JMBAG **IME I PREZIME**

Algoritmi i strukture podataka –2. ispitni rok

1. rujna 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.*
*Ispit nosi maksimalno 70 bodova, a prag za prolaz pismenog ispita je **35** bodova.*

Zadatak 1. (15 bodova)

Jedan zapis datoteke organizirane po načelu raspršenog adresiranja definiran je strukturom:

```
typedef struct{
      int sifra;
      char naziv[100+1];
} zapis;
```

Zapis je prazan ako je na mjestu šifre (članska varijabla `sifra`) vrijednost nula. Parametri za raspršeno adresiranje nalaze se u datoteci *parametri.h* i oni su:

- BLOK ..... veličina bloka na disku
- N ..... broj zapisa
- C ..... broj zapisa u jednom pretincu
- M ..... broj pretinaca

Preljevi su realizirani ciklički, upisom u prvi sljedeći slobodni pretinac. Ključ zapisa je šifra, a transformacija ključa u adresu obavlja se zadanom funkcijom

```
int adresa(int sifra);
```

Napišite funkciju koja će naći sve potpuno popunjene pretince i za svaki takav pretinac odrediti postotak zapisa koji su u pretinac zapisani kao preljev. Funkcija u pozivajući program mora vratiti postotke preljeva u tim pretincima te broj potpuno popunjenih pretinaca. Prototip funkcije je:

```
void statistika(char *imeDatoteke, float *postotciPreljeva, int *brPunihPret);
```

gdje je:

- `imeDatoteke` – ime datoteke organizirane po načelu raspršenog adresiranja
- `postotciPreljeva` – cjelobrojno polje u koje funkcija mora pohraniti postotke preljeva
- `brPunihPret` – broj potpuno popunjenih pretinaca

Ako među M pretinaca postoji samo P potpuno popunjenih, funkcija treba upisati njihove postotke preljeva u prvih P elemenata polja `postotciPreljeva`. Pretpostaviti da na lokaciji na koju pokazuje `postotciPreljeva` postoji dovoljno mjesta za upis M elemenata.

Zadatak 2. (15 bodova)

Napišite rekurzivnu funkciju `samoUzlazno` čiji je prototip:

```
int samoUzlazno(int n);
```

Funkcija za zadani pozitivni cijeli broj *n* vraća broj koji je načinjen od onih znamenaka broja *n* koje su (gledano slijeva na desno) u uzlaznom poretku, pri čemu se poredak znamenaka (kakav je bio u *n*) ne smije mijenjati.

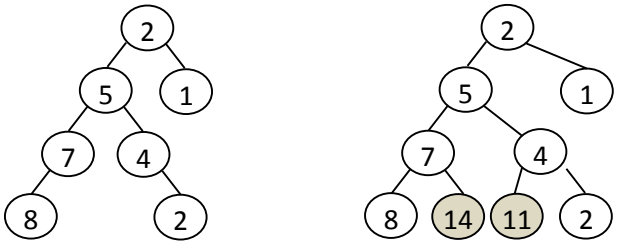
Primjer: za broj 175739 treba vratiti 179, za broj 21003 treba vratiti 23, a za 2062 treba vratiti 26.

Zadatak 3. (18 bodova)

Čvorovi binarnog stabla definirani su strukturom **cvor**:

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

Napišite funkciju **nadopuni** koja čvorovima stabla koji imaju samo jedno dijete dodaje drugo dijete, koje je list, a vrijednost mu je jednaka zbroju vrijednosti svih čvorova na putu od tog djeteta do korijena (uključujući i vrijednost korijena). Primjer nadopune na slici:



Zadatak 4. (10 bodova)

Odredite a priori složenost (u O-notaciji) zadanih odsječaka programskog koda uz pretpostavku konstantne asimptotske složenosti funkcije f(n) i da je n >> 1. Obrazložite odgovor.

a) (**4 boda**)

```
a = 0;
for(i=1; i<=n; i++) {
  for(j=1; j<=i; j++) {
    a += f(j);
  }
}
```

b) (**6 bodova**)

```
a = 0;
for(i=1; i<=n; i++) {
  for(j = pow(n, 1/(i*i)); j>=1; j/=2) {
    a += f(i);
  }
}
```

Naputak: $\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \approx 1{,}64$

Zadatak 5. (12 bodova)

U cjelobrojnom polju pohranjen je sljedeći niz brojeva:

12, 1, 10, 7, 11, 4, 2, 8, 14, 15, 5

a) (**6 bodova**) Ilustrirajte uzlazno sortiranje algoritmom Shellsort s koracima {5, 2, 1}.

b) (**6 bodova**) Ilustrirajte uzlazno sortiranje algoritmom Quicksort. Stožer za Quicksort bira se metodom aproksimacije medijana temeljem prvog, srednjeg i zadnjeg člana, pri čemu vrijedi da je `cutoff = 3` nakon čega se sortira bez navođenja koraka.

Potrebno je prikazati sadržaj polja nakon svake promjene.

