JMBAG **IME I PREZIME**

Algoritmi i strukture podataka – Završni ispit

17. lipnja 2014.

Nije dopušteno korištenje globalnih i statičkih varijabli te naredbe ***goto****. Neefikasna rješenja mogu nositi manje bodova. Ispit nosi maksimalno 30 bodova, a prag za prolaz završnog ispita je* ***10*** *bodova.*

3. zadatak rješavate na ovom obrascu, a ostale zadatke rješavate na svojim listovima papira. Ovaj obrazac, naravno, morate predati.

Zadatak 1. (6 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 čiji je prototip:

```
void podijeliListu(atom **glava, int ocjena);
```

koja će listu podijeliti s obzirom na zadanu ocjenu i pri tome obrnuti redoslijed elemenata.
Znači, nakon završetka funkcije svi čvorovi sa zapisima o studentima čija je ocjena manja ili jednaka zadanoj ocjeni trebaju biti u prvom dijelu liste u obrnutom redoslijedu od redoslijeda u zadanoj listi, dok ostali čvorovi trebaju biti u drugom dijelu liste, također u obrnutom redoslijedu.

Primjer: prije poziva funkcije lista sadrži sljedeće ocjene: 5, 2, 4, 3, 4, 3, 2, 4, 5, 1.

Nakon poziva funkcije `podijeliListu` sa zadanom ocjenom 3, lista treba biti: 1, 2, 3, 3, 2, 5, 4, 4, 4, 5.

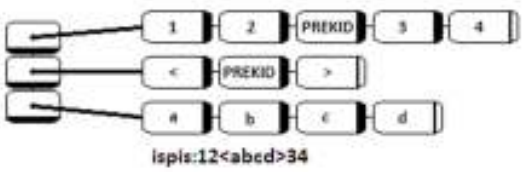
Zadatak 2. (6 bodova)

Definiran je tip podataka (varijabli) `Red` koji odgovara strukturi red čiji elementi su riječi (nizovi znakova). Za dohvat sljedećeg elementa reda koristi se funkcija:

```
char *redSkini(Red *red);
```

Također je definirana i struktura `Stog` koja se koristi za LIFO pohranu spomenutih redova, a za rad sa stogom na raspolaganju su funkcije:

```
void stogInit(Stog *stog);
void stogDodaj(Stog *stog, Red *element);
Red *stogSkini(Stog *stog);
```



Funkcija za dohvaćanje (skidanje) sa stoga vraća NULL ako je stog prazan.

Važno! Redovi riječi imaju strogo propisanu strukturu kao na slici, dakle svaki red osim posljednjeg (prvog stavljenog na stog) sadrži točno jednu riječ „`PREKID`“, dok posljednji red ne sadrži tu riječ. Također, može se pretpostaviti da stog nije prazan, tj. barem je jedan red riječi na stogu.

Potrebno je napisati funkciju `ispis` prototipa

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

koja ispisuje sve riječi svih redova na stogu na standardni izlaz tako da se trenutačni red ispisuje riječ po riječ sve do riječi „`PREKID`“. Riječ „`PREKID`“ se ne ispisuje, nego se nastavlja ispisivanjem sadržaja sljedećeg reda na stogu. Nakon što je sljedeći red na stogu ispisan, funkcija nastavlja ispisivati ostatak sadržaja prethodnog reda (čiji je ispis bio prekinut). Za stog i redove u primjeru na slici, ispis treba biti `12<abcd>34`.

Nakon povratka iz funkcije `ispis`, stog mora ostati sačuvan (iako redovi čije su adrese na stogu ne moraju).

Zadatak 3. (6 bodova)

Binarno stablo jednoznačno je zadano svojim *preorder* i *inorder* ispisima. U nacrtanom okviru skicirajte stablo čiji su *preorder* i *inorder* ispisi (redom s lijeva na desno)

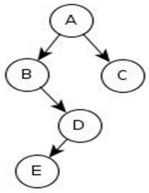
Preorder: `97, 3, 51, 37, 44, 33, 22, 61, 83`

Inorder: `51, 44, 37, 3, 97, 22, 33, 83, 61`

Zadatak 4. (6 bodova)

Čvor stabla definiran je odsječkom:

```
struct cv {
  int vrijednost;
  struct cv *lijevo, *desno;
};
```



Zadani su prototipovi dviju funkcija:

```
int sirinaRazine(cvor* korijen, int razina);
int sirinaStabla(cvor* korijen);
```

Funkcija `sirinaRazine` treba vratiti broj čvorova na razini `razina`. Funkcija `sirinaStabla` treba vratiti najveću širinu svih razina u binarnom stablu, dakle broj čvorova na najširoj razini. Po dogovoru, korijen stabla je na razini 1.

U primjeru na slici širina razine dva je 2, što je najveća širina u ovom stablu pa time i vrijednost koju funkcija `sirinaStabla` mora vratiti.

Napišite funkcije `sirinaRazine` i `sirinaStabla`.

Zadatak 5. (6 bodova)

Zadan je niz brojeva: **36, 58, 44, 28, 96, 62, 76, 38**.

a) Ilustrirajte (nacrtajte stablo nakon svake promjene) stvaranje gomile s relacijom ***veći od*** (max heap) od zadanog polja brojeva algoritmom čija je složenost za najgori slučaj *O(n).*

b) Počevši od gomile iz podzadatka a), prikažite sortiranje zadanog niza *heapsortom*, jasno prikazujući svaki korak sortiranja (nacrtajte stablo nakon svakog koraka algoritma).

