Sadržaj

Glava 1

Testovi i ispiti 2005/06

1.1 Test 1, 03.04.2006.

1. Korišćenjem nizova izračunati približnu vrednost realnog broja $1/x$ (za $0 < x \leq 2$) sa zadatom tačnošću ϵ ($0 \leq \epsilon \leq 0.01$) na sledeći način:

- odrediti elemente nizova (a) i (c) koju su zadati sa:

  $a[0] = 1$

  $c[0] = 1 - x$

  $a[i] = a[i-1] \cdot (1 + c[i-1]), \quad \text{za} \quad i \geq 1$

  $c[i] = c[i-1] \cdot c[i-1], \quad \text{za} \quad i \geq 1$

- za vrednost broja $1/x$ uzima se ona vrednost $a[n]$ za koju je zadovoljen uslov $-\epsilon \leq a[n] - a[n-1] \leq \epsilon$.

Prostor za nizove (a) i (c) dinamički alocirati i povećavati im dužinu (u većim blokovima) dok ne bude zadovoljena tražena tačnost.
Na primer:
za $x = 0.5$ i $\epsilon = 0.0001$ program treba da vrati 2.00000
za $x = 0.5$ i $\epsilon = 0.01$ program treba da vrati 1.999969

2. Napisati funkciju koja kao (jedini) arugment ima ime datoteke koja sadrži dimenziju kvadratne matrice i njene elemente redom (po vrstama). Funkcija treba da ispiše elemente matrice u grupama koje su paralelne sa sporednom dijagonalom matrice.
Na primer, datoteka sa sadržajem `3 1 2 3 4 5 6 7 8 9` opisuje matricu
1 2 3
4 5 6
7 8 9
a funkcija treba da ispiše sledeće:
1
2 4
3 5 7
6 8
9
Može se pretpostaviti da matrica nije dimenzije veće od 100×100.

3. Napraviti program za ažuriranje reda čekanja u studentskoj poliklinici. Treba obezbediti:

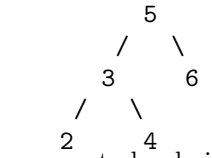
- funkciju koja sa standardnog ulaza čita broj indeksa studenta koji je došao (samo jedan ceo broj);

- funkciju koja ubacuje studenta (sa datim brojem indeksa) na kraj liste čekanja u vremenu $O(1)$;
- funkciju koja određuje sledećeg studenta za pregled i briše ga sa liste u vremenu $O(1)$;
- funkciju koja štampa trenutno stanje reda čekanja.
- funkciju koja oslobadja celu listu.

1.2 Test 2, 01.06.2006.

1. Napisati program koji za uredjeno binarno stablo ispisuje elemente na najvećoj dubini. Napisati funkcije za kreiranje čvora, unošenje elementa u stablo, oslobadjanje stabla.

Na primer, za stablo



program treba da ispiše: `2 4`.

2. Sa ulaza čitamo niz reči (dozvoljeno je ponavljanje reči) i smeštamo ih u listu (koja osim reči čuva i broj pojavljivanja za svaku reč). Napisati funkciju koja sortira listu algoritmom *bubble sort* po broju pojavljivanja reči. Napisati funkciju koja ispisuje listu počevši od reči koja se pojavila najviše puta.

3. Sa standardnog ulaza učitavamo prvo broj studenata a zatim i njihove podatke. Za svakog studenta dobijamo ime (niska od najviše 30 karaktera) i broj indeksa (ceo broj). Napisati program koji sortira ovaj niz studenata po imenima studenata pozivom standardne funkcije *qsort* a zatim pronalazi broj indeksa studenta čije se ime zadaje sa standardnog ulaza pozivom funkcije *bsearch*.

4. Napisati program koji simulira rad sa velikim brojevima. Cifre velikog broja smeštati u niz. Pretpostavljamo da broj neće imati više od 1000 cifara.

Napisati funkcije za:

- Unošenje broja sa standardnog ulaza (pri čemu cifre broja treba uneti na standardni način tj. počevši od cifre najveće težine) i smeštanje u niz koji se predaje kao argument funkcije. Dužinu niza (odnosno broja) vratiti kao povratnu vrednost funkcije.
- Ispis broja (odnosno niza) na standardni izlaz.
- Oduzimanje dva velika broja (niz za smeštanje rezultata proslediti funkciji kao argument). Pretpostavljamo da je prvi argument veći od drugog.

Pre poziva funkcije za oduzimanje brojeva obezbediti da cifra najmanje težine bude smeštena u $a[0]$ (ako je a niz koji predstavlja veliki broj).

Za unete brojeve `456 189` program treba da ispise `267`.

Za unete brojeve `456 89` program treba da ispise `367`.

1.3 Programiranje II, Završni ispit, jun 2006.

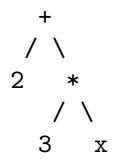
1. Napisati program za izračunavanje koeficijenata polinoma $T_n(x)$ za proizvoljno veliko n. Pri tome je $T_0(x) = a$, $T_1(x) = b \cdot x + c$ (celobrojne vrednosti n, a, b i c su argumenti komandne linije), a T_n se izračunava na osnovu formule $T_n = x \cdot T_{n-1} + T_{n-2}$. Polinom upisati u datoteku čije se ime zadaje kao peti argument komandne linije. Primer: nakon poziva

   ```
   program 2 1 2 3 izlaz.txt
   ```

   u datoteci `izlaz.txt` biće sadržaj:

   ```
   T2(x) = 2*x^2 + 3*x^1 + 1
   ```

2. Celobrojni aritmetički izraz koji uključuje jednu promenljivu predstavljen je binarnim stablom. Na primer, izraz $2 + (3 * x)$ predstavljen je stablom:



Izraz može da sadrži samo binarne operatore `+` i `*`.

- Definisati strukturu `cvor` kojom se mogu opisati čvorovi ovakvog stabla.
- Napisati funkciju

  `int vrednost(char ime_promenljive, int vrednost_promenljive, cvor* koren)`

  koja za promenljivu `ime_promenljive` sa vrednošću `vrednost_promenljive` računa vrednost izraza koji je predstavljen stablom čiji je koren `*koren` i vraća tu vrednost kao povratnu vrednost funkcije. Ukoliko u izrazu postoji promenljiva čija vrednost nije zadata, ispisuje se poruka o grešci i vraća vrednost 0.

  Na primer, ako je `izraz` pokazivač na koren stabla koje opisuje navedni izraz, onda se pozivom funkcije

  `vrednost('x', 1, izraz)`.

  dobija vrednost 5.

  Ako je `izraz` pokazivač na koren stabla koje opisuje navedni izraz, onda se pozivom funkcije

  `vrednost('y', 2, izraz)`.

  dobija poruka:

  `Promenljiva x nije definisana`

  i povratna vrednost 0.
- Napisati funkcije za ispis u prefiksnom i u infiksnom poretku drveta koje opisuje izraz.

Podrazumevati da su svi izrazi koji se dobijaju kao argumenti ispravno formirani.

3. U datotekama čija se imena zadaju sa standardnog ulaza nalaze se reči koje su leksikografski sortirane (unutar svake datoteke). Napisati program koji reči iz ovih datoteka smešta sortirano u treću datoteku (čije se ime takođe zadaje sa standardnog ulaza). Ne koristiti dinamički alociranu memoriju, niti funkcije za pozicioniranje unutar datoteka. Pretpostaviti da su sve reči dužine manje od 100 karaktera.

1.4 Programiranje II, Završni ispit, septembar 2006

1. Napisati program koji formira sortiranu listu od niza celih brojeva koji se unose sa standardnog ulaza. Oznaka za kraj unosa je 0. Napisati funkcije za formiranje čvora liste, ubacivanje elementa u već sortiranu listu, ispisivanje elemenata liste i oslobađanje liste.

2. Iz datoteke čije se ime zadaje kao argument komandne linije čita se prvo dimenzija kvadratne matrice n, pa zatim elementi matrice. Prostor za matricu alocirati dinamički. Napisati program koji:

   (a) pronalazi sve elemente matrice A koji su jednaki zbiru svih svojih susednih elemenata i štampa ih u obliku (broj vrste, broj kolone, vrednost elementa). Pod susednim elementima elementa $a[i][j]$ podrazumevamo elemente $a[i-1][j-1]$, $a[i-1][j]$, $a[i-1][j+1]$, $a[i][j-1]$, $a[i][j+1]$, $a[i+1][j-1]$, $a[i+1][j]$ i $a[i+1][j+1]$ (ako postoje).

(b) nalazi i štampa sve četvorke oblika
`( A(i,j), A(i+1,j), A(i,j+1),A(i+1,j+1) )` u kojima su svi elementi međusobno različiti.

3. Napisati program koji formira uređeno binarno stablo bez ponavljanja elemenata. Elementi stabla su celi brojevi i unose se sa ulaza, a oznaka za kraj unosa je 0. Napisati funkciju koja proverava da li je uneto stablo uravnoteženo. Stablo je uravnoteženo ako za svaki čvor stabla važi da mu se visina levog i desnog podstabla razlikuju najviše za jedan.

1.5 Programiranje II, Završni ispit, oktobar 2006

1. Napisati funkciju koja za dva niza karaktera proverava da li je:

   (a) prvi podniz drugog (elementi prvog niza se ne moraju nalaziti na susednim pozicijama u drugom nizu).

   (b) prvi podstring drugog (karakteri prvog stringa se moraju nalaziti na susednim pozicijama u drugom stringu). Ako jeste funkcija treba da vrati poziciju prvog niza u drugom odnosno -1 ako nije.

2. Napisati program koji simulira rad sa stekom. Napraviti funkcije push (za ubacivanje elementa u stek), pop (za izbacivanje elementa iz steka) i funkciju peek (koja na standardni izlaz ispisuje vrednost elementa koji se nalazi na vrhu steka).

3. Napisati program koji formira binarno uredjeno stablo. Napisati funkcije za:

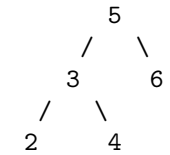
   (a) ubacivanje elementa u stablo, ispisivanje stabla, oslobadjanje stabla

   (b) sabiranje elemenata u listovima stabla, izračunavanje ukupnog broja čvorova stabla i izračunavanje dubine stabla.

