Algoritmi i strukture podataka

Završni ispit

23. lipnja 2009.

Napomena za sve zadatke:
- Nije dopušteno korištenje naredbe **goto** te statičkih i globalnih varijabli.

Zadatak 1. *(8 bodova)*

U jednostruko povezanu nesortiranu listu spremljeni su podaci o studentima. Lista je zadana sljedećim strukturama:

```
struct zapis {
  char imeprezime[80+1];
  int ocjena;
};
```

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

typedef struct at atom;
```

Napišite funkciju koja će iz zadane liste sve čvorove koji sadrže studente čija je ocjena veća ili jednaka prosjeku ocjena svih studenata u listi prebaciti **(bez stvaranja novih čvorova)** u novu jednostruko povezanu listu. Funkcija treba vratiti pokazivač na glavu nove liste.
Funkcija mora imati prototip:

```
atom * razdvojiListe(atom **glava);
```

Napomena: U izvornoj listi na koju pokazuje `glava` ostaju čvorovi koji sadrže studente čija je ocjena manja od prosjeka ocjena svih studenata.

Primjer: `glava` prije poziva funkcije pokazuje na listu čvorova koji sadrže sljedeće ocjene: `5, 2, 4, 3, 4, 3, 2, 4, 5, 1.`

Prosjek ocjena je 3,3 pa nakon poziva funkcije `razdvojiListe` varijabla `glava` pokazuje na listu čvorova koji sadrže sljedeće ocjene: 2, 3, 3, 2, 1. Funkcija vraća pokazivač na novu listu koja sadrži čvorove sa sljedećim ocjenama: 5, 4, 4, 4, 5.

Zadatak 2. *(7 bodova)*

Binarno stablo sadrži cjelobrojne elemente te je čvor definiran sljedećim kodom:

```
typedef struct s{
   int x;
   struct s *lijevo, *desno;
} cvor;
```

Zadana je funkcija

```
int magicnaTrojka(int a, int b, int c);
```

koja vraća 1 ako `a`, `b` i `c` čine magičnu trojku, a 0 inače.

Napišite funkciju koja će vratiti koliko čvorova u stablu čini magičnu trojku sa svojom djecom. Prototip tražene funkcije je:

```
int brojMagicnih(cvor *korijen);
```

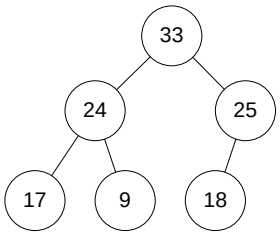
Zadatak 3. *(8 bodova)*

Zadan je niz brojeva: 12, 5, 7, 1, 50, 22, 5, 43.

a) *(3 boda)* Ilustrirajte (nacrtajte stablo nakon **svake** promjene) stvaranje gomile od zadanog polja brojeva algoritmom čija je složenost za najgori slučaj **O(n).**

b) *(3 boda)* Ilustrirajte (nacrtajte stablo nakon **svake** promjene) stvaranje gomile od zadanog polja brojeva algoritmom čija je složenost za najgori slučaj **O(nlogn).**

c) *(2 boda)* Zadana je gomila:



Ilustrirajte izgled gomile nakon **svake** promjene dva čvora, za prva 3 koraka uzlaznog *heap sorta*.

Zadatak 4. *(7 bodova)*

Implementirajte razred *Kugla* s dvije (privatne) varijable jednostruke preciznosti koje predstavljaju polumjer kugle i gustoću materijala od kojeg je kugla napravljena.

Implementirajte metode za dohvat i postavljanje vrijednosti polumjera i gustoće.

Implementirajte i dvije metode koje računaju volumen i masu kugle.

Napišite odsječak glavnog programa u kojem instancirajte objekte na dva načina:

- na stogu,
- na hrpi (engl. *heap*).

Demonstrirajte njihovo korištenje pozivima svih metoda obaju objekata.

Za instanciranje objekata koristite podrazumijevani konstruktor (**ne trebate pisati vlastite konstruktore**).

Formule:

$$masa = volumen * gustoca$$

$$volumen = \frac{4}{3} * polumjer^3 * \pi$$

Rješenja

1. zadatak

