1. međuispit iz predmeta Algoritmi i strukture podataka

29. ožujka 2010.

Odgovore na prva tri 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. *(4 boda)*

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

```
typedef struct{
      int sifra;
      char naziv[50+1];
      double cijena;
} zapis;
```

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

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

Ključ zapisa je šifra artikla, a transformacija ključa u adresu obavlja se zadanom funkcijom:

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

Zadana je i funkcija **`upisiZapis`** koja zadani zapis upisuje u zadani pretinac ako u njemu ima mjesta.

```
int upisiZapis (FILE *f, zapis z, int adresaPretinca);
```

Funkcija vraća 1 ako je mjesto pronađeno i zapis zapisan, a u suprotnom vraća 0.

Napisati funkciju **`urediHash`** koja će preljeve s pomoću funkcije **`upisiZapis`** prebaciti u za njih predviđene pretince ako tamo ima mjesta. Dovoljno je kroz sve pretince proći samo jednom (već provjerene preljeve ne treba ponovno provjeravati). Funkcija treba vratiti broj prebačenih preljeva. Prototip funkcije treba biti:

```
int urediHash(FILE *f);
```

Zadatak 2. *(3 boda)*

Napisati **rekurzivnu** funkciju za određivanje je li broj prost.
Prototip funkcije je:

```
short jeProst(int broj, int djelitelj);
```

Funkcija vraća 1 ako je broj prost, inače vraća 0. Pri prvom pozivu funkcije djelitelj je postavljen na 2.

Napomena: Nerekurzivno rješenje neće se priznavati.

Zadatak 3. *(3 boda)*

Napisati funkciju koja će iz ulaznog polja cijelih brojeva izbaciti sve one manje od prosjeka, a zatim i dinamički smanjiti veličinu polja na osnovu novog broja članova. Funkcija kao rezultat vraća pokazivač na navedeno polje. Prototip funkcije je:

```
int *izbaci_podprosj(int *polje, int *br_clanova);
```

Napisati i glavni program u kojem će se učitati broj elemenata, polje se dinamički alocirati te napuniti slučajno odabranim vrijednostima iz raspona [10, 1000], pozvati funkcija i ispisati novo polje te osloboditi dinamički stvorena memorija.

Zadatak 4. *(2 boda)*

Odredite apriornu složenost sljedećeg programskog odsječka i obrazložite odgovor:	Odredite asimptotsku složenost sljedećeg programskog odsječka i obrazložite odgovor:

`int i, j, n, s=0;`
`...`
`for (j=2; j<n; j++){`
`  for(i=1; i<n; i*=j){`
`  s+= i*j;`
`  }`
`}`
	`int i, j, n, s=0;`
	`...`
	`for (i=4; i<n; i++){`
	`  for (j=i-4; j<i; j++){`
	`  s+= a[j];`
	`  }`
	`}`

Rješenje: O(_____) Obrazloženje:	Rješenje: Obrazloženje:

Zadatak 5. *(1 bod)*

Što će ispisati funkcija `main`?

```
char *funkcija (char *niz, char znak){
  if (*niz==0) return 0;
  else {
  if (*niz==znak) strcpy (niz, niz+1);
  return funkcija(niz+1, znak);
  }
}
```

```
int main () {
  char polje[]="AB1 2B2";
  funkcija(polje, 'B');
  printf ("%s", polje);
  return 0;
}
```

Rješenje: _____

Zadatak 6. *(2 boda)*

Prikazati sadržaj stoga u trenutku neposredno prije izlaska iz funkcije **`zamijeni`** i naznačite veličine varijabli.

```
void zamijeni(int *a, int *b) {
      int pom;
      pom = *a;
      *a = *b;
      *b = pom;
}
void sortiraj(int *polje, int n) {
      int i,j;
      for (i = 0; i < n-1; i++)
            for (j = i; j < n; j++)
                  if (polje[j] > polje[i])
                        zamijeni(polje+j, polje+i);
}

int main() {
      int polje[100];
      ...
      sortiraj(polje, 100);
}
```

Rješenja

1.

```
int urediHash(FILE *f)
{
	zapis pretinac[C];
	int pp, brojPrebac, j, i, imaPromjena=0;
	i=0;
	for(i=0;i<M;i++)
	{
		imaPromjena=0;
		fseek(f,i*BLOK,SEEK_SET);
		fread(pretinac,sizeof(pretinac),1,f);
		for(j=0;j<C;j++)
		{
			upisao=0;
			pp = adresa(pretinac[j].sifra);
			if(pretinac[j].sifra!=0 && pp!=i)
			{
				upisao=upisiZapis(f, pretinac[j], pp);
				if (upisao){
					pretinac[j].sifra = 0;
					brojPrebac++;
					imaPromjena=1;
				}

			}
		}
		if (imaPromjena){
			fseek(f,i*BLOK,SEEK_SET);
			fwrite(pretinac,sizeof(pretinac),1,f);
		}
}
	}
	return brojPrebac;
}
```

2.

```
short jeProst(int broj, int i)
{
	if (broj==i) return 1;
	if (broj%i == 0) return 0;
	return jeProst (broj, i+1);
}
```

3.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int *izbaci_podprosj(int *polje, int *br_clanova){
	int i, j, suma=0;
	float prosjek;
	for (i=0;i<*br_clanova;i++){
		suma+=polje[i];
	}
	prosjek= (float) suma/ *br_clanova;
	for(i=0 ; i<*br_clanova ;){
		if (polje[i] < prosjek){
			for(j=i+1 ; j<*br_clanova ; j++)
```

```
                        polje[j-1] = polje[j];
                (*br_clanova)--;
        }
        else
                i++;
    }
    polje = (int*) realloc(polje, *br_clanova * sizeof(int));
    return polje;
}

int main(){
    int *polje, i, br_clanova;
    scanf("%d",&br_clanova);
    polje = (int*) malloc(br_clanova * sizeof(int));
    srand((unsigned) time(NULL));
    for(i=0; i<br_clanova ; i++){
        polje[i] = rand()%991+10;
    }
    printf("prije:\n");
    for(i=0; i<br_clanova ; i++){
        printf("%d\n", polje[i]);
    }
    polje = izbaci_podprosj(polje, &br_clanova);
    printf("poslije:\n");
    for(i=0; i<br_clanova ; i++){
        printf("%d\n", polje[i]);
    }
    free(polje);
    return 0;
}
```

4. a) **O(n logn),** uz objašnjenje: vanjska petlja vrti se n puta, unutarnja: i se množi s j iz vanjske, a kako j ide do n, to je složenost unutarnje log n;
*(ili **O(n)** uz objašnjenje : u unutarnjoj petlji imamo množenje sa i; ako je i veći od sqrt(n) onda imamo najviše dva prolaza, što je O(1) - takvih za koje to vrijedi ima n -sqrt(n); za prvih sqrt(n) možemo reći da je složenost O(logn); znači ukupno imamo O((n-sqrt(n))* 1) + O(logn * sqrt(n)); prvi član je O(n) a drugi je očito <= O(n) => složenost je O(n))*
b) **~4n** uz objašnjenje: unutarnja petlja vrti se 4 puta, a vanjska n puta.

Napomena: bez valjanog objašnjenja rješenje se ne priznaje!

5. A1 22

6.

```
pom
pod.adr. zamijeni
polje+j       (ili a)
polje+i       (ili b)
j
i
pov.adr. sortiraj
polje
100           (ili n)
```