1.6 Programiranje 2, Završni ispit, januarski rok, 2007

1. Napisati program koji za uredjeno binarno stablo ispisuje sve listove (list je čvor stabla koji nema potomaka). Napisati funkcije za kreiranje čvora, unošenje elementa u stablo, ispis stabla i oslobadjanje stabla.

   Na primer: za stablo



   program treba da ispiše: 2 4 6.

2. Napisati funkciju koja za dva niza karaktera proverava da li je prvi podstring drugog (elementi prvog stringa se moraju nalaziti na susednim pozicijama u drugom stringu).

3. Iz datoteke čije se ime zadaje kao argument komandne linije čita se prvo dimenzija kvadratne matrice n, pa zatim elementi matrice. Prostor za matricu alocirati dinamički. Napisati program koji računa sumu elemenata matrice koji se nalaze iznad glavne dijagonale.

   Na primer, za matricu:

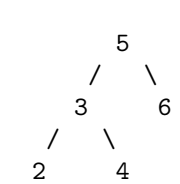
```
1 2 3
4 5 6
7 8 9
```

program izračunava sumu $2 + 3 + 6$ i ispisuje vrednost 12.

1.7 Programiranje 2, Završni ispit, februarski rok, 2007

1. Napisati program koji za uredjeno binarno stablo računa ukupan broj listova u stablu (list je čvor stabla koji nema potomaka). Napisati funkcije za kreiranje čvora, unošenje elementa u stablo, ispis stabla i oslobadjanje stabla.

   Na primer: za stablo



   program treba da ispiše: 3.

2. Napisati funkciju koja za dva niza karaktera proverava da li je jedan string permutacija drugog (jedan string je permutacija drugog stringa ako je od njega dobijen samo premeštanjem njegovih karaktera - bez ikakvog brisanja ili dodavanja karaktera).

   Npr. za stringove *abc* i *cba* funkcija treba da ispiše poruku da stringovi jesu permutacija. Za stringove *aab* i *ab* funkcija treba da ispiše poruku da stringovi nisu permutacija.

3. Iz datoteke čije se ime zadaje kao argument komandne linije čita se prvo dimenzija kvadratne matrice n, pa zatim elementi matrice. Prostor za matricu alocirati dinamički. Napisati program koji računa suma elemenata matrice koji se nalaze na sporednoj dijagonali.

   Na primer, za matricu:

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

   program izračunava sumu $3 + 5 + 7$ i ispisuje vrednost 15.

Glava 2

Testovi i ispiti 2006/07

2.1 Test 1, 17.04.2007. Grupa 1

1. Neka je, za zadati ceo broj n, n_1 proizvod cifara broja n, n_2 proizvod cifara broja n_1, ..., n_k proizvod cifara broja n_{k-1}, pri čemu je k najmanji prirodan broj za koji je n_k jednocifren. Na primer:

   - za $n = 10$, $n_1 = 1 * 0 = 0$, znači $k = 1$
   - za $n = 25$, $n_1 = 2 * 5 = 10$, $n_2 = 1 * 0 = 0$, znači $k = 2$
   - za $n = 39$, $n_1 = 3 * 9 = 27$, $n_2 = 2 * 7 = 14$, $n_3 = 1 * 4 = 4$, znači $k = 3$

   Napisati: (a) rekurzivnu; (b) iterativnu funkciju koja za dato n računa k. Zadatak rešiti bez korišćenja nizova.

2. Napisati program koji sa standardnog ulaza učitava pozitivne cele brojeve dok ne učita nulu kao oznaku za kraj. Na standardni izlaz ispisati koji broj se pojavio najviše puta među tim brojevima. Na primer, ako se na ulazu pojave brojevi: `2 5 12 4 5 2 3 12 15 5 6 6 5` program treba da vrati broj 5. Zadatak rešiti korišćenjem dinamičke realokacije.

3. Napisati program koji iz datoteke, čije se ime zadaje kao prvi argument komandne linije, čita prvo dimenziju kvadratne matrice n, a zatim elemente matrice (pretpostavljamo da se u datoteci nalaze brojevi pravilno raspoređeni, odnosno da za dato n, sledi $n \times n$ elemenata matrice). Matricu dinamički alocirati. Nakon toga, na standardni izlaz ispisati redni broj kolone koja ima najveći zbir elemenata. Na primer, za datoteka sa sadržajem:

   ```
   3
   1 2 3
   7 3 4
   5 3 1
   ```

   program treba da ispiše redni broj 0 (jer je suma elemenata u nultoj koloni `1 + 7 + 5 = 13`, u prvoj `2 + 3 + 3 = 8`, u drugoj `3 + 4 + 1 = 8`).

2.2 Test 1, 17.04.2007. Grupa 2

1. Neka je, za zadati ceo broj n, n_1 suma cifara broja n, n_2 suma cifara broja n_1, ..., n_k suma cifara broja n_{k-1}, pri čemu je k najmanji prirodan broj za koji je n_k jednocifren. Na primer:

   - za $n = 10$, $n_1 = 1 + 0 = 1$, znači $k = 1$
   - za $n = 39$, $n_1 = 3 + 9 = 12$, $n_2 = 1 + 2 = 3$, znači $k = 2$
   - za $n = 595$, $n_1 = 5 + 9 + 5 = 19$, $n_2 = 1 + 9 = 10$, $n_3 = 1 + 0 = 1$, znači $k = 3$

Napisati: (a) rekurzivnu; (b) iterativnu funkciju koja za dato n računa k. Zadatak rešiti bez korišćenja nizova.

2. Napisati program koji sa standardnog ulaza učitava pozitivne cele brojeve dok ne učita nulu kao oznaku za kraj. Na standardni izlaz ispisati koji broj se pojavio najviše puta među tim brojevima. Na primer, ako se na ulazu pojave brojevi: `2 5 12 4 5 2 3 12 15 5 6 6 5` program treba da vrati broj 5. Zadatak rešiti korišćenjem dinamičke realokacije.

3. Napisati program koji iz datoteke, čije se ime zadaje kao prvi argument komandne linije, čita prvo dimenziju kvadratne matrice n, a zatim elemente matrice (pretpostavljamo da se u datoteci nalaze brojevi pravilno raspoređeni, odnosno da za dato n, sledi $n \times n$ elemenata matrice). Matricu dinamički alocirati. Nakon toga, na standardni izlaz ispisati redni broj vrste koja ima najveći zbir elemenata. Na primer, za datoteka sa sadržajem:

   ```
   3
   1 2 3
   7 3 4
   5 3 1
   ```

   program treba da ispiše redni broj 1 (jer je suma elemenata u nultoj vrsti `1 + 2 + 3 = 6`, u prvoj `7 + 3 + 4 = 14`, u drugoj `5 + 3 + 1 = 9`).

2.3 Test 2, 19.06.2007., grupa 1.

1. Napisati program koji implementira red pomoću jednostruko povezane liste (čuvati pokazivače i na početak i na kraj reda). Sa standardnog ulaza se unose brojevi koje smeštamo u red sa nulom (0) kao oznakom za kraj unosa. Napisati funkcije za:

   - Kreiranje elementa reda;
   - Ubacivanje elementa na kraj reda;
   - Izbacivanje elementa sa početka reda;
   - Ispisivanje reda;
   - Oslobađanje reda.

2. Napisati program koji formira uređeno binarno stablo koje sadrži cele brojeve. Brojevi se unose sa standardnog ulaza sa nulom kao oznakom za kraj unosa. Napisati funkcije za:

   - Ubacivanje elementa u uređeno stablo (bez ponavljanja elemenata);
   - Ispisivanje stabla u inorder (infix) redosledu;
   - Oslobađanje stabla;
   - Određivanje najmanje dubine lista stabla.
     Na primer, za stablo:

     ```
        5
       / \
      3   7
     / \
    2   4
     ```

     funkcija treba da vrati 2 (jer se na toj dubini nalazi list 7).

3. Napisati program koji sa standardnog ulaza učitava podatke o studentima tako što za svakog studenta dobijamo prezime (karakterska niska od najviše 30 karaktera) i broj indeksa (ceo broj). Pretpostavka je da studenata nema više od 100.

   Sortirati ovaj niz studenata po prezimenima studenata pozivom standardne funkcije `qsort` i ispisati ih na standardni izlaz.

2.4 Test 2, 19.06.2007., grupa 2.

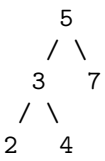
1. Napisati program koji implementira stek pomoću jednostruko povezane liste. Sa standardnog ulaza se unose brojevi koje smeštamo u stek sa nulom (0) kao oznakom za kraj unosa. Napisati funkcije za:

   - Kreiranje elementa steka;
   - Ubacivanje elementa na početak steka;
   - Izbacivanje elementa sa početka steka;
   - Ispisivanje steka;
   - Oslobađanje steka.

2. Napisati program koji formira uređeno binarno stablo koje sadrži cele brojeve. Brojevi se unose sa standardnog ulaza sa nulom kao oznakom za kraj unosa. Napisati funkcije za:

   - Ubacivanje elementa u uređeno stablo (bez ponavljanja elemenata);
   - Ispisivanje stabla u preorder (prefix) redosledu;
   - Oslobađanje stabla;
   - Određivanje najveće dubine lista stabla.

     Na primer, za stablo:

     funkcija treba da vrati 3 (jer se na toj dubini nalazi list 4).

3. Napisati program koji sa standardnog ulaza učitava podatke o studentima tako što za svakog studenta dobijamo prezime (karakterska niska od najviše 30 karaktera) i broj indeksa (ceo broj). Pretpostavka je da studenata nema više od 100.

   Sortirati ovaj niz studenata po brojevima indeksa studenata pozivom standardne funkcije `qsort` i ispisati ih na standardni izlaz.

2.5 Programiranje 2, Završni ispit, juni 2007.

1. Napisati program koji sa standardnog ulaza učitava dve karakterske niske dužine do 20 karaktera i proverava da li je prva niska podniz druge niske (odnosno da li se svi karakteri prve niske nalaze u drugoj nisci, ne obavezno u istom redosledu). Napisati funkciju koja vraća 1 ako je prva niska podniz druge, odnosno 0 u suprotnom.

2. Napisati program koji za datoteku čije se ime zadaje kao prvi argument komandne linije određuje i ispisuje reč koja se pojavljuje najviše puta u toj datoteci (pretpostavljamo da su reči dužine najviše 20 karaktera). Zadatak rešiti korišćenjem dinamičke realokacije.

