Algoritmi i strukture podataka –1. ispitni rok

2. srpnja 2015.

*Nije dopušteno korištenje globalnih i statičkih varijabli te naredbe **goto**. Neefikasna rješenja mogu donijeti manje bodova.Nerekurzivne funkcije se ne priznaju kao rješenja u zadacima u kojima se traži rekurzivna funkcija i obratno.*
*Ispit nosi maksimalno 70 bodova, a prag za prolaz pismenog ispita je **35** bodova.*

Zadatak 1. (22 boda)

Tablica raspršenog adresiranja sastoji se od primarnog područja veličine **P** pretinaca i preljevnog područja veličine **S** pretinaca. U svakom pretincu nalazi se **C** zapisa definiranih strukturom

```
typedef struct {
    int sifra;              //ključ za određivanje pretinca
    struct Podaci podaci;   //podaci koji se čuvaju
} zapis;
```

Zapis je prazan ako je **`sifra = 0`**, a podrazumijeva se da su u pretincu svi prazni zapisi (ako ih ima) smješteni na kraju pretinca. Preljevi su realizirani upisom u prvi slobodni pretinac u preljevnom području. Transformacija ključa u adresu obavlja se zadanom funkcijom **`int adresa(int kljuc)`** koja vraća cijeli broj između 0 i P-1 (broj pretinca u koji bi zapis trebao ići).

Napravite funkciju **`int brisiZapis(int sifra, FILE *f)`** koja briše zapis i pri tome čuva strukturu pretinca, tj. čuva poredak zapisa unutar pretinca tako da prazni zapisi budu na kraju pretinca. Funkcija vraća 1, ako je zapis pronađen i obrisan, a 0 ako nije.

Veličina bloka na disku je BLOK, a svi potrebni parametri i struktura definirani su u `hash.h` datoteci zaglavlja.

Zadatak 2. (18 bodova)

Napisati funkciju koja će od dva stabla napraviti novo u kojem elementi stabla čine zbroj elemenata prvog stabla i elemenata drugog stabla. Ako jedno od ulaznih stabala nema neki čvor koje ima drugo stablo zbroj čini samo čvor iz postojećeg stabla (nepostojeći čvor ima vrijednost 0 kod zbrajanja).

Čvor stabla zadan je strukturom:

```
typedef struct cv {
    int vrijednost;
    struct cv *lijevo;
    struct cv *desno;
} cvor;
```

Prototip funkcije je:

```
cvor *zbroji (cvor *s1, cvor *s2);
```

Ulazna stabla moraju ostati nepromijenjena.

Zadatak 3. (15 bodova)

Zadano je polje **a** s ukupno **n** elemenata čiji su elementi pozitivni cijeli brojevi. Potrebno je napisati rekurzivnu funkciju **minNepar** koja će naći najmanji neparni broj u polju **a,** ako takav postoji, a u suprotnom vraća -1. Prototip funkcije je

```
int minNepar(int a[], int n);
```

Nerekurzivno rješenje se neće priznavati.

Zadatak 4. (10 bodova)

Napisati funkciju koja će iz jednostruko povezane linearne liste izbrisati sve čvorove koji imaju vrijednost manju ili jednaku prosječnoj vrijednosti svih prethodnih elemenata u listi (računajući i eventualno izbačene te trenutni element koji je kandidat za brisanje). Prototip funkcije je **`void brisi (atom **glava)`**.

Prilikom brisanja nije potrebno oslobađati memoriju obrisanih elemenata, već ih samo izbaciti iz liste.

Atom liste zadan je strukturom:

```
typedef struct at {
    int elem;
    struct at *sljed;
} atom;
```

Primjer: za listu sa elementima navedenima ispod funkcija izbacuje precrtane elemente (glava liste je prvi element slijeva):

~~1~~ 5 ~~3~~ 7 9 ~~2~~ 6 ~~0~~ ~~3~~ ~~2~~

Zadatak 5. (5 bodova)

Napisati preorder ispis binarnog stabla za pretraživanje kod kojeg je lijevo dijete manje od roditelja, a desno veće i čiji je postorder ispis zadan nizom:

1, 6, 7, 5, 4, 11, 8, 3

U oba ispisa podrazumijeva se ispis s lijeva na desno.

