Algoritmi i strukture podataka
Ponovljeni završni ispit
28.6.2007.

Napomene za sve zadatke:

- Nije dopušteno korištenje goto naredbe, te statičkih i globalnih varijabli.
- U svakom zadatku u kojem se koristi struktura, strukturu treba i definirati

1. (8 bodova) Ukoliko je red zadan sljedećom strukturom:

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

te, ukoliko su definirane funkcije za dodavanje elementa u red i skidanje elementa iz reda

```
int DodajURed (int element, atom **ulaz, atom **izlaz);
int SkiniIzReda (int *element, atom **ulaz, atom **izlaz);
```

napisati funkciju koja će sve elemente reda kopirati u binarno sortirano stablo. Funkcija mora vratiti pokazivač na korijen stabla. Nakon završetka funkcije, sadržaj reda mora ostati očuvan. Smijete koristiti pomoćne funkcije.

```
cvor *dodaj_u_stablo(cvor *korijen, int broj){
      if (korijen){
            if (broj < korijen->element)
                  korijen->lijevo =
                        dodaj_u_stablo(korijen->lijevo , broj);
            else
                  korijen->desno =
                        dodaj_u_stablo(korijen->desno , broj);
            return korijen;
      }
      else{
            cvor *novi = (cvor *) malloc(sizeof(cvor));
            novi->element = broj;
            novi->lijevo = novi->desno = NULL;
            return novi;
      }
}

cvor *red_u_stablo(atom **ulaz, atom **izlaz){
      cvor *korijen = NULL;
      atom *pom_ulaz = NULL, *pom_izlaz = NULL;
      int broj;
      while(SkiniIzReda(&broj, ulaz, izlaz)){
            DodajURed(broj, &pom_ulaz, &pom_izlaz);
            korijen = dodaj_u_stablo(korijen, broj);
      }
      while(SkiniIzReda(&broj, &pom_ulaz, &pom_izlaz)){
            DodajURed(broj, ulaz, izlaz);
```

```
	}
	return korijen;
}
```

2. (7 bodova) Definirati dvostruko povezanu listu koja sadrži točke iz koordinatnog sustava te napisati funkciju za brisanje svih točaka iz nekog kvadranta. Kvadrant (1,2,3 ili 4) je zadan kao argument funkcije. Napisati odsječak glavnog programa u kojem se poziva navedena funkcija i definiraju (i postavljaju na inicijalne vrijednosti) sve varijable koje se koriste pri pozivima navedenih funkcija. Smijete koristiti pomoćne funkcije. *Napomena: Pretpostavite da točke na koordinatnim osima ne pripadaju niti jednom kvadrantu.*

```
struct at {
	int element;
	struct at *preth;
	struct at *sljed;
};
typedef struct at atom;

void izbaci(atom **glava, int kvadrant){
	atom *trenutni;
	/* prvo izbaci sve s glave koji su u traženom kvadrantu*/
	while(*glava){
		if ( ((*glava)->x > 0 && (*glava)->y > 0 && kvadrant==1) ||
		   ( (*glava)->x < 0 && (*glava)->y > 0 && kvadrant==2) ||
		   ( (*glava)->x < 0 && (*glava)->y < 0 && kvadrant==3) ||
		    ( (*glava)->x > 0 && (*glava)->y < 0 && kvadrant==4) ){

				atom *temp = *glava;
				if ((*glava)->sljed)
					(*glava)->sljed->preth = NULL;
				*glava = (*glava)->sljed;
				free(temp);
		}
	}
	if (*glava == NULL) return;
	/* izbaciti one iz sredine */
	trenutni = (*glava)->sljed;
	while(trenutni){
		if (( trenutni->x > 0 && trenutni->y > 0 && kvadrant==1) ||
		 ( trenutni->x < 0 && trenutni->y > 0 && kvadrant==2) ||
		 ( trenutni->x < 0 && trenutni->y < 0 && kvadrant==3) ||
		 ( trenutni->x > 0 && trenutni->y < 0 && kvadrant==4) ){

				atom *temp = trenutni;
				trenutni->preth->sljed = trenutni->sljed;
				if (trenutni->sljed){
					trenutni->sljed->preth =
						trenutni->preth;

				}
				trenutni = trenutni->sljed;
				free(temp);
```

```
                        }
        }
}
```

Odsječak glavnog programa:

```
        atom *glava = NULL;int kvadrant;
        izbaci(&glava, kvadrant);
```

3. (8 bodova) Napisati funkciju čiji je prototip

int prepisi(cvor *korijen, atom **glava)

koja će sve elemente stabla (cijeli brojevi) kopirati u vezanu listu tako da vezana lista bude sortirana silazno. Definirati strukture cvor i atom. Smijete koristiti pomoćne funkcije.

```
void prepisi(cvor *korijen, atom **glava){
        atom *novi;
        if (korijen){
                prepisi(korijen->lijevo, glava);
                novi = (atom*) malloc(sizeof(atom));
                novi->element = korijen->element;
                novi->sljed = *glava;
                *glava = novi;
                prepisi(korijen->desno, glava);
        }
}
```

4. (7 bodova) Za niz brojeva 3,8,9,1,4,2,6,5,7 ilustrirati nastajanje gomile algoritmom

a) složenosti O(n)

b) složenosti O(nlogn)

Napomena: U rješenjima pažljivo označiti koje rješenje odgovara kojem algoritmu. U slučaju zamjene algoritama maksimalan broj bodova u zadatku je 3.5.