3. Napisati program koji formira sortiranu listu od celih brojeva koji se unose sa standardnog ulaza. Oznaka za kraj unosa je 0.

   Napisati funkcije za:

   - Formiranje čvora liste,

- Ubacivanje elementa u već sortiranu listu,
- Ispisivanje elemenata liste u rastućem poretku u vremenu $O(n)$,
- Ispisivanje elemenata liste u opadajućem poretku u vremenu $O(n)$,
- Oslobađanje liste.

Napomena: potrebno je da lista bude takva da funkcije za ispis liste u rastućem i u opadajućem poretku *ne koriste rekurziju niti dodatnu alociranu memoriju* a rade u vremenu $O(n)$.

2.6 Programiranje 2, Završni ispit, septembar 2007.

1. Napisati program koji formira uređeno binarno stablo bez ponavljanja elemenata. Elementi stabla su celi brojevi i unose se sa standardnog ulaza (oznaka za kraj unosa je 0). Napisati funkcije za kreiranje elementa stabla, umetanje elementa u uređeno stablo, štampanje stabla, brisanje stabla i određivanje sume elemenata u listovima stabla.

2. Napisati program koji simulira rad sa stekom. Napisati funkcije *push* (za ubacivanje elementa u stek), *pop* (za izbacivanje elementa iz steka) i funkciju *peek* (koja na standardni izlaz ispisuje vrednost elementa koji se nalazi na vrhu steka - bez brisanja tog elementa iz steka).

3. Napisati program koji sa standardnog ulaza učitava tekst nepoznate dužine i smešta ga u (dinamički) niz karaktera. Oznaka za kraj unosa je uneta 0. Nakon toga ispisati uneti tekst na standardni izlaz po 10 karaktera u redu. Zadatak rešiti korišćenjem dinamičke realokacije.

   Na primer, ako je uneto:

   "Ja polazem ispit iz programiranja2"

   program treba da ispise:

   ```
   Ja polazem
   ispit iz p
   rogramiran
   ja2
   ```

4. Napisati program koji sa standardnog ulaza unosi prvo broj artikala a zatim i podatke o artiklima (ime artikla - karakterska niska dužine do 20 karaktera i cena artikla - ceo broj), sortira ih po ceni (pozivom funkcije qsort) i nakon toga (pozivom funkcije bsearch) određuje naziv artikla čiju cenu korisnik zadaje sa standardnog ulaza.

2.7 Programiranje 2, Završni ispit, oktobar 2007.

1. Napisati program koji formira uređeno binarno stablo bez ponavljanja elemenata čiji elementi su imena studenata i brojevi njihovih indeksa (struktura). Pretpostavka je da ime studenta nije duže od 30 karaktera i da je indeks dat kao ceo broj. Napisati program koji sa standardnog ulaza čita podatke o studentima, smešta ih u stablo (uređeno prema brojevima indeksa) i štampa podatke o studentima u opadajućem poredku prema brojevima indeksa. Oznaka za kraj unosa je kada se umesto imena studenta unese niska "*kraj*". Napisati funkcije za kreiranje čvora stabla, umetanje studenta u stablo, brisanje stabla i ispis stabla na opisan način.

2. Napisati program koji simulira red u studentskoj poliklinici. Napisati funkcije *add* (za ubacivanje studenta na kraj reda), *get* (za izbacivanje studenta sa početka reda) i funkciju *peek* (koja na standardni izlaz ispisuje ime studenta koji se nalazi na početku reda - bez brisanja tog studenta iz reda). Podaci o studentu se sastoje od imena studenta (karakterska niska dužine ne veće od 30) i broja indeksa studenta (ceo broj).

3. Napisati program koji sa standardnog ulaza učitava prvo dimenzije matrice (n i m) a zatim redom i elemente matrice (ne postoje pretpostavke o dimenziji matrice). Nakon toga u datoteku čije se ime zadaje kao prvi argument komandne linije, zapisati indekse (i i j) onih elemenata matrice koji su jednaki zbiru svih svojih susednih elemenata (pod susednim elementima podrazumevamo okolnih 8 polja matrice ako postoje). Na primer, za matricu:

```
1 1 2 1 3
0 8 1 9 0
1 1 1 0 0
0 3 0 2 2
```

polja na pozicijama [1][1], [1][3], [3][2] i [3][4] zadovoljavaju traženi uslov pa u datoteku treba upisati:

```
1 1
1 3
3 2
3 4
```

4. Napisati program koji sa standardnog ulaza unosi prvo broj studenata a zatim i podatke o studentima (ime studenata - karakterska niska dužine do 30 karaktera i broj indeksa studenta - ceo broj), sortira ih po imenu studenta leksikografski (pozivom funkcije qsort) i nakon toga (pozivom funkcije bsearch) određuje broj indeksa studenta čije ime korisnik zadaje sa standardnog ulaza.

Glava 3

Testovi i ispiti 2007/08

3.1 Test 1, 14.04.2008.

1. Napisati program koji iz datoteke čije se ime zadaje kao prvi argument komandne linije čita sve reči i zatim određuje i ispisuje reč koja se pojavila najviše puta u toj datoteci (pretpostavljamo da su sve reči u datoteci dužine najviše 20 karaktera). Koristiti pogodno definisanu strukturu.

2. Za zadati ceo broj n veći od 9 možemo primetiti da je suma cifara tog broja uvek manja od broja n. Neka je n_1 suma cifara broja n, neka je n_2 suma cifara broja n_1, n_3 suma cifara broja n_2, ..., n_{i+1} suma cifara broja n_i. Važi $n > n_1 > n_2 > \ldots > n_i$ dok god su brojevi n_i veći od 9, pa se u jednom trenutku sigurno dolazi do jednocifrenog broja n_k. Taj indeks k zavisi od početnog broja n.

   Na primer:

   - za $n = 10$, $n_1 = 1 + 0 = 1$, važi $k = 1$
   - za $n = 39$, $n_1 = 3 + 9 = 12$, $n_2 = 1 + 2 = 3$, važi $k = 2$
   - za $n = 595$, $n_1 = 5 + 9 + 5 = 19$, $n_2 = 1 + 9 = 10$, $n_3 = 1 + 0 = 1$, važi $k = 3$

   Napisati program koji za uneto n sa standardnog ulaza računa njemu odgovarajući broj k. Zadatak rešiti bez korišćenja nizova.

3. Za datu kvadratnu matricu kažemo da je *magični kvadrat* ako je suma elemenata u svakoj koloni i svakoj vrsti jednaka. Napisati program koji sa standardnog ulaza učitava prirodni broj n ($n < 10$) i zatim elemente kvadratne matrice, proverava da li je ona *magični kvadrat* i ispisuje odgovarajuću poruku na standardni izlaz.

   Primer, matrica:

   ```
   1 5 3 1
   2 1 2 5
   3 2 2 3
   4 2 3 1
   ```

   je magični kvadrat.

3.2 Test 2, 17.06.2008.

1. Napisati program koji implementira red pomoću jednostruko povezane liste (čuvati pokazivače i na početak i na kraj reda). Sa standardnog ulaza se unose brojevi koje smeštamo u red sa nulom (0) kao oznakom za kraj unosa. Napisati funkcije za:

- Kreiranje elementa reda;
- Ubacivanje elementa na kraj reda;
- Izbacivanje elementa sa početka reda;
- Ispisivanje reda;
- Oslobađanje reda.

2. Napisati program koji formira uređeno binarno stablo bez ponavljanja elemenata. Elementi stabla su celi brojevi i unose se sa ulaza, a oznaka za kraj unosa je nula. Napisati funkciju koja proverava da li je uneto stablo *lepo*. Stablo je *lepo* ako za svaki čvor stabla važi da mu se broj čvorova u levom i u desnom podstablu razlikuju najviše za jedan.

3. Iz datoteke *ulaz.txt* učitavamo niz celih brojeva (pri čemu dužina datoteke nije unapred poznata). Napisati program koji ovaj niz sortira pozivom funkcije *qsort* i zatim ga upisuje u datoteku *izlaz.txt*.

3.3 Završni ispit, juni 2008.

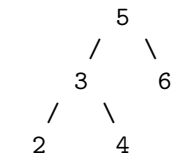
1. Napisati program koji sa standardnog ulaza učitava pozitivne cele brojeve dok ne učita nulu kao oznaku za kraj. Na standardni izlaz ispisati koji broj se pojavio najviše puta među tim brojevima. Na primer, ako se na ulazu pojave brojevi: `2 5 12 4 5 2 3 12 15 5 6 6 5` program treba da vrati broj 5.

   Zadatak rešiti korišćenjem dinamičke realokacije i strukture koja sadrži ceo broj i broj njegovih pojavljivanja.

2. Napisati program koji formira sortiranu listu od niza celih brojeva koji se unose sa standardnog ulaza. Oznaka za kraj unosa je 0. Napisati funkcije za formiranje čvora liste, ubacivanje elementa u već sortiranu listu, ispisivanje elemenata liste i oslobađanje liste.

3. Napisati funkcije potrebne za ispisivanje elemenata koji se nalaze na najvećoj dubini binarnog stabla.

   Na primer, za stablo

   treba ispisati: `2 4`.

   (Pretpostavljamo da je stablo već zadato. *Ne treba* pisati dodatne funkcije za kreiranje čvora, unošenje elementa u stablo i oslobađanje stabla)

4. Sa standardnog ulaza učitavamo prvo broj studenata a zatim i njihove podatke. Za svakog studenta dobijamo ime (niska od najviše 30 karaktera) i broj indeksa (ceo broj). Napisati program koji sortira ovaj niz studenata po imenima studenata pozivom standardne funkcije *qsort* i zatim štampa tako dobijeni niz na standardni izlaz.

3.4 Završni ispit, septembar 2008.

1. Napisati program koji sa standardnog ulaza učitava tekst nepoznate dužine i smešta ga u (dinamički) niz karaktera. Oznaka za kraj unosa je uneta 0. Nakon toga ispisati uneti tekst na standardni izlaz po 10 karaktera u redu. Zadatak rešiti korišćenjem dinamičke realokacije.