Rješenja:

Zadatak 1. (22 boda)

```
int brisiZapis(int sifra, FILE *f) {
  zapis pretinac[C];
  int i,j;
  int adr=adresa(sifra);

  // pogledaj je li zapis u primarnom području
  fseek (f, adr*BLOK, SEEK_SET);
  fread (pretinac, sizeof (pretinac), 1, f);
  for(i=0; i<C; i++){
    if(pretinac[i].sifra==sifra){
      for (j=C-1; j>i; j--){
        if (pretinac[j].sifra != 0){
          pretinac[i].sifra=pretinac[j].sifra;
          pretinac[i].podaci=pretinac[j].podaci;
          pretinac[j].sifra=0;
          fseek (f, adr*BLOK, SEEK_SET);
          fwrite (pretinac, sizeof (pretinac), 1, f);
          return 1;
        }
      }
    }
  }

  // sada ista stvar u preljevnom području
  for(k=P; k<P+S; k++){
    fseek (f,k*BLOK, SEEK_SET);
    fread (pretinac, sizeof (pretinac), 1, f);
    for(i=0; i<C; i++){
      if(pretinac[i].sifra==sifra){ // nasao sam pravi zapis, treba ga izbrisati ...
        for (j=C-1; j>i; j--){
          if (pretinac[j].sifra != 0){
            pretinac[i].sifra=pretinac[j].sifra;
            pretinac[i].podaci=pretinac[j].podaci;
            pretinac[j].sifra=0;
            fseek (f, k*BLOK, SEEK_SET);
            fwrite (pretinac, sizeof (pretinac), 1, f);
            return 1;
          }
        };
      };
    }
  }
  return 0;
}
```

Zadatak 2. (18 bodova)

```
cvor *zbroji(cvor *s1, cvor *s2) {
        cvor *d1 = NULL, *d2 = NULL, *l1 = NULL, *l2 = NULL, *c;
        int v1 = 0, v2 = 0;
        if (s1 == NULL && s2 == NULL) return NULL;
        if (s1) {
                d1 = s1->desno; l1 = s1->lijevo; v1 = s1->vrijednost;
        }
        if (s2) {
                d2 = s2->desno; l2 = s2->lijevo; v2 = s2->vrijednost;
        }
        c = (cvor*)malloc(sizeof(cvor));
        c->vrijednost = v1 + v2;
        c->lijevo = zbroji(l1, l2);
        c->desno = zbroji(d1, d2);
        return c;
}
```

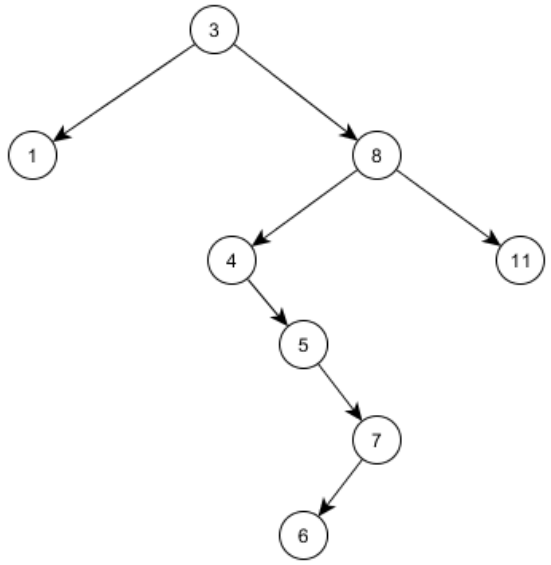
Zadatak 3. (15 bodova)

```
int minNepar(int* a, int n) {
	int znamenka, vrati, min;
	if (n==0) return -1;		// ako nema elemenata polja, vrati -1

	if (a[n-1] % 2==1){		 // ako je zadnji element polja neparan
		min=a[n-1];		 // onda je trenutno najmanji
	}
	else min=-1;
	vrati=minNepar(a,n-1);		// što vraća funkcija?
	if (vrati==-1) return min; // ako nema neparnih, onda je trenutni najmanji, najmanji
	if (min==-1) return vrati; // ako je trenutno najmanji -1, onda nastavi s onim što je vraćeno

	if (min<vrati){			// ako obje vrijednosti postoje, usporedi ih i vrati što treba.
		return min;
	}
	else return vrati;
}
```

Zadatak 4. (10 bodova)

```
void brisi(atom **glava) {
	int sum = 0, br = 0;
	while (*glava){
		br++;
		sum += (*glava)->elem;
		if ((*glava)->elem <= ((float)sum / br)){
			*glava = ((*glava)->sljed);
		}
		else
		{
			glava = &((*glava)->sljed);
		}
	}
}
```

Zadatak 5. (5 bodova)



Tražen je samo ispis, stablo je priloženo kao pomoć pri analizi.

Ispis:
3, 1, 8, 4, 5, 7, 6, 11