

## Algoritmi i strukture podataka - 2. međuispit

2. svibnja 2011.

Odgovore na prva četiri pitanja napišite na svojim papirima i predajte ih u košuljici. Odgovore na ostala pitanja napišite na za to predviđenom mjestu uz zadatke.

Nije dopušteno korištenje globalnih i statičkih varijabli te naredbe **goto**.

### Zadatak 1. (7 bodova)

Zadan je tip podatka **Stog** za koji su definirane funkcije za inicijalizaciju stoga, dodavanje elementa u stog te za skidanje elementa iz stoga. Prototipovi navedenih funkcija su:

```
void init_stog(Stog *stog);
int dodaj(double element, Stog *stog);
int skini(double *element, Stog *stog);
```

Funkcije **dodaj** i **skini** vraćaju 1 ako je operacija uspjela, a 0 ako nije.

Napišite funkciju čiji je prototip:

```
void preurediStog(Stog *stog);
```

koja će preurediti poredak elemenata u zadanom početnom stogu tako da sva pojavljivanja najmanjeg elementa stavi na dno stoga, dok se poredak ostalih elemenata ne mijenja.

Napišite i glavni program koji iz neformatirane datoteke "**podaci.dat**" čita brojeve i pohranjuje ih u stog te poziva funkciju **preurediStog**.

Primjer: Stog s elementima (od vrha prema dnu) **3.1, 2.7, 4.5, 4.8, 4.5, 3.1, 2.7, 4.9** funkcija će promijeniti u **3.1, 4.5, 4.8, 4.5, 3.1, 4.9, 2.7, 2.7**.

### Zadatak 2. (6 bodova)

Zadan je tip podatka **Red** za koji su definirane funkcije za inicijalizaciju reda, dodavanje elementa u red te za skidanje elementa iz reda. Prototipovi navedenih funkcija su:

```
void init_red(Red *red);
int dodaj(int element, Red *red);
int skini(int *element, Red *red);
```

Funkcije **dodaj** i **skini** vraćaju 1 ako je operacija uspjela, a 0 ako nije.

Napišite funkciju koja će sva pojavljivanja višekratnika zadanog broja iz zadanog reda izdvojiti u novi red koji treba vratiti pozivatelju. Prototip funkcije je:

```
Red *izdvojiVisekratnike(Red *red, int broj);
```

U ulaznom redu moraju ostati svi preostali elementi.

Napišite i dio glavnog programa u kojem ćete definirati redove i ispravno pozvati napisanu funkciju. Dio koda u kojem se puni red nije potrebno pisati.

Primjer: Za ulazni red sadržaja **3, 8, 2, 5, 35, 2, 24, 15** poziv funkcije **izdvojiVisekratnike** za **broj=3** stvara novi red sadržaja **3, 24, 15** dok u starom redu ostaju **8, 2, 5, 35, 2**.

### Zadatak 3. (6 bodova)

Zadan je stog u kojem su pohranjeni podaci o točkama koordinatnog sustava tipa **Točka**, koji će sadržavati dvije realne vrijednosti **x** i **y**. Definirajte tip podataka **Točka**.

Napišite **rekurzivnu** funkciju **tockeNaPravcu** koja će sa zadanog stoga **stog** u red **red** kopirati sve elemente koji leže na pravcu definiranom koeficijentom smjera **a** i odsječkom na osi **y** **b**. Prototip funkcije je:

```
int tockeNaPravcu (Stog *stog, Red *red, float a, float b);
```

Funkcija treba vratiti broj točaka koje leže na zadanom pravcu. Nakon izlaska iz funkcije stog mora ostati očuvan, a poredak elemenata u redu nije bitan.

**Napomena:** Nerekurzivno rješenje neće se priznavati.

### Zadatak 4. (3 boda)

Zadano je polje brojeva s elementima: **8, 6, 4, 9, 2, 7, 5, 0, 3, 1**. Ilustrirajte sortiranje zadanog niza brojeva (ispišite polje nakon svake promjene i označite sve brojeve relevantne za sljedeći korak – zamjene, stožere, aproksimacije medijana) algoritmom **quicksort**. Stožer odaberite metodom aproksimacije medijana temeljem prvog, srednjeg i zadnjeg člana polja.

### Zadatak 5. (2 boda)

Napišite dio programskog koda kojim se deklarira struktura atoma linearne liste koji u podatkovnom dijelu sadrži realni broj dvostruke preciznosti. Skicirajte red realiziran linearnom listom koji sadrži tri atoma s vrijednostima **2.4**, **3.4** i **5.0**.