   Na primer, ako je uneto:

   `"Ja polazem ispit iz programiranja2"`

   program treba da ispiše:

```
Ja polazem
ispit iz p
rogramiran
ja2
```

2. Napisati program koji formira uređeno binarno stablo koje sadrži cele brojeve. Brojevi se unose sa standardnog ulaza sa nulom kao oznakom za kraj unosa. Napisati funkcije za:

   - Kreiranje jednog čvora stabla;
   - Ubacivanje elementa u uređeno stablo (bez ponavljanja elemenata);
   - Ispisivanje stabla u preorder (prefix) redosledu;
   - Određivanje zbira elemenata stabla;
   - Ispisivanje listova stabla;
   - Oslobađanje stabla.

   Primer, za stablo:

   ```
      5
     / \
    3   7
   / \
  2   4
   ```

   zbir svih elemenata je $5 + 3 + 7 + 2 + 4 = 21$ a listovi su: 2, 4 i 7.

3. Napisati program koji iz datoteke čije se ime zadaje kao argument komandne linije, čita prvo broj elemenata niza pa zatim i elemente niza (celi brojevi). Ovaj niz sortirati pozivom funkcije *qsort* a zatim za uneti ceo broj sa standradnog ulaza proveriti, pozivom funkcije *bsearch*, da li se nalazi u nizu ili ne i ispisati odgovarajuću poruku.

3.5 Završni ispit, oktobar 2008.

1. Napisati program koji sa standardnog ulaza unosi sortirani niz celih brojeva dok ne unesemo nulu kao oznaku za kraj (niz alocirati dinamički). U slučaju da niz nije sortiran ispisati podatak o grešci na standardni izlaz a u suprotnom ispisati *medijanu* tog niza. Medijana (sortiranog) niza koji ima n članova je, u slučaju kada je n neparan broj, središnji element niza odnosno, u slučaju kada je n paran broj, srednja vrednost dva središnja elementa.

   Na primer,

   ako je dat niz 1, 3, 5, 6, 12, medijana je broj 5

   a ako je dat niz 3, 5, 8, 13, 20, 25, medijana je broj 10.5.

2. Napisati program koji kreira jednostruko povezanu listu. Elementi (celi brojevi) se unose sa standardnog ulaza dok se ne unese nula kao oznaka za kraj. Napisati:

   - funkciju koja kreira jedan čvor liste, `CVOR* napravi_cvor(int br)`
   - funkciju za ubacivanje broja na kraj liste, `void ubaci_na_kraj(CVOR** pl, int br)`
   - funkciju koja skida element sa početka liste (pri čemu se menja početak liste) i vraća vrednost tog broja kao povratnu vrednost,
     `int izbaci_sa_pocetka(CVOR** pl)`

- funkciju za ispisivanje elemenata liste, `void ispisi_listu(CVOR* l)`
- funkciju za oslobađanje liste, `void oslobodi_listu(CVOR* l)`

3. Napisati program koji sa standardnog ulaza učitava broj artikala (ne više od 50) a zatim imena (karakterske niske dužine do 20 karaktera) i cene artikala (ceo broj). Ovaj niz artikala sortirati po ceni (pozivom funkcije *qsort*) a zatim za uneti ceo broj c sa standardnog ulaza pronaći (pozivom funkcije *bsearch*) naziv artikla sa tom cenom. Ako takav artikal ne postoji ispisati odgovarajuću poruku.

Glava 4

Testovi i ispiti 2008/09

4.1 Test 1, 01.04.2009.

1. Napisati program koji sa standardnog ulaza učitava pozitivne cele brojeve dok ne učita nulu kao oznaku za kraj. Na standardni izlaz ispisati koji broj se pojavio najviše puta među tim brojevima. Na primer, ako se na ulazu pojave brojevi: `2 5 12 4 5 2 3 12 15 5 6 6 5` program treba da vrati broj 5. Zadatak rešiti korišćenjem dinamičke realokacije i uz korišćenje struktura.

2. Napisati rekurzivnu i iterativnu funkciju koja za uneto n sa standardnog ulaza računa $f(n)$ ako je $f(n)$ definisan na sledeći način:

   $f(1) = 1$, $f(2) = 2$, $f(3) = 3$, $f(n+3) = f(n+2) + f(n+1) + f(n)$, za $n > 0$. Napisati i program koji poziva ove dve funkcije.

3. Napisati program koji sa standardnog ulaza unosi prvo dimenziju matrice ($n < 10$) pa zatim elemente matrice i izračunava sumu elemenata iznad sporedne dijagonale matrice. Napisati funkciju koja računa sumu elemenata iznad glavne dijagonale koja se izvršava u što manjem broju koraka.

   Primer:

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

   Suma elemenata iznad sporedne dijagonale je: $1 + 2 + 3 + 5 + 6 + 9 = 26$

4.2 Test 2, 06.06.2009.

1. Napisati funkcije za rad sa stekom čiji su elementi celi brojevi. Treba napisati (samo) funkcije za dodavanje elementa u stek `void push(cvor** s, int br)` , brisanje elementa iz steka `void pop(cvor** s)` i funkciju za izračunavanje zbira svih elemenata koji su parni brojevi `int zbir_parnih(cvor* s)`.

2. Napisati funkcije za rad sa uređenim binarnim stablom čiji su elementi celi brojevi. Treba napisati (samo) funkcije za dodavanje elementa u stablo `void dodaj(cvor** s, int br)` , brisanje stabla `void obrisi(cvor* s)` i funkciju za izračunavanje zbira listova stabla `int zbir_listova(cvor* s)`.

3. Napisati program koji radi sa tačkama. Tačka je predstavljena svojim x i y koordinatama (celi brojevi). Sa standardnog ulaza se učitava prvo broj tačaka a zatim koordinate tačaka. Dobijeni niz struktura sortirati pozivom funkcije *qsort*. Niz sortirati po x koordinati, a ako neke dve tačke imaju istu x koordinatu onda ih sortirati po y koordinati.

   Ako na ulazu dobijete niz: $(4, 6)$, $(2, 9)$, $(4, 5)$; sortirani niz će izgledati: $(2, 9)$, $(4, 5)$, $(4, 6)$.

4.3 Završni ispit, juni 2009.

Napisati program koji iz datoteke "reci.txt"čita redom sve reči (maksimalne dužine 20) i smesta ih u:

1. Niz struktura (pretpostaviti da različitih reči u datoteci nema više od 100) u kome će se za svaku reč čuvati i njen broj pojavljivanja.

2. Red koji će za svaku reč čuvati i njen broj pojavljivanja. Napisati funkcije za ubacivanje elementa u red, štampanje reda, brisanje reda i određivanja najduže reči koja se pojavila u redu.

3. Uređeno binarno drvo (uređeno leksikografski) koje u svakom čvoru čuva reč i broj pojavljivanja te reči. Napisati funkcije za dodavanje elementa u stablo, ispisivanje stabla, brisanje stabla i računanje ukupnog broja pojavljivanja svih reči u stablu.

4.4 Završni ispit, septembar 2009.

1. Napisati program koji iz datoteke čije se ime zadaje kao argument komandne linije učitava cele brojeve i smešta ih u listu. Napisati funkcije za rad sa listom, pravljenje elementa liste, ubacivanje elementa na kraj liste, ispisivanje liste i brisanje liste. Pretpostavka je da datoteka sadrži samo cele brojeve.

2. Napisati program koji sa standardnog ulaza učitava cele brojeve dok se ne unese 0 kao oznaka za kraj. Brojeve smestiti u uređeno binarno stablo i ispisati dobijeno stablo. Napisati funkcije za formiranje elementa stabla, ubacivanje elementa u stablo, ispisivanje stabla u prefiksnom poretku, brisanje stabla.

3. 3. Sa standardnog ulaza učitavamo prvo broj studenata a zatim i njihove podatke. Za svakog studenta dobijamo ime (niska od najviše 30 karaktera) i broj indeksa (ceo broj). Napisati program koji sortira ovaj niz studenata po imenima studenata pozivom standardne funkcije *qsort* a zatim pronalazi broj indeksa studenta čije se ime zadaje sa standardnog ulaza pozivom funkcije *bsearch*.

4.5 Završni ispit, oktobar 2009

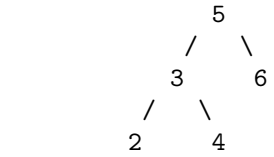
1. Napisati program koji sa standardnog ulaza učitava pozitivne cele brojeve dok ne učita nulu kao oznaku za kraj. Na standardni izlaz ispisati koji broj se pojavio najviše puta među tim brojevima. Na primer, ako se na ulazu pojave brojevi: `2 5 12 4 5 2 3 12 15 5 6 6 5` program treba da vrati broj 5.

   Zadatak rešiti korišćenjem dinamičke realokacije i strukture koja sadrži ceo broj i broj njegovih pojavljivanja.

2. Napisati program koji formira listu od niza celih brojeva koji se unose sa standardnog ulaza. Oznaka za kraj unosa je 0. Napisati funkcije za formiranje čvora liste, ubacivanje elementa na kraj liste, ispisivanje elemenata liste i oslobađanje liste i u programu demonstrirati pozive ovih funkcija.

3. Napisati funkcije potrebne za ispisivanje elemenata koji se nalaze na najvećoj dubini binarnog stabla.

   Na primer, za stablo

   treba ispisati: `2 4`.

   (Pretpostavljamo da je stablo već zadato. *Ne treba* pisati dodatne funkcije za kreiranje čvora, unošenje elementa u stablo i oslobađanje stabla)

4. Sa standardnog ulaza učitavamo prvo broj studenata a zatim i njihove podatke. Za svakog studenta dobijamo ime (niska od najviše 30 karaktera) i broj indeksa (ceo broj). Napisati program koji sortira ovaj niz studenata po imenima studenata pozivom standardne funkcije *qsort* i zatim štampa tako dobijeni niz na standardni izlaz.

4.6 Završni ispit, Oktobar 2 2009.

1. Napisati program koji formira uređeno binarno stablo bez ponavljanja elemenata čiji elementi su imena studenata (karakterska niska do 30 karaktera). Napisati program koji sa standardnog ulaza čita podatke o studentima, smešta ih u stablo (uređeno leksikografski) i štampa podatke o studentima infiksno. Oznaka za kraj unosa je kada se umesto imena studenta unese niska "*kraj*". Napisati funkcije za kreiranje čvora stabla, umetanje studenta u stablo, brisanje stabla i ispis stabla na opisan način.