```
atom * razdvojiListu (atom **glava) {
  float prosjek, suma=0;
  int brojac=0;
  atom *novi, *p, *zap=NULL, *preth;
  novi=NULL;
  for (p = *glava; p; p = p->sljed){
            suma+=p->element.ocjena;
            brojac++;
  }
  if (brojac==0) return novi;
  prosjek=suma/brojac;
  for (p = *glava; p; p = p->sljed){
        if (p->element.ocjena>=prosjek) {
              if (novi==NULL)
                novi=p;
              else
                   zap->sljed=p;
              zap=p;
              if (p==*glava)
                    *glava = (*glava)->sljed;
              else{
                  preth->sljed=p->sljed;
                  p=preth;
              }
        }
        preth=p;
  }
 if (zap) zap->sljed=NULL;
  return novi;
}
```

2. zadatak

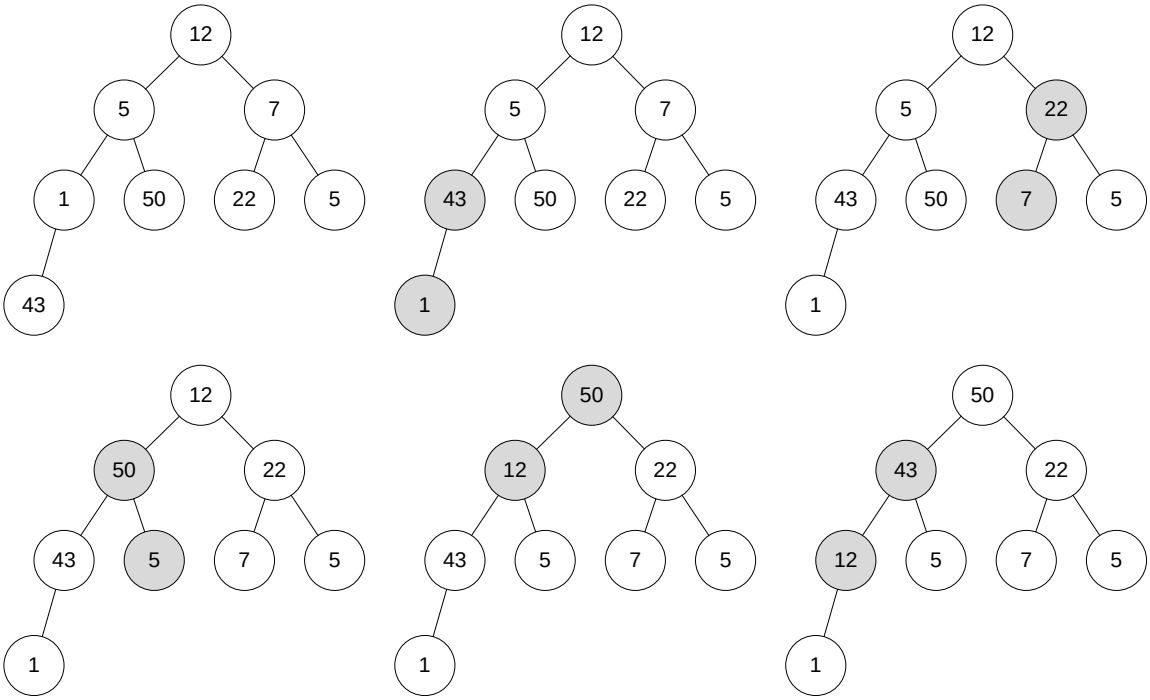
```
int brojMagicnih(cvor *korijen){
  int mt = 0;
  if (korijen == NULL)
  {
    return 0;
  }
  if (korijen->lijevo != NULL && korijen->desno != NULL)
  {
    mt = magicnaTrojka(korijen->x, korijen->lijevo->x, korijen->desno->x);
  }
  return mt + brojMagicnih(korijen->lijevo) + brojMagicnih(korijen->desno);
}
```

ili

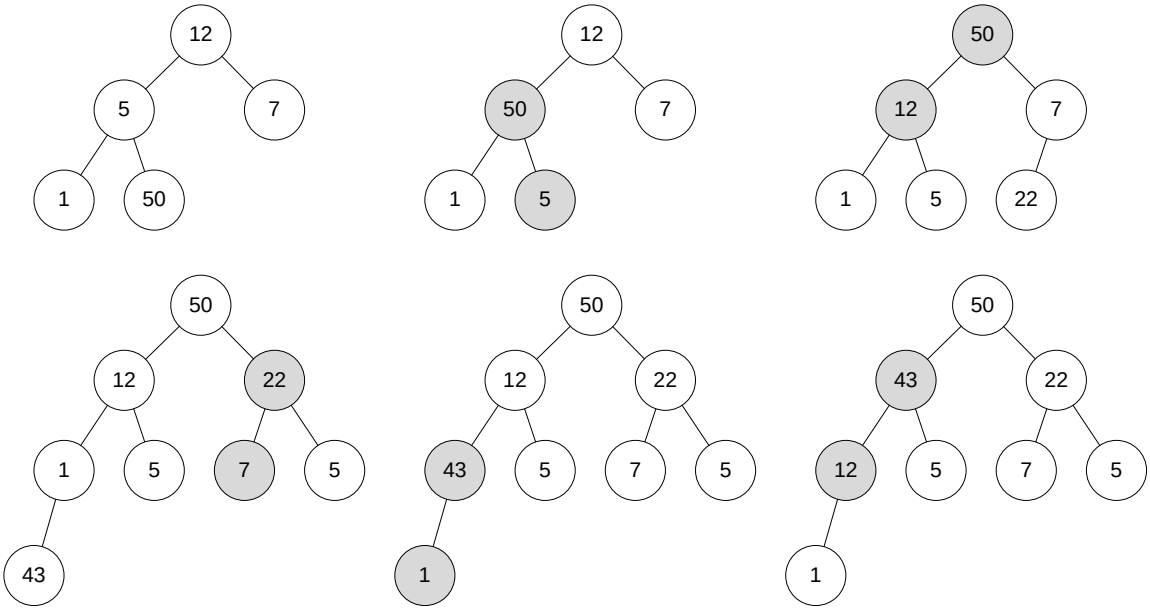
```
int brojMagicnih(cvor *korijen){
  if (korijen == NULL)
  {
       return 0;
  }
  else if (korijen->lijevo != NULL && korijen->desno != NULL)
  {
       return magicnaTrojka(korijen->x, korijen->lijevo->x,
                    korijen->desno->x) +
              brojMagicnih(korijen->lijevo) + brojMagicnih(korijen->desno);