### Zadatak 6. (2 boda)

Što radi zadana funkcija? Koja je njena apriorna složenost u ovisnosti o broju elemenata stoga **n**?

```
void misterija(int element, Stog *stog){    Odgovor:
    int x;
    if (skiniSaStoga(&x, stog)) {
        misterija (element, stog);
        dodajNaStog(x, stog);
    }
    else {
        dodajNaStog(element, stog);
    }
}
```

Složenost: \_\_\_\_\_

## Rješenja:

1.

```
void preurediStog(Stog *stog){
    Stog pom1, pom2;
    double element, min;
    int br;

    init_stog(&pom1);
    init_stog(&pom2);
    if (skini(&element, stog)){ // Prvo nađem najmanji element
        min=element;
        dodaj(element, &pom1);
        dodaj(element, &pom2);
    }
    else return;
    while(skini(&element, stog)){
        if (element < min) min=element;
        dodaj(element, &pom1);
        dodaj(element, &pom2);
    }
    br=0; // Sada nađem koliko puta se najmanji element pojavljuje
    while(skini(&element, &pom1)){
        if (element==min) br++;
    }
    while(br>0){ // prvo u red zapišem sva pojavljivanja minimalnog elementa...
        dodaj(min, stog);
        br--;
    }
    // ...a zatim i sve ostale elemente, izbjegavajući minimalni
    while(skini(&element, &pom2)){
        if (element!=min) dodaj(element, stog);
    }
}

int main () {
    Stog stog;
    FILE *f;
    double b;
    init_stog(&stog);
    f=fopen("podaci.dat", "rb");
    while(fread (&b, sizeof(double), 1, f)==1){
        staviNaStog(b, &stog);
    }
    fclose(f);
    preurediStog(&stog);
}
```

## 2.

```
Red *IzdvojiVisekratnike(Red *red, int broj){

    Red *pom, pom2;
    int element, temp;

    pom=(Red*) malloc(sizeof(Red));
    init_red(pom);
    init_red(&pom2);

    while(skini(&element, red)){
        if (element % broj==0)
            dodaj(element, pom);
        else dodaj(element, &pom2);
    }
    while(skini(&element, &pom2)){
        dodaj(element, red);
    }
    return pom;
}

int main (){
    Red red, *novi_red;

    init_red(&red);
    /*... napuni red i učitaj broj... */
    novi_red = IzdvojiVisekratnike(&red, 3);
    ...
}
```

## 3.

```
typedef struct {
    float x,y;
} tip;

int tockeNaPravcu(Stog * stog, Red * red, float a, float b){
    float y;
    tip element;
    int brojac=0;

    if (skiniSaStoga(&element, stog)){
        y = a*element.x + b;
        if (y==element.y) {
            dodajURed(element, red);
            brojac=1;
        }
        brojac += tockeNaPravcu(stog, red, a, b);
    }
    else{
        return 0;
    }
    dodajNaStog(element, stog);
    return brojac;
}
```

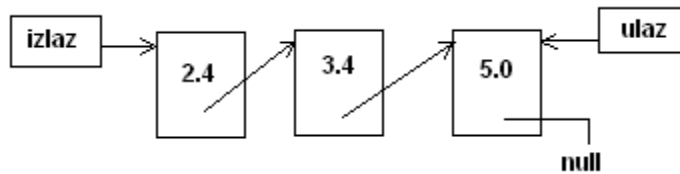
4.

quicksort

8, 6, 4, 9, 2, 7, 5, 0, 3, 1  
1, 6, 4, 9, 2, 7, 5, 0, 3, 8 stožer je 2  
1, 6, 4, 9, 3, 7, 5, 0, 2, 8 stožer sakrijemo  
1, 6, 4, 9, 3, 7, 5, 0, 2, 8 6 i 0 mijenjaju mjesto  
1, 0, 4, 9, 3, 7, 5, 6, 2, 8 i i j su se ukrižali i stožer vraćam na mjesto 4  
1, 0, 2, 9, 3, 7, 5, 6, 4, 8  
0, 1, 2, 9, 3, 7, 5, 6, 4, 8 između ovih biram novi stožer i to je 8  
0, 1, 2, 5, 3, 7, 8, 6, 4, 9  
0, 1, 2, 5, 3, 7, 8, 6, 4, 9 stožer sakrivamo sa predzadnjim elementom  
0, 1, 2, 5, 3, 7, 4, 6, 8, 9 i i j su se ukrižali i ništa se nije mijenjalo  
0, 1, 2, 5, 3, 7, 4, 6, 8, 9 tražimo novi stožer i to je 6  
0, 1, 2, 5, 3, 6, 4, 7, 8, 9  
0, 1, 2, 5, 3, 4, 6, 7, 8, 9 stožer sakrijemo  
0, 1, 2, 5, 3, 4, 6, 7, 8, 9 nije bilo promjena i samo treba sortirat 534  
0, 1, 2, 3, 4, 5, 6, 7, 8, 9

5.

```
struct at {  
    double element;  
    struct at *sljed;  
};
```



6.

Na dno stoga stavlja element,  $O(n)$