2. Napisati program koji simulira red u studentskoj poliklinici. Napisati funkcije *add* (za ubacivanje studenta na kraj reda), *get* (za izbacivanje studenta sa početka reda) i funkcije za štampanje i brisanje reda. Podaci o studentu se sastoje od imena studenta (karakterska niska dužine ne veće od 30) i broja indeksa studenta (ceo broj).

3. Napisati program koji sa standardnog ulaza učitava prvo dimenzije matrice (n i m) a zatim redom i elemente matrice (ne postoje pretpostavke o dimenziji matrice). Nakon toga u datoteku čije se ime zadaje kao prvi argument komandne linije, zapisati sumu elemenata iznad glavne dijagonale.

4. Napisati program koji sa standardnog ulaza unosi prvo broj studenata a zatim i podatke o studentima (ime studenata - karakterska niska dužine do 30 karaktera i broj indeksa studenta - ceo broj), sortira ih po imenu studenta leksikografski (pozivom funkcije qsort) i ispisuje sortiran niz na standardni ulaz.

Glava 5

Testovi i ispiti 2009/10

5.1 Programiranje 2, I smer, 2009/10, Kolokvijum 1

1. Napisati funkciju koja kao argumente prima kvadratnu matricu celih brojeva i njenu dimenziju, a vraća 1 ako je matrica donja trougaona, odnosno 0 ako nije. Pretpostavka je da je maksimalna dimenzija matrice 100. Matrica je donja trougaona ako se u gornjem trouglu (iznad glavne dijagonale, ne uključujući je) nalaze sve nule. Napisati prateći program koji omogućava učitavanje matrice iz datoteke čija je putanja zadata prvim argumentom komandne linije, a format datoteke je takav da je prvi pročitani broj dimenzija matrice, a potom sledi niz brojeva koji popunjavaju matricu sleva nadesno, odozgo na dole (vrstu po vrstu).

2. Napisati program koji omogućava učitavanje artikala iz datoteke koja se zadaje prvim argumentom komandne linije. Jedan artikal je definisan strukturom

   ```
   typedef struct {
     char ime[50];
     float tezina;
     float cena;
   } Artikal;
   ```

   Datoteka je ispravno formatirana i sadrži najviše 500 artikala. Program treba da, nakon učitavanja, ispiše sve artikle sortirane po zadatom kriterijumu. Kriterijum se zadaje kao drugi argument komandne linije i može biti: `i` - sortirati po imenu; `t` - sortirati po težini; `c` - sortirati po ceni.

   Podrazumeva se da je sortiranje u rastućem redosledu, a da se za sortiranje po imenu koristi leksikografski poredak (dozvoljeno je korišćenje funkcije `strcmp`). Koristiti generičku funkciju za sortiranje `qsort` iz standardne biblioteke.

3. Napisati funkciju sa prototipom

   ```
   double treci_koren(double x, double a, double b, double eps);
   ```

   koja, metodom polovljenja intervala, računa treći koren zadatog broja x ($x \geq 1$), tj. za dato x određuje broj k za koji važi $k^3 = x$, sa tačnošću *eps* i sa početnom pretpostavkom da se broj k nalazi u datom intervalu $[a, b]$. Napisati prateći program koji omogućava korisniku da unese broj x i traženu tačnost. Korisnik ne unosi početnu procenu intervala, a za početni interval može se uzeti $[0, x]$. Ispisati poruku o grešci ako je uneto x manje od 0.

5.2 Programiranje 2, I smer, 2009/10, Završni ispit (juni 2010)

1. Napisati program koji sa standardnog ulaza učitava pozitivne cele brojeve dok ne učita nulu kao oznaku za kraj. Na standardni izlaz ispisati koji broj se pojavio najviše puta među tim brojevima. Na primer, ako se na ulazu pojave brojevi: 2 5 12 4 5 2 5, program treba da ispiše broj 5. Zadatak resiti korišćenjem dinamičke realokacije i strukture koja sadrži ceo broj i broj njegovih pojavljivanja. Složenost izvršavanja napisanog programa nije bitna.

2. Napisati funkciju `cvor* nova_lista(cvor* L1, cvor* L2);` koja od dve date liste `L1` i `L2`, u kojima se čuvaju celi brojevi, formira novu listu koja koja sadrži alternirajuće raspoređene elemente iz lista `L1` i `L2` (prvi element iz `L1`, prvi element iz `L2`, drugi element `L1`, drugi element `L2` itd.) sve dok ima elemenata u *obe* liste. Ne formirati nove čvorove, već samo postojeće čvorove povezati u jednu listu, a kao rezultat vratiti početak te formirane liste.

3. Neka su čvorovi binarnog stabla koje opisuje aritmetički izraz opisani sledećom strukturom:

```
typedef struct tagCvor
{
    int tipCvora;
    int tipOperatora;
    int vrednostKonstante;
    char oznakaPromenljive;

    struct tagCvor *levo;
    struct tagCvor *desno;
} CVOR;
```

Član `tipCvora` moze imati sledeće vrednosti:

0 - čvor je operator i mora imati oba deteta;

1 - čvor je konstanta i ne sme imati decu, vrednost je u članu `vrednostKonstante`;

2 - čvor je promenljiva sa oznakom koju definiše član `oznakaPromenljive` i ne sme imati decu.

Član `tipOperatora` može imati sledeće vrednosti:

0 - sabiranje

1 - množenje

(a) Napisati funkciju `void dodela(CVOR* koren, char promenljiva, int vrednost);` koja menja izraz dat korenom u izraz u kojem je svako pojavljivanje promenljive sa datom oznakom zamenjeno konstantom sa datom vrednošću.

(b) Napisati funkciju `int imaPromenljivih(CVOR *koren);` koja ispituje da li izraz dat korenom sadrži promenljive. Funkcija vraća vrednost različitu od nule ako izraz ima promenljivih, odnosno nulu u suprotnom.

(c) Napisati funkciju `int vrednost(CVOR *koren, int* v);` koja na adresu na koju pokazuje drugi argument funkcije smešta vrednost izraza datog korenom `koren` i vraća vrednost `0`, u slučaju da izraz nema promenljivih. U slučaju da izraz ima promenljivih funkcija treba da vraća `-1`.

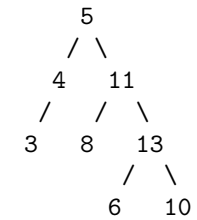
Podrazumevati da je drvo izraza koje se prosleđuje funkcijama ispravno konstruisano.

5.3 Programiranje 2, I smer, 2009/10, Završni ispit (septembar 2010)

1. Napisati funkciju `void zamene(CVOR** p)` koja menja povezanu listu koja je zadata svojim početkom tako da zameni mesta 1. i 2. elementu, potom 3. i 4. itd. Na primer, lista `1->2->3->4` postaje `2->1->4->3`. Lista `1->2->3->4->5->6` postaje `2->1->4->3->6->5`. Lista može sadržati i neparan broj elemenata, pri čemu u tom slučaju poslednji ostaje na svom mestu. Nije dozvoljeno formiranje nove liste - lista se može jedino preurediti u okviru postojećih struktura.

2. Dato je binarno stablo koje sadrži cele brojeve. Napisati funkciju `CVOR* nadjiNajblizi(CVOR* koren)` koja vraća pokazivač na čvor koji je najbliži korenu i pri tom je deljiv sa 3. Ako ima više čvorova na istom rastojanju od korena koji zadovoljavaju svojstvo, vratiti pokazivač na bilo koji.

   Na primer, u binarnom stablu

   Čvorovi sa vrednostima 3 i 6 zadovoljavaju uslov, ali je 3 bliži korenu, pošto su potrebna dva poteza da bi se stiglo do njega, odnosno 3 poteza da bi se stiglo do broja 6.

3. Data je datoteka `apsolventi.txt`. U svakom redu datoteke nalaze se podaci o apsolventu: *ime* (ne veće od 20 karaktera), *prezime* (ne veće od 20 karaktera), *broj preostalih ispita*. Datoteka je dobro formatirana i broj redova datoteke nije poznat. Potrebno je učitati podatke iz datoteke, odrediti prosečan broj zaostalih ispita i potom ispisati imena i prezimena studenta koji imaju veći broj zaostalih ispita od prosečnog u datoteku čije ime je zadato kao argument komadne linije. NAPOMENA: koristiti strukturu

```
typedef struct {
     char ime[20];
     char prezime[20];
     int br_ispita;
} APSOLVENT;
```

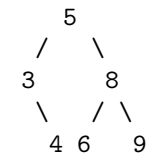
5.4 Programiranje 2, I smer, 2009/10, Završni ispit (oktobar 2010)

1. Argument programa je putanja tekstualne datoteke koja sadrži isključivo cele brojeve. Napisati program koji pronalazi i ispisuje na standardnom izlazu dva broja koja se najmanje razlikuju (ako ima više parova koji se isto razlikuju, ispisati bilo koji par). Uputstvo: koristiti funkciju `qsort`.

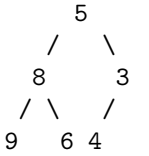
2. Neka je čvor binarnog stabla definisan kao

```
typedef struct cvorStabla
{
  int broj;
  struct cvorStabla* levi;
  struct cvorStabla* desni;
} CVOR;
```

Napisati funkciju `void obrni(CVOR* koren);` koja menja mesto levom i desnom podstablu za svaki čvor. Na primer, stablo



bi posle transformacije postalo:



Dozvoljeno je isključivo reorganizovanje strukture drveta pomoću postojećih pokazivača, ne i formiranje novog drveta.

3. Data je lista. Svaki čvor liste opisan je strukturom:

```
typedef struct CVOR
{
    int vr;
    struct CVOR* sled;
}cvor;
```

Napisati funkciju `void udvaja(cvor* lista, int br)` koja udvaja svako pojavljivanje elementa `br` u listi `lista`.
Npr. za listu `1->7->6->7->1->4->7` i br = 7 potrebno je dobiti listu: `1->7->7->6->7->7->1->4->7->7`.

5.5 Završni ispit, Oktobar 2 2010.

1. Date su dve liste. Čvor liste opisan je strukturom

```
tupedef struct cvorliste
{
    int vr;
    struct cvorliste *sled;
}cvor;
```

Napisati funkciju `void uredjuje(cvor **lista1, cvor* lista2)` koja izbacije iz liste `lista1` sve elemente koji se nalaze u listi `lista2`.

