1. (7 bodova) Ukoliko je red, koji sadrži cijele brojeve, realiziran listom, napisati funkcije za dodavanje novog elementa u red i skidanje elementa iz reda. Također, deklarirati sve potrebne strukture za navedenu implementaciju i napisati odsječak glavnog programa koji demonstrira korištenje navedenih funkcija.

```
Dodavanje 3 boda
Skidanje 3 boda
glavni prog 1 bod
```

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

int DodajURed (int element, atom **ulaz, atom **izlaz) {
  atom *novi;
  if (novi = malloc (sizeof (atom))) {
    novi->element = element;
    novi->sljed = NULL;
    if (*izlaz == NULL) *izlaz = novi;
      else (*ulaz)->sljed = novi;
    *ulaz = novi;
    return 1;
  }
  return 0;
}

int SkiniIzReda (int *element, atom **ulaz, atom **izlaz) {
  atom *stari;
  if (*izlaz) {
    *element = (*izlaz)->element;
    stari = *izlaz;
    *izlaz = (*izlaz)->sljed;
    free (stari);
    if (*izlaz == NULL) *ulaz = NULL;
   return 1;
  }
  return 0;
}
```

Glavni program:
Definicija reda i jednog elementa:

```
      atom *ulaz = NULL; atom *izlaz = NULL; int broj;
```

Dodavanje broja u red: `DodajURed (broj, &ulaz, &izlaz);`
Skidanje broja u red: `SkiniIzReda(broj, &ulaz, &izlaz);`

2. (8 bodova) Definirati jednostruko povezanu linearnu listu koja sadrži razlomke (brojnik i nazivnik kao cijele brojeve). Ukoliko je lista uzlazno sortirana po vrijednostima razlomaka, napisati:

a) funkciju za dodavanje novog razlomka u vezanu listu. Ukoliko razlomak (identični i brojnik i nazivnik) već postoji u listi, funkcija ne radi ništa.
b) funkciju za izbacivanje svih razlomaka iz liste koji su veći od `p`, a manji od `q`, gdje su `p` i `q` realni brojevi, argumenti funkcije.

Napisati odsječak glavnog programa koji demonstrira korištenje navedenih funkcija Također, deklarirati sve varijable koje se koristi pri pozivima navedenih funkcija.

```
struktura 0.5
odsječak glavnog programa 1
dodavanje 3
izbacivanje 3.5
```

```c
typedef struct{
	int br;
	int naz;
} razlomak;

typedef struct _atom{
	razlomak r;
	struct _atom *sljed;
} atom;

/*vraca istinu ako je razlomak r1 manji od r2*/
int manji(razlomak r1, razlomak r2){
	return r1.br * r2.naz < r2.br * r1.naz;
}

/* prema tekstu zadataka, provjerava se samo
da li su međusobno jednaki brojnici, odnosno nazivnivi */
int jednaki(razlomak r1, razlomak r2){
	return r1.br == r2.br && r1.naz == r2.naz;
}

void dodaj(atom **glava, razlomak raz){
	atom *novi;
	novi = (atom*) malloc(sizeof(atom));
	novi->r = raz;
	if (*glava == NULL ||  manji(raz, (*glava)->r)){
		novi->sljed = *glava;
		*glava = novi;
	}
	else{
		atom *temp;
		temp = *glava;
		while(temp->sljed && manji(temp->sljed->r, raz))
			temp = temp->sljed;
		if (temp->sljed && jednaki(temp->sljed->r, raz))
			return;
		novi->sljed = temp->sljed;
		temp->sljed = novi;
	}
```

```
}
void izbaci(atom **glava, double p, double q){
	atom *temp, *preth;
	//briši glavu dok je ona između p i q
	while(*glava && (float) (*glava)->r.br / (*glava)->r.naz > p &&
		(float) (*glava)->r.br / (*glava)->r.naz < q){
			temp = *glava;
			*glava = (*glava)->sljed;
			free(temp);
	}
	if (*glava){
		preth = *glava;
		//dođi do prvog koji je veći od p
		while(preth->sljed &&
		  (float) preth->sljed->r.br / preth->sljed->r.naz <= p)

			preth = preth->sljed;

