ELEKTROTEHNIČKI FAKULTET
BANJA LUKA

PROGRAMIRANJE II
(01.09.2021)

❶ (**10 bodova**) Neka je dat tip:

```
typedef struct point{
    double x, y;
} POINT;
```

kojim se reprezentuje tačka u dvodimenzionom koordinatnom sistemu.

Napisati funkciju sa promjenljivim brojem parametara koja formira i vraća dinamički niz tačaka. Funkcija prihvata obavezan parametar *n* koji predstavlja dimenziju niza, a zatim *n* tačaka (podaci tipa POINT) koji predstavljaju elemente niza. Prototip funkcije je:

POINT* point_init(int n, ...);

Napisati funkciju koja u izlaznu tekstualnu datoteku čiji je naziv prvi parametar funkcije, upisuje niz od *n* tačaka. Izlazna tekstualna datoteka treba da bude formatirana tako da je u prvi red upisan ukupan broj tačaka, a zatim u svakom narednom redu datoteke treba da bude upisan podatak o jednoj tački u formatu:

(x,y)

gde su *x* i *y* koordinate tačke.

Funkcija treba da vrati vrijednost 1 ako su podaci o tačkama uspješno upisani u datoteku. U suprotnom, funkcija treba da vrati vrijednost 0.

Prototip funkcije je:

int point_write(const char *file, POINT *arr, int n);

❷ (**10 bodova**) Neka je dat tip:

```
typedef struct predstava {
    char naslov[128];
    char autor[128];
} PREDSTAVA;
```

kojim se reprezentuju podaci o jednoj predstavi.

Napisati funkciju koja niz od *n* podataka o predstavama sortira po naslovu u rastućem redoslijedu korištenjem *shell-sort* algoritma, a čiji je prototip:

void sortiraj(PREDSTAVA *niz, int n);

Napisati nerekurzivnu funkciju koja vrši interpolaciono pretraživanje (po naslovu) niza podataka o predstavama. Funkcija vraća adresu pronađenog podatka ili NULL ako predstava sa datim naslovom ne postoji. Prototip funkcije je:

PREDSTAVA* trazi(PREDSTAVA *niz, int n, const char *naslov);

❸ **(10 bodova)** Neka je dat tip:

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

kojim se reprezentuje čvor jednostruko povezane liste.

Neka je dat tip:

```
typedef struct red {
    char *niz[100];
    int f, r;
} RED;
```

kojim se reprezentuje sekvencijalna reprezentacija kružnog bafera, sa prepisivanjem.

Napisati funkciju koja kreira i kao rezultat vraća novu listu sa informacionim sadržajem onih čvorova originalne liste koji ispunjavaju određeni uslov. Uslov je ispunjen ukoliko pokazivač na funkciju (parametar *criteria*) koji je proslijeđen kao parametar funkcije vraća vrijednost 1. Parametar *head* je glava originalne liste. Potrebno je implementirati i sve pomoćne funkcije. Prototip funkcije je:

NODE* add(NODE* head, int (*criteria)(NODE *node));

Napisati funkciju koja kreira i popunjava kružni bafer sa informacionim sadržajem čvorova liste. Potrebno je implementirati i sve pomoćne funkcije. Parametar *head* je glava liste. Prototip funkcije je:

RED* add(NODE* head);

❹ **(10 bodova)** Definisati i koristiti tip GRAPH čija je implementacija vremenski najefikasnija za rješavanje problema opisanog u funkciji *exists*.

Napisati funkciju koja provjerava da li u grafu (parametar *graph*) postoji grana između dva čvora koji su zadati kao parametri funkcije. Pri tome su drugi i treći parametar indeksi čvorova datog grafa. Funkcija vraća vrijednost 1 ako grana postoji. U suprotnom, funkcija treba da vrati vrijednost 0. Prototip funkcije je:

int exists(const GRAPH* graph, int firstNode, int secondNode);

Napisati funkciju koja provjerava (korištenjem prethodno definisane funkcije *exists*) da li je graf neusmjeren. Funkcija vraća vrijednost 1 ako je graf neusmjeren. U suprotnom, funkcija treba da vrati vrijednost 0. Prototip funkcije je:

int undirected(const Graph* graph);