2. Ime tekstualne datoteke zadaje se kao argument komadne linije. U svakom redu datoteke nalazi se ime proizvoda (ne više od 20 karaktera) i količina u kojoj se proizvodi (broj redova datoteke nije poznat). Proizvodi su leksikografski poređani. Sa standardnog ulaza učitava se ime proizvoda. Korišćenjem sistemske funkcije `bsearch` pronaći u kojoj meri se dati proizvod proizvodi. NAPOMENA: koristiti strukturu

```
tupedef struct
{
    char ime[20];
    int kolicina;
}proizvod;
```

3. Binarno drvo je perfektno balansirano akko za svaki čvor važi da se broj čvorova levog i desnog podrveta razlikuje najviše za jedan. Napisati funkciju koja proverava da li binarno drvo perfektno balansirano.

Glava 6

Testovi i ispiti 2010/11

6.1 Programiranje 2, I smer, 2010/11, Test 2 (zadaci)

6.1.1 I grupa

1. Napisati rekurzivnu funkciju koja iz datog broja n izbacuje pojavljivanja svih parnih cifara koje su se nalaze na parnim mestima broja n, i svih neparnih cifara koje se nalaze na neparnim mestima broja n. Mesto cifre u broju čitati sdesna na levo, počev od indeksa 1. Odrediti vremensku složenost algoritma.

6.1.2 II grupa

1. Napisati rekurzivnu funkciju koja za zadato k i n crta "stepenice". Svaka stepenica je širine k, a ima n stepenika. Na primer k = 4, n = 3:

   ```
   ****
       ****
           ****
   ```

   Izračunati vremensku složenost algoritma.

6.1.3 III grupa

1. Dat je broj n i neka su $a_1, \ldots, a_k$ sleva na desno cifre broja n. Napisati rekurzivnu funkciju koja izračunava: $a_1 + 2 * a_2 + \ldots + k * a_k$. Izračunati vremensku složenost.

6.2 Programiranje 2, I smer, 2010/11, Test 3 i Test 4 (zadaci)

6.2.1 I grupa

1. Napisati rekurzivnu funkciju koja svaku parnu cifru c u broju n zamenjuje sa $c/2$. Napisati glavni program koji kao argument komandne linije dobija broj n, a na standardni izlaz ispisuje novi broj.

2. Sa standardog ulaza se zadaje ime tekstualne datoteke koja sadrži podatke o artiklima prodavnice. Datoteka je u formatu:
   `<bar kod> <ime artikla> <proizvodjac> <cena>` i sortirana je prema `<bar kod>`. Nije unapred po-znat broj artikala u datoteci. Učitati podatke o artiklima u niz (niz alocirati dinamički). Zatim se sa standardnog ulaza unose bar kodovi artikla sve dok se ne unese 0. Izračunati ukupnu cenu unetih proizvoda. (koristiti ugradjenu f-ju `bsearch` za traženje artikla sa datim bar kodom).

6.2.2 II grupa

1. Napisati rekurzivnu funkciju koja svaku parnu cifru c u broju n zamenjuje sa $c/2$. Napisati glavni program koji kao argument komandne linije dobija broj n, a na standardni izlaz ispisuje novi broj.

2. Sa standardog ulaza se zadaje ime tekstualne datoteke koja sadrži podatke o artiklima prodavnice. Datoteka je u formatu:
`<bar kod> <ime artikla> <proizvodjac> <cena>` i sortirana je prema `<bar kod>`. Nije unapred po-znat broj artikala u datoteci. Učitati podatke o artiklima u niz (niz alocirati dinamički). Zatim se sa standardnog ulaza unose bar kodovi artikla sve dok se ne unese 0. Izračunati ukupnu cenu unetih proizvoda. (koristiti ugradjenu f-ju `bsearch` za traženje artikla sa datim bar kodom).

6.3 Programiranje 2, I smer, 2010/11, Test 7 (zadaci)

6.3.1 I grupa

1. Napisati f-ju koja invertuje n bitova počevši od pozicije p u broju x.

2. Data je struktura student:

```
typedef struct
{
    char ime[20];
    char prezime[20];
}student;
```

   Napisati funkciju
   `void pronadji(student *studenti, int n, int *max_p, int *max_p_i)`, koja bibliotečkim qsort–om, sortira niz, prvo prema pema prezimenu, a zatim i prema imenu, a zatim u sortiranom nizu studenata pronalazi maksimalan broj studenata koji imaju isto prezime i maksimalan broj studenata koji imaju isto ime i prezime. Ove podatke vraća kroz prosleđene parametre max_p i max_p_i.

3. Napisati funkciju `Cvor* dodaj_u_listu(Cvor *glava, Cvor *novi, int n)` koja dodaje novi čvor na n-to mesto u listi. Ukoliko lista ima manje od n elemenata novi čvor se dodaje na kraj liste. Kao rezultat funkcija vraća adresu nove glave liste.

6.3.2 II grupa

1. Napisati funkciju koja izračunava razliku suma bitova na parnim i neparnim pozicijama broja x.

2. Definisati tip podataka koji definiše tačku u ravni. Sa standardnog ulaza se unosi broj tačaka , a zatim i njihove koordinate. Maximalni broj tačaka nije unapred poznat. Sortirati dati niz tačaka na osnovu rastojanja od koordinatnog početka.

3. Napisati funkciju `void umetni(cvor* lista)` koja između svaka dva elementa u listi umeće element koji predstavlja razliku susedna dva.

6.4 Programiranje 2, I smer, 2010/11, završni ispit, juni 2011

1. a) Definisati tip podataka za predstavljanje studenata, za svakog studenta poznato je: nalog na Alasu (oblika napr. mr97125, mm09001), ime (maksimalne dužine 20 karaktera) i broj poena.

b) Podaci o studentima se nalaze u datoteci `studenti. txt` u obliku: nalog ime br.poena. Kao argument komadne linije korisnik unosi opciju i to može biti `-p` ili `-n`. Napisati program koji sortira korišćenjem ugrađene funkcije `qsort` studente i to: po broju poena ako je prisutna opcija `-p`, po nalogu ukoliko je prisutna opcija `-n`, ili po imenu ako nije prisutna ni jedna opcija. Studenti se po nalogu sortiraju tako što se sortiraju na osnovu godine, zatim na osnovu smera i na kraju na osnovu broja indeksa.

2. Grupa od n plesača (na čijim kostimima su u smeru kazaljke na satu redom brojevi od 1 do n) izvodi svoju plesnu tačku tako što formiraju krug iz kog najpre izlazi k-ti plesač (odbrojava se pov cev od plesača označenog brojem 1 u smeru kretanja kazaljke na satu). Preostali plesači obrazuju manji krug iz kog opet izlazi k-ti plesač (odbrojava se počev os sledećeg suseda prethodno izbačenog, opet u smeru kazaljke na satu). Izlasci iz kruga se nastavljaju sve dok svi plesači ne budu isključeni. Celi brojevi `n, k, (k<n)` se učitavu sa standardnog ulaza. Napisati program koji će na standardni izlaz ispisati redne brojeve plesača u redosledu napuštanja kruga. PRIMER: za `n = 5, k = 3` redosled izlazaka je 3 1 5 2 4. NAPOMENA: Zadatak rešiti korišćenjem kružne liste.

3. Ime datoteke je argument komadne linije. U svakom redu datoteke nalazi se podatak: broj ili ime druge datoteke (zavrsava se sa .txt). Svaka datoteka je istog sadržaja. Napisati program koji posećuje sve datoteke do kojih može stići iz prve (rekurzivno), pri tome ne posećuje istu datoteku dva puta i štampa na standardni izlaz brojeve iz njih. NAPOMENA: Za pamćenje imena datoteka koristiti binarno pretraživačko drvo.

6.5 Programiranje 2, I smer, 2010/11, završni ispit, septembar 2011

1. a) Napisati funkciju `int skalarni_proizvod(int* a,int* b, int n)` koja računa skalarni proizvod vektora a i b dužine n. (Sklarni proizvod dva vektora $a = (a_1, \ldots, a_n)$ i $b = (b_1, \ldots, b_n)$ je suma $S = a_1 \cdot b_1 + a_2 \cdot b_2 + \ldots + a_n \cdot b_n$)

   b) Napisati funkciju `int ortonormirana(int** A, int n)` kojom se proverava da li je zadata kvadratna matrica A dimenzije $n \times n$ ortonormirana. Matrica je ortonormirana ako je skalarni proizvod svakog para različitih vrsta jednak 0, a skalarni proizvod vrste sa samom sobom 1. Funkcija vraća 1 ukoliko je matrica ortonormirana, 0 u suprotnom.

   c) Napisati glavni program u kome se dimenzija matrice `n` zadaje kao argument komandne linije. Nakon toga se elementi matrice učitavaju sa standardnog ulaza i pozivom funkcije se utvrđuje da li je matrica ortonormirana. Maksimalna dimenzija matrice nije unapred poznata.

2. Iz datoteke `brojevi.txt` se učitavaju celi brojevi u niz i nije poznato koliko ima brojeva u datoteci. Brojeve sortirati pozivom ugrađene funkcije `qsort` pa onda ugrađenom funkcijom `bsearch` pronaći ceo broj koji se zadaje sa standardnog ulaza.

3. Dva binarna stabla su identična ako su ista po strukturi i sadržaju, odnosno oba korena imaju isti sadržaj i njihova odgovarajuća podstabla su identična. Napistati funkciju koja proverava da li su dva binarna stabla identična.

6.6 Programiranje 2, I smer, 2010/11, završni ispit, oktobar 2011

1. U svakom redu datoteke `transakcije.txt` nalazi se identifikacija (niska maksimalne dužine 20) korisnika banke i iznos transakcije koju je korisnik napravio (ceo broj). Jedan korisnik može imati više transakcija, a svaka je predstavljena celim brojem (negativan - isplata sa računa, pozitivan - uplata na račun). Ispisati identifikacioni broj korisnika koji je najviše zadužen.