  }
  else
  {
       return brojMagicnih(korijen->lijevo) + brojMagicnih(korijen->desno);
  }
}
```

3. zadatak

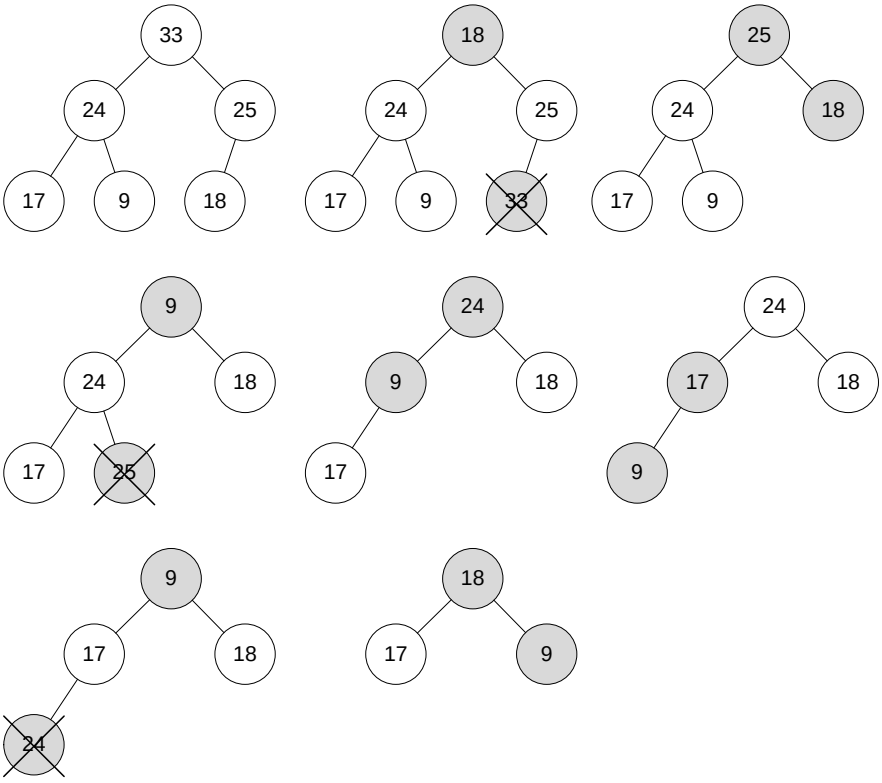
a)


12
5
7
1
50
22
5
43
12
5
7
43
50
22
5
1
12
5
22
43
50
7
5
1
12
50
22
43
5
7
5
1
50
12
22
43
5
7
5
1
50
43
22
12
5
7
5
1

b)


12
5
7
1
50
12
50
7
1
5
50
12
7
1
5
22
50
12
22
1
5
7
5
43
50
12
22
43
5
7
5
1
50
43
22
12
5
7
5
1

c)


33
24
25
17
9
18
18
24
25
17
9
33
25
24
18
17
9
9
24
18
17
25
24
9
18
17
24
17
18
9
9
17
18
24
18
17
9

4. zadatak

```
#include <stdio.h>
#include <math.h>

const float PI=3.14;

class Kugla
{
private:
	float polumjer, gustoca;

public:
	void SetPolumjer(float polumjer)
	{
		this->polumjer=polumjer;
	}
	float GetPolumjer()
	{
		return polumjer;
	}
	void SetGustoca(float gustoca)
	{
		this->gustoca=gustoca;
	}
	float GetGustoca()
	{
		return gustoca;
	}

	float volumen()
	{
		return 4./3*pow(polumjer,3)*PI;
	}

	float masa(){

		return volumen()*gustoca;
	}
};



int main(){
	Kugla k1;
	Kugla *k2= new Kugla();
	k1.SetGustoca(2.1);
	k1.SetPolumjer(3);
	k2->SetGustoca(11.9);
	k2->SetPolumjer(8);
	printf("kugla 1:\n radijus=%f\ngustoca=%f\nvolumen=%f\nmasa=%f\n",
		k1.GetPolumjer(),k1.GetGustoca(),k1.volumen(),k1.masa());
	printf("kugla 2:\n radijus=%f\ngustoca=%f\nvolumen=%f\nmasa=%f\n",
		k2->GetPolumjer(),k2->GetGustoca(),k2->volumen(),k2->masa());
	delete k2;
	return 0;
}
```