Rješenja:

Zadatak 1. (15 bodova)

```
void statistika(char *imeDatoteke, float *postotciPreljeva, int *brPunihPret) {
    FILE *fUlaz;
    zapis z, pretinac[C];
    int i, j, brZapisaUPretincu, brPreljevaUPretincu;
    /*Otvaranje datoteke*/
    fUlaz = fopen(imeDatoteke, "rb");
    /*Inicijalizacija brojaca*/
    *brPunihPret = 0;
    /*Prolaz kroz sve pretince*/
    for (i = 0; i < M; i++) {
        fseek(fUlaz, i * BLOK, SEEK_SET);
        fread(pretinac, sizeof(pretinac), 1, fUlaz);
        brZapisaUPretincu = 0;
        brPreljevaUPretincu = 0;
        /*Prolaz kroz sve zapise u pretincu*/
        for (j = 0; j < C; j++) {
            if (pretinac[j].sifra == 0) {
                /*Prazan zapis - pretinac nije pun - izlazak iz petlje*/
                break;
            } else if (adresa(pretinac[j].sifra) != i) {
                /*Uvecaj brojac preljeva*/
                brPreljevaUPretincu++;
            }
            /*Uvecaj brojac zapisa u pretincu*/
            brZapisaUPretincu++;
        }
        /*Provjera je li pretinac pun*/
        if (brZapisaUPretincu == C) {
            /*Izracun postotka zapisa koji su u pretinac zapisani kao preljev*/
            postotciPreljeva[*brPunihPret]=(brPreljevaUPretincu/(float)C)*100.0f;
            /*Uvecaj brojac punih pretinaca*/
            (*brPunihPret)++;
        }
    }
    /*Zatvaranje datoteke*/
    fclose(fUlaz);
}
```

Zadatak 2. (15 bodova)

```
int samoUzlazno(int n){
      int tmp,zn;
      if(n<10) return n;

      zn=n % 10;
      tmp=samoUzlazno(n/10);
      if (tmp % 10 < zn) tmp=tmp*10+zn;
      return tmp;
}
```

Zadatak 3. (18 bodova)

```
void nadopuni(cvor *korijen, int tmp){
	cvor *novi;

	if (!korijen) return;
	if (!korijen->lijevo && !korijen->desno) return;

	tmp +=korijen->vrijednost;
	if (korijen->lijevo && korijen->desno){
		nadopuni(korijen->lijevo,tmp);
		nadopuni(korijen->desno,tmp);
	}
	else{
		novi=(cvor *) malloc(sizeof(cvor));
		novi->lijevo=novi->desno=NULL;
		novi->vrijednost=tmp;
		if (!korijen->lijevo){
			nadopuni(korijen->desno,tmp);
			korijen->lijevo=novi;
		}
		else{
			nadopuni(korijen->lijevo,tmp);
			korijen->desno=novi;
		}
	}
}
```

Zadatak 4. (10 bodova)

(Zadatak sličan zadatku od prije 2 godine, izazov za studente je da nisu baš napament naučili ono rješenje)

a) $O(n^2)$

b) $O(n) + O(\log(n^{1/1}) + \log(n^{1/4}) + \log(n^{1/9}) + \cdots) = O(n) + O\left(\left(1 + \frac{1}{4} + \frac{1}{9} + \cdots\right)\log(n)\right) = O(n) + O(\log(n)) = O(n)$

Zadatak 5. (12 bodova)

Masnim su označeni elementi koji ulaze u iduću zamjenu. Studenti mogu označiti te elemente ili pri svakoj zamjeni upravo zamijenjene elemente ili ništa od toga.

Shellsort
korak: 5
12, 1, 10, 7, 11, **4**, 2, 8, 14, 15, **5**
4, 1, **10**, 7, 11, 12, 2, **8**, 14, 15, 5
4, 1, 8, 7, 11, **12**, 2, 10, 14, 15, **5**
korak: 2
4, 1, 8, **7**, 11, **5**, 2, 10, 14, 15, 12
4, 1, 8, 5, **11**, 7, **2**, 10, 14, 15, 12
4, 1, **8**, 5, **2**, 7, 11, 10, 14, 15, 12
4, 1, **2**, 5, 8, 7, 11, 10, 14, 15, 12
2, 1, 4, 5, 8, 7, 11, 10, **14**, 15, **12**
korak: 1
2, **1**, 4, 5, 8, 7, 11, 10, 12, 15, 14
1, 2, 4, 5, **8**, **7**, 11, 10, 12, 15, 14
1, 2, 4, 5, 7, 8, **11**, **10**, 12, 15, 14
1, 2, 4, 5, 7, 8, 10, 11, 12, **15**, **14**
1, 2, 4, 5, 7, 8, 10, 11, 12, 14, 15

Quicksort
12, 1, 10, 7, 11, **4**, 2, 8, 14, 15, **5** odabir stozera
4, 1, 10, 7, 11, **5**, 2, 8, 14, **15**, 12 sklanjanje stozera
4, 1, **10**, 7, 11, 15, **2**, 8, 14, 5, 12 i→2; 6←j; zamjena
4, 1, 2, **7**, 11, 15, 10, 8, 14, **5**, 12 i→3>=2←j, obnova stožera
4, 1, 2, 5, 11, 15, 10, 8, 14, 7, 12 |L| <= cutoff, sortiraj "napamet"
1, 2, 4, 5, **11**, 15, 10, **8**, 14, 7, **12** odabir stozera
1, 2, 4, 5, 8, 15, 10, **11**, 14, **7**, 12 sklanjanje stozera
1, 2, 4, 5, 8, **15**, 10, **7**, 14, 11, 12 i →5;7←j; zamjena
1, 2, 4, 5, 8, 7, 10, **15**, 14, **11**, 12 i→7 >= 6 ←j, obnova stožera
1, 2, 4, 5, **8, 7, 10**, 11, 14, 15, 12 |LD| <= cutoff, sortiraj "napamet"
1, 2, 4, 5, 7, 8, 10, 11, **14, 15, 12** |DD| <= cutoff, sortiraj "napamet"
1, 2, 4, 5, 7, 8, 10, 11, 12, 14, 15