NAPOMENA: Kreirati strukturu `klijent`, učitati sve korisnike u dinamicki alociran niz koji je u svakom trenutku sortiran po identifikaciji (obavezno koristiti ugrađenu funkciju `qsort` i za pronalaženje korisnika ugrađenu funkciju `bsearch`), a zatim u jednom prolasku kroz niz naći najzaduženijeg korisnika.

2. Napisati funkciju koja sažima listu tako što izbacuje svaki element koji se više puta pojavljuje u listi. PRIMER: zadata lista: 1 3 8 3 1 2 3 6; rezultat: 8 2 6 dj

3. Neka je dat pokazivač na koren binarnog stabla čiji čvorovi sadrže cele brojeve. Napisati sledeće funkcije:

   (a) Funkciju koja vraća broj čvorova koji su po sadržaju veći od svih svojih potomaka.

   (b) Funkciju koja ispisuje čvorove koji su veći od sume svih svojih potomaka.

Glava 7

Testovi i ispiti 2011/2012

7.1 Programiranje 2, I smer, 2011/12, Popravi test

7.1.1 Deo II - Grupa 1

Sa standardnog ulaza unosi se lista celih brojeva dok se ne unese 0, a potom se unosi jedan broj. Odrediti poziciju prvog pojavljivanja broja u listi i ispisati na standardni izlaz. Ukoliko broja nema u listi ispisati -1. Pozicija u listi se broji počevši od 1.

U datoteci `liste.h` nalaze se funkcije za rad sa listom:

```
void oslobodi(cvor* lista)
cvor* ubaci_na_pocetak(cvor* lista, int br)
cvor* novi_cvor(int br)
void ispis(cvor* lista)
```

Napraviti main.c u kome se testira rad programa. U main.c treba da stoji `#include "liste.h"`.
Kompaliranje: `gcc main.c liste.c`
Pokretanje: `./a.out`

NAPOMENA: Ukoliko se zadatak uradi bez korišćenja liste broj osvojenih poena je 0.
Primeri:

```
Primer 1:                       Primer 2:
ulaz: -20 7 -31 4 5 6 0 -31     ulaz: 4 90 234 -8 0 322

izlaz: 3                        izlaz: -1


Primer 3:                       Primer 4:
ulaz: 9 7 -42 7 343 7 0 7       ulaz: 8 90 8 56 3 2 0 8

izlaz: 2                        izlaz: 1
```

7.1.2 Deo II - Grupa 1

Sa standardnog ulaza unosi se lista celih brojeva dok se ne unese 0. Odrediti broj pojavljivanja datog broja u listi.

U datoteci `liste.h` nalaze se funkcije za rad sa listom:

```
void oslobodi(cvor* lista)
cvor* ubaci_na_pocetak(cvor* lista, int br)
cvor* novi_cvor(int br)
void ispis(cvor* lista)
```

Napraviti main.c u kome se testira rad programa. U main.c treba da stoji `#include "liste.h"`.
Kompaliranje: `gcc main.c liste.c`
Pokretanje: `./a.out`

NAPOMENA: Ukoliko se zadatak uradi bez korišćenja liste broj osvojenih poena je 0.

Primeri:

```
Primer 1:                    Primer 2:
ulaz: 56 -34 234 0 -34          ulaz: 3456 -67 -23 -67 45 0 -67

izlaz: 1                        izlaz: 2


Primer 3:                    Primer 4:
ulaz: 89 0 567 -34 0 2       ulaz: 907 8 8 2 3 8 45 0 8

izlaz: 0                     izlaz: 4
```

7.2 Programiranje 2, I smer, 2011/12, Završni ispit — jun

7.2.1 Deo II

1. Kao argumenti komandne linije zadaju su brojevi, nepoznato koliko. Napisati funkciju `int f1(char** brojevi, unsigned int broj_brojeva)` koja računa njihov zbir.

   Primeri:

   ```
   Primer 1:          Primer 2:          Primer 3:              Primer 4:

   ./a.out 3 4        ./a.out            ./a.out 3 -56 231      ./a.out 0 561 -34 2 34 56

   7                  0                   178                   619
   ```

2. U datoteci `tekst.txt` nalazi se tekst. Ako datoteka ne postoji ispisati -1 na standardni izlaz. Napisati funkciju `int f2()` koja otvara datoteku, prebrojava i vraća broj cifara koje se pojavljuju u datoteci.

   Primeri (dati tekst se nalazi u datoteci `tekst.txt`):

   ```
   Primer 1:                          Primer 2:
   danas je 5 puta lepsi dana         od 2011. godine moze
   nego juce                          se putovati eko-taksijem kroz Bec
   ```

```
rez: 1                                  rez: 4



Primer 3:                               Primer 4:
ovde nema cifara                        a ov23de se cif89re
                                        nalaze u 8 recima

rez: 0                                  rez: 5
```

3. Napisati funkciju `int f3()` koja učitava cele brojeve sa standarnog ulaza sve dok se ne učita 0. Broj brojeva nije unapred poznat. Funkcija vraća broj brojeva većih od proseka. Ako nema unetih brojeva funkcija treba da vrati 0.

   Primeri:

```
Primer 1:           Primer 2:       Primer 3:           Primer 4:
2 3 4 5 0           0               56 -45 231 -34 0    76 -45 2 0

rez: 2              rez: 0          rez: 2              rez: 1
```

4. Data je datoteka `voce.txt` koja ima najviše 100 redova. U svakom redu se nalazi ime voćke (ne duže od 20 karaktera) i cena (int). Napisati funkciju `int f4(int p, char* ime, int** cena)` koja otvara datoteku i učitava podatke iz nje u niz struktura i sortira taj niz po cenama. U promenljive `ime` i `cena` upisuju se ime voćke i cena voćke koja se nalazi na poziciji `p` u sortiranom nizu. Ukoliko `p` ima nekorektnu vrednost funkcija vraća -1 a inače vraća 1. `p` je nekorektno ako je manje od 0 ili veće ili jednako od broja elemenata niza.

   Primeri(podaci o voćkama su u datoteci `voce.txt`):

```
Primer 1:           Primer 2:           Primer 3:           Primer 4:

jabuka 10           ananas 12           jagode 14           limun 45
kruska 15           kivi 34             breskve 23          grejpfrut 23
sljiva 23                               pomorandze 11
malina 7            p = 5               banane 3            p = -1

p = 2               funkcija vraca -1;  p = 1               funkcija vraca -1;

ime = kruska;                           ime = pomorandze
cena = 15;                              cena = 11
funkcija vraca 1;                       funkcija vraca 1;
```

5. Napisati funkciju `unsigned int f6(unsigned int x)` koja vraća broj dobijen izdvajanjem prvih 8 bitova broja (bitovi na najvećim tezinama), a ostatak broja popunjava jedinicama. Testirati pozivom u main-u.

   Primeri:

```
Primer 1:           Primer 2:           Primer 3:           Primer 4:
x = 88888899        x = 679012345       x = 34              x = 837312000

rez: 100663295      rez: 687865855      rez: 16777215       rez: 838860799
```

6. Napisati funkciju `int f7(cvor* lista)` koja menja elemente liste na sledeći način: ako je tekući element paran, sledeći element uvećava za 1, a ako je element neparan sledeći element smanjuje za 1. Parnost broja se odnosi na tekuću, promenjenu vrednost broja. Funkcija vraća vrednost poslednjeg elementa liste.

   Primeri:

```
Primer 1:                Primer 2:
1->2->3->4->5->NULL      6->78->2->3->1->NULL

rez: 4                   rez: 2



Primer 3:                        Primer 4:
-87->9->45->2->2->NULL           22->-34->0->2->1->NULL
rez: 1                           rez: 0
```

7. Napisati funkcju `int f8(cvor* drvo, int nivo)` koja vraća broj elemenata drveta koji se nalaze na nivou `nivo` (`nivo>0`). Testirati pozivom u main-u.

   Primeri:

```
Primer 1:                Primer 2:
sa standardnog ulaza     sa standardnog ulaza
se unose redom:          se unose redom:
6 4 12 3 10 32           7 2 10 23

      6                     7
     / \                   / \
    4   12                2   10
   /   /  \                     \
  3   10  32                    23

nivo: 3                  nivo: 1

rez: 3                   rez: 1



Primer 3:                Primer 4:
sa standardnog ulaza     sa standardnog ulaza
se unose redom:          se unose redom:
10 12                    7 4 20 3 15 23 2

      10                    7
        \                  / \
         12               4   20
                         /   /  \
nivo: 3                 3   15   23
                       /
rez: 0                2
```

```
nivo: 2

rez: 2
```

7.3 Programiranje 2, I smer, 2011/12, Završni ispit — juli

7.3.1 Deo II

1. Argumenti komadne linije su opcija (`-m`, `-v` ili `-mv`) i reč. Ukoliko je opcija `-m` pretvoriti sva slova reči u mala slova, ukoliko je opcija `-v` pretvoriti sva slova u reči u velika slova, a ukoliko je opcija `-mv` pretvoriti sva mala slova u velika, a sva velika slova u mala slova. Ukoliko opcija nije zadata ili je netačno navedena ispisati -1.

```
ulaz: ./a.out -m Dan28Mesec2        ulaz: ./a.out -v DanDanas;Pamtim

izlaz: dan28mesec2                  izlaz: DANDANAS;PAMTIM
-------------------------------------------------------------------------------

ulaz: ./a.out -mv VArljivoLeto68    ulaz: ./a.out -nn greska

izlaz: vaRLJIVOlETO68               izlaz: -1
```

2. Napisati funkciju `void f2(char* ime, int n)` koja preko argumenata komandne linije dobija ime datoteke i ceo broj n. Iz datoteke ispisati na standardni izlaz svaku n-tu reč. Ukoliko argumenti nisu pravilno zadati ispisati -1. Greške mogu biti da nije zadato ime datoteke ili ceo broj. Pretpostavlja se da ako je unešen drugi argument je sigurno ceo broj. Ukoliko datoteka ne postoji ispisati -1.

```
ulaz: ./a.out primer1.txt 3             ulaz: ./a.out 4

izlaz: Volfganga Nemca koji godine,     izlaz: -1
       da pravi zarobljenik veliki
-----------------------------------------------------------------------------------------
ulaz: ./a.out                           ulaz: ./a.out primer2.txt 10

izlaz: -1                               izlaz: Ban Bil mir
```