		//briši dok ima čvorova i dok su oni manji od q
		while(preth->sljed &&
		 (float) preth->sljed->r.br / preth->sljed->r.naz < q){
			temp = preth->sljed;
			preth->sljed = temp->sljed;
			free(temp);
		}
	}
}
```

Glavni program:
Definicija liste i razlomka

```
	atom *glava = NULL; razlomak r; double p, q;
```

Dodavanje razlomka u listu:

```
	dodaj (&glava, r);
```

Brisanje čvorova iz liste:

```
	izbaci(&glava, p, q);
```

Napomena: Moglo se riješiti i sa funkcijama čiji su prototipovi

```
	atom *dodaj(atom *glava, razlomak raz)
	atom *izbaci(atom *glava, double p, double q)
```

U tom slučaju je glavni program nešto drugačiji. Nije nužno definirati tip razlomak, već se mogu prenositi dva cijela broja kao brojnik i nazivnik.

3. (8 bodova) Binarno stablo sadrži točke iz koordinatnog sustava te je čvor definiran na sljedeći način:

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

Napisati funkciju čiji je prototip:

```
int prebroji(cvor *korijen, int n)
```

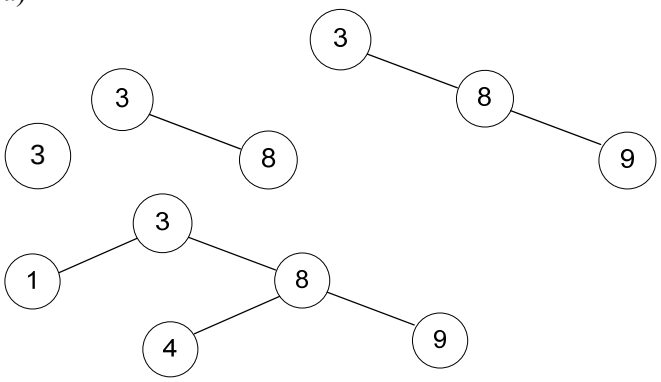