Rješenja:

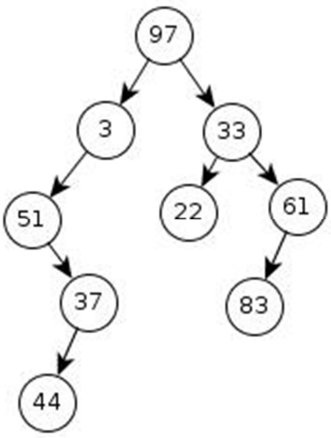
Zadatak 1. (6 bodova)

```
void podijeliListu(atom **glava, int element) {

	atom *lijevo = NULL, *desno = NULL, *sljedeci, *pom = *glava;

	while (pom != NULL) {
		sljedeci = pom->sljed;
		if (pom->element.ocjena <= element) {
			pom->sljed = lijevo;
			lijevo = pom;
		} else {
			pom->sljed = desno;
			desno = pom;
		}
		pom = sljedeci;
	}
	//spoji lijevu i desnu
	if (lijevo == NULL) {
		*glava = desno;
	} else {
		*glava = lijevo;
		while (lijevo->sljed != NULL)
			lijevo = lijevo->sljed;
		lijevo->sljed = desno;
	}
}
```

Zadatak 2. (6 bodova)

```
void ispisi( Stog *stog ) {
	Red *pomRed;
	char *rijec;
	Stog ostaci;
	stogInit( &ostaci );

	pomRed = stogSkini( stog );
	while( pomRed != NULL ) {
		rijec = redSkini( pomRed );
		if( rijec == NULL ) {
			stogDodaj( stog, pomRed );
			pomRed = stogSkini( &ostaci );
		}
		else if( strcmp( rijec, "PREKID" ) == 0 ) {
			stogDodaj( &ostaci, pomRed );
			pomRed = stogSkini( stog );
		}
		else {
			printf( "%s", rijec );
		}
	}
}
```

Zadatak 3. (6 bodova)



Način formiranja stabla iz danih ispisa:

1. prvi navedeni u preorder-u je glava
2. gledajući u inorder ispisu, očito će sve lijevo od glave biti lijevo podstablo, a sve desno od glave desno podstablo
3. postupak ponavljam rekurzivno za svaki dobiveni podskup

Zadatak 4. (6 bodova)

```
int sirinaRazine(cvor* korijen, int razina) {

	if(korijen==NULL || razina<1)  return 0;
	if(razina==1) return 1;

	return sirinaRazine(
			korijen->lijevo, razina-1) +
		sirinaRazine(
			korijen->desno,razina-1);

}

int sirinaStabla(cvor* korijen) {
	int max=0;
	int razina=1;
	int temp=0;
	do {
		temp=sirinaRazine(korijen, razina);
		++razina;
		max=(max>temp)?max:temp;
	} while (temp>0)
	return max;
}
```

Zadatak 5. (6 bodova)

a)
```
      36
    58 44
  28 96 62 76
38
```

```
      36
    58 44
  38 96 62 76
28
```

```
      36
    58 76
  38 96 62 44
```

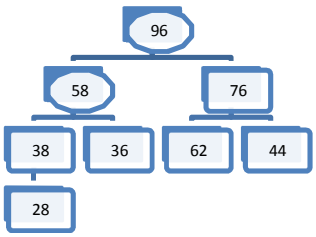
28

36
96 76
38 58 62 44
28

96
36 76
38 58 62 44
28

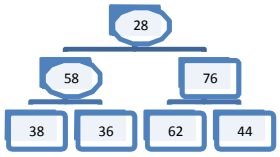
96
58 76
38 36 62 44
28

b)



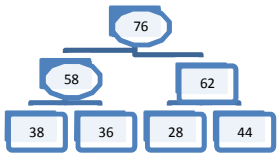
96	58	76	38	36	62	44	28

Zamjena



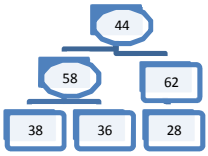
28	58	76	38	36	62	44	96

Podešavanje:



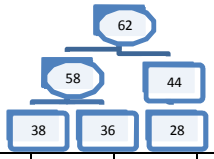
76	58	62	38	36	28	44	96

Zamjena



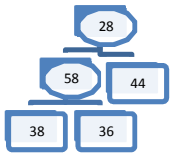
44	58	62	38	36	28	76	96

Podešavanje:



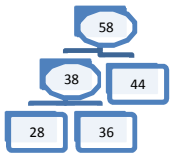
62	58	44	38	36	28	76	96

Zamjena:



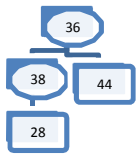
28	58	44	38	36	62	76	96

Podešavanje:



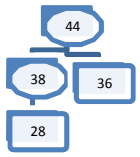
58	38	44	28	36	62	76	96

Zamjena



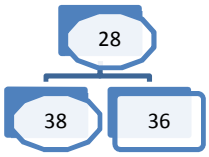
36	38	44	28	58	62	76	96

Podešavanje



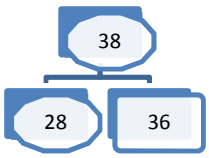
44	38	36	28	58	62	76	96

Zamjena



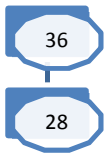
28	38	36	44	58	62	76	96

Podešavanje



38	28	36	44	58	62	76	96

Zamjena



36	28	38	44	58	62	76	96

Podešavanje – ok
Zamjena

28	36	38	44	58	62	76	96