakon toga su upisani podaci.

Napisati funkciju koja podatke upisane u datoteku učitava u memoriju (heap) i vraća adresu početka podataka u memoriji, dok informacije o broju podataka i veličini jednog podatka vraća preko parametara *b* (broj podataka) i *v* (veličina jednog podatka). Prototip funkcije je:

void* ucitaj(FILE *f, unsigned *b, unsigned char *v);

U glavnom programu (korištenjem funkcije *ucitaj*) pročitati podatke iz datoteke čiji je naziv prvi argument komandne linije, te ispisati broj učitanih podataka.

❷ **(20 bodova**) Neka je definisan tip:

```
typedef struct oglas {
    int d, m, g; // datum objavljivanja
    char naziv[16];
    int id; // identifikator
} OGLAS;
```

kojim se reprezentuju informacije o jednom oglasu. Napisati funkciju koja niz od *n* oglasa sortira po nazivu u rastućem redoslijedu korištenjem *insert-sort* algoritma, a čiji je prototip:

void sortiraj(OGLAS *niz, int n);

Napisati nerekurzivnu funkciju koja vrši binarno pretraživanje (po identifikatoru oglasa) niza oglasa. Funkcija vraća adresu pronađenog podatka ili NULL ako oglas sa datim identifikatorom ne postoji. Prototip funkcije je:

OGLAS* trazi(OGLAS *niz, int n, int id);

❸ **(20 bodova)** Neka je definisan tip:

```
typedef struct cvor {
    char *podatak;
    struct cvor *sljedeci;
} CVOR;
```

kojim se reprezentuje čvor jednostruko povezane uređene liste čiji je informacioni sadržaj (dinamički) string.

Napisati funkciju koja dodaje novi string u listu tako da je poredak elemenata u listi uvijek u opadajućem redoslijedu (leksikografski). Prototip funkcije je:

void dodaj(CVOR **pg, char *string);

Napisati rekurzivnu funkciju koja računa i vraća ukupan broj karaktera (parametar *a*) u svim čvorovima liste. Prototip funkcije je:

int prebroji(CVOR *g, char a);

❹ **(20 bodova)** Neka je definisan tip:

```
typedef struct node {
    int info;
    struct node *next;
} NODE;
```

kojim se reprezentuje čvor jednostruko ulančane liste.

Neka je definisan tip:

```
typedef struct graph {
    int n;    // broj cvorova
    NODE **al; //liste susjednosti tj. niz od
    //n ulancanih listi
} GRAPH;
```

koji predstavlja ulančanu reprezentaciju usmjerenog grafa kod kojeg liste susjednosti ne moraju biti uređene.

Napisati funkciju koja kreira i vraća novi dinamički graf. Graf se inicijalizuje brojem čvorova *n*, pri čemu graf inicijalno nema grane. Prototip funkcije je:

GRAPH* g_create(int n);

Napisati funkciju koja provjerava i vraća informaciju o tome da li u grafu *g* postoji grana između čvorova čiji su indeksi *u* i *v* (0 – ne postoji, 1 – postoji). Prototip funkcije je:

int g_check_edge(const GRAPH *g, int u, int v);

Napisati funkciju koja u graf *g* dodaje novu granu između čvorova čiji su indeksi *u* i *v*. Ako ta grana već postoji u grafu, ignorisati pokušaj dodavanja grane. Prototip funkcije je:

void g_add_edge(GRAPH *g, int u, int v);