koja će vratiti koliko se magičnih čvorova nalazi u binarnom stablu na razini n. Čvor je magičan ukoliko je suma njegovih koordinata magičan broj. Funkcija koja određuje da li je broj magičan ili ne ima sljedeći prototip: `int magican(int broj);`
Napomena: Smijete koristiti pomoćne funkcije

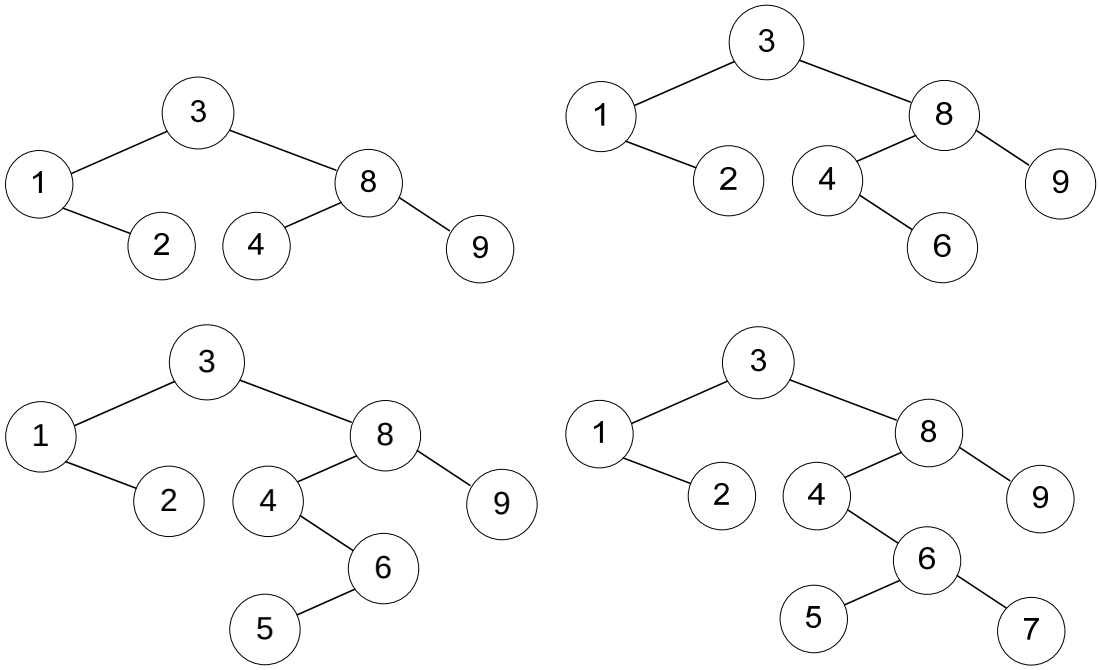
```
int prebroji(cvor *korijen, int n){
      if (korijen){
            n--;
            if (n>0)
            return prebroji(korijen->lijevo, n) +
                   prebroji(korijen->desno, n);
            else if (n==0)
                  return magican(korijen->x + korijen->y);
      }
      return 0;
}.
```

4. (7 bodova) Za ulazni niz brojeva 3,8,9,1,4,2,6,5,7 nacrtati
a) binarno sortirano stablo (stablo za traženje, uređeno stablo)
b) gomilu
Napomena: Prilikom crtanja nacrtati stanje stabla nakon svakog dodavanja broja.

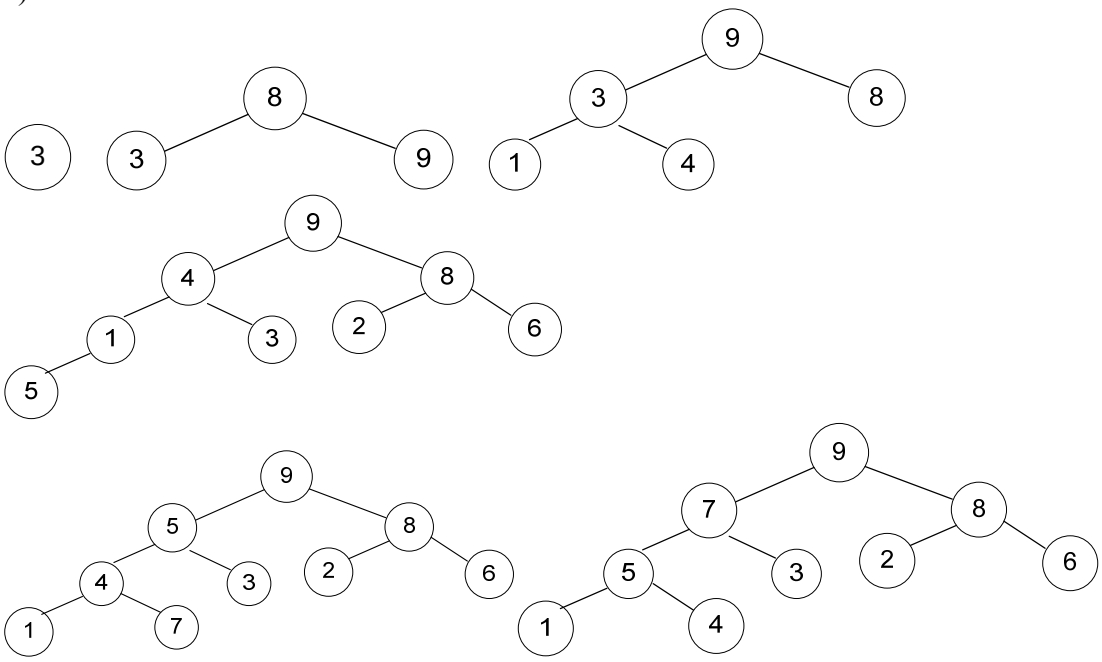
a) 3,5
b) 3,5

a)




3
1
8
2
4
9
3
1
8
2
4
9
6
3
1
8
2
4
9
6
5
3
1
8
2
4
9
6
5
7

b)


3
3
8
9
9
3
8
1
4
9
4
8
1
3
2
6
5
9
5
8
4
3
2
6
1
7
9
7
8
5
3
2
6
1
4