3. Sa standarnog ulaza unosi se ceo pozitivan broj i pozitivan broj `p`. Napisati funkciju `unsigned int f3(unsigned int x,unsigned int p)` koja menja mesta prvih i poslednjih `p` bitova (npr. p = 3 i broj x = 111....010 treba da se dobije izlaz x = 010...111)

```
x = 1234567890      x = 567890567       x = 7484            x = 827262627
p = 5               p = 7               p = 4               p = 10

izlaz: 2442527433   izlaz: 265900688    izlaz: 3221232944    izlaz: 2832139461
```

4. U datoteci `podaci.txt` se nalaze reči i pozitivni celi brojevi. Napisati funkciju `void f4(int m, int n)` koja sortira leksikografski reči u rastućem poretku i ispisuje reč na poziciji `m`. Potom sortirati brojeve u opadajućem poretku i ispisati broj na poziciji `n`. Ukoliko datoteka ne postoji ispisati -1. Ukoliko `m` i `n` nisu u odgovarajućem opsegu ispisati -1. Reči su maksimalne dužine 20 karaktera.

```
podaci.txt: Tekst 32 je sa 67 brojevima 166
            i 890 zato 2 je cudan.
m = 2
n = 2

izlaz: cudan
       67
```

```
podaci.txt: Drugi 789
            interesantni 56 3 4
            primer
m = 4
n = 2

izlaz: -1
```

```
podaci.txt: 38 17 Londonski 67danas umetnik Patrik Vejl
            postao je 92 internet senzacija 14 42.
m = 4
n = 1

izlaz: internet
       42
```

5. Napisati funkciju `void f5(cvor* l1, cvor* l2)` koja spaja dve rastuće sortirane liste tako da rezultat i dalje bude sortiran. Rezultat sačuvati u prvoj listi. Rezulatujuću listu ispisati na standardni izlaz. Lista se učitava sve dok se ne unese 0.

```
(napomena: lista se unosi obrnutim redosledom i na kraju se unosi 0)

ulaz: 8->9->15->62->NULL
      15->67->100->102->NULL

izlaz: 8->9->15->15->62->67->100->102->NULL
```

```
ulaz: 8->9->15->62->NULL
      2->4->16->NULL

izlaz: 2->4->8->9->15->16->NULL
```

6. Napisati funkciju koja iz datoteke `karakteri.txt` ucitava karaktere i smešta ih u binarno pretraživacko drvo. Uz svaki karakter čuvati i broj pojavljivanja karaktera u datoteci. Ispisati na standardni izlaz karakter koji se pojavljuje najveći broj puta u datoteci.

```
karakteri.txt: Danas je lep dan.

izlaz: a
```

```
karakteri.txt: 78665555512

izlaz: 5
```

```
karakteri.txt: U Datoteci
               je probni test
               za Zadatke sa ispita.

izlaz: a (izlaz moze biti i: t, ' ' (blanko znak))
```

Programiranje 2, I smer, 2010/11, završni ispit, oktobar 2012

1. Napisati funkciju `int f1(int x, int p, int q)` koja kao argumente dobija 3 cela broja `x`, `p`, `q`, a kao rezultat vraća broj koji je jednak broju `x` kome je invertovan svaki drugi bit između pozicije `p` i `q`, čitano sa desna na levo. Ukoliko `p` ili `q` izlaze iz opsega ili `p` > `q` vratiti `-1` kao rezultat.

```
Primer 1:
ulaz: 3456 2 10

izlaz: 2260
```

```
Primer 2:
ulaz: -895 13 20

izlaz: -697215
```

```
Primer 3:
ulaz: 678910 21 31

izlaz: -1431675906
```

```
Primer 4:
ulaz: 657 19 33

izlaz: -1
```

2. Napisati funkciju `cvor* f2(cvor* L)` koja dobija glavu liste L kao argument, obrće listu L i vraća novu glavu.

```
ulaz: 4->1->2->67->0       ulaz: 0          ulaz: -56->28->31->-1000->0    ulaz: 10->9->8->1000000
izlaz: 67 2 1 4            izlaz:           izlaz: -1000 31 28 -56         izlaz: 1000000 8 9 10
```

3. Stablo se učitava tako da su u listovima realni brojevi, a u unutrasnjim cvorovima neka od operacija `+`, `-`, `*` i `/`. Funkcija `float f3(svor* drvo)` računa izraz u stablu i vraća rezultat. Ukoliko dođe do deljenja nulom funkcija vraća -1.

```
Primer 1:          Primer 2:            Primer 3:                Primer 4:

ulaz: + 3 4        ulaz: + * 5 4 2      ulaz: + 3 * * 5 4 3      ulaz: / + 5 4 3
izlaz: 7           izlaz: 22            izlaz: 63                izlaz: 3
```

4. Argumenti komandne linije su cu celi brojevi `a`, `n`, `m`, `p`. Napisati program koji računa niz

```
a mod n,
2a mod n,
3a mod n
...
m*a mod n
```

sortira ga u rastućem poretku i ispisuje u datoteku `rez.txt` p-ti član niza (brojanje indeksa pocinje od 0)

```
Primer 1:                Primer 2:                  Primer 3:
ulaz: ./a.out 5 6 3 1    ulaz: ./a.out 23 10 5 2    ulaz: ./a.out 10 10 100 45
rez.txt: 4               rez.txt: 5                 rez.txt: 0
-------------------------------------------------------------------------------------------------
```

```
Primer 4:
ulaz: ./a.out 634 2 34 67
rez.txt: -1
```

Ukoliko je došlo do neke od grešaka (nije tačan broj argumenata komandne linije, p>n,...) u `rez.txt` upisati `-1`.

5. Argumenat komandne linije su ime datoteke i ceo broj `n`. Napisati funkciju `void f5(char* ime_dat, int n)` koja ispisuje na standardni izlaz svaku `n`-tu reč iz datoteke. Ukoliko datoteka ne postoiji ili je broj `n` neispravno zadat ili nema dovoljno argumenenata komadne linije ispisati `-1`. Pretpostaviti da je maksimalna dužina reči 100.

```
Primer 1:                                          Primer 2:

./a.out dat.c 3                                    ./a.out p.txt 2
dat.c: Kada 7642oIO solja za kafu                  U sobi ima 2
       ekspres lonac, hleb i mleko.                kreveta, sto i racunar
       Spisak za kupovinu.                         tepih i jos neka sitna oprema.

izlaz: solja ekspres i za                          izlaz: sobi 2 sto racunar i neka oprema.
-------------------------------------------------------------------------------------------
```

```
Primer 3:

./a.out dat.c -67

izlaz: -1
```

```
Primer 4:

./a.out primer.txt

izlaz: -1
```

6. U datoteci `kupci.txt` se nalaze podaci o kupcima oblika `ime kolicina` gde je `ime` ime kupca koji je kupio proizvod, a `kolicina` količina proizvoda. Kupac se može više puta pojaviti u datoteci. Napraviti binarno pretraživačko drvo prema imenima kupaca. Na standarni izlaz ispisati ime onog kupca koji je uzeo najviše proizvoda. Pretpostavlja se da je datoteka dobro struktuirana.

```
kupci.txt:
pera 30
marko 10
jelena 14
marko 0

izlaz: pera
```

```
kupci.txt:
mika 10
mira 15
mika 6

izlaz: mika
```

```
kupci.txt:
zika 50
ana 70

izlaz: ana
```

```
kupci.txt:

izlaz:
```

7.4 Programiranje 2, I smer, 2011/12, Završni ispit — januar

7.4.1 Deo II

1. Napisati funkciju `float f5(float a, float b, float eps)` koja računa nulu funkcije $f(x) = 5 * sin(x) * ln(x)$ na intervalu `(a, b)` sa tačnošću `eps`. Brojevi `a`, `b` i `eps` unose se sa standardnog ulaza.
Napomena1: koristiti algoritam binarne pretrage
Napomena2: u `math.h` nalaze se `float sin(float)` za računanje sinusa i `float log(float x)` za računanje prirodnog logaritma. Testirati funkciju pozivom u main-u.

```
Primer 1:
0.9 2 0.01

0.998828
```

```
Primer 2:
2 4 0.000001

3.141593
```

```
Primer 3:
4 10 0.001

6.283142
```

```
Primer 4:
10 20 0.001

12.566376
```

2. Napisati funkciju koja pronalazi zbir parnih brojeva na neparnim pozicijama u nizu koji se zadaje kao argument komandne linije. Svi brojevi su pozitivni (ovo ne treba proveravati). Brojanje pozicija pocinje indeksom 0. Rezultat se ispisuje na standardni izlaz.

```
Primer 1:
\a.out 23 4 7 89 12

4
```

```
Primer 2:
\a.out

0
```

```
Primer 3:
\a.out 54 1 23 76 18 90 322 45678 12

 45844
```

```
Primer 4:
\a.out 18 17 16

0
```

3. Napisati program koji iz datoteke `brojevi.txt` učitava cele brojeve (nije unapred poznato koliko se brojeva nalazi u datoteci) pa zatim ceo broj sa standardnog ulaza. Program treba da vrati indeks tog broja u nizu (ako se broj nalazi u nizu) ili indeks i vrednost elementa niza koji je po apsolutnoj vrednosti najbliži tom broju. Ukoliko ne bude korišćena dinamička alokacija memorije zadatak neće biti priznat. Indeksiranje niza počinje od 0.

```
Primer 1:                                Primer 2:                          Primer 3:
brojevi.txt: 23 -4 56 13 4 56 7          brojevi.txt: -7 8 14 12            brojevi.txt: -7 8 14 12
56                                       -8                                 -13

2                                         1                                 2
```

```
Primer 4:
brojevi.txt: -7 8 14 12
56

2
```

4. Napisati funkciju `void f4(cvor* lista)` koja iz liste izbacuje svaki drugi element. U listi se nalaze celi brojevi. Lista se unosi sa standardnog ulaza sve dok se ne unese 0.

```
Primer 1:            Primer 2:        Primer 3:          Primer 4:
1 2 3 4 5 6 0        1 2 3 4 5 0      34 5 -78 23 0      -16 72 13 24 98 0

1 3 5                1 3 5            34 -78             -16 13 98
```

5. Napisati program koji u zavisnosti od toga da li je suma bitova broja parna ili neparna, šiftuje bitove broja za jedno mesto ulevo odnosno u desno.

```
Primer 1:          Primer 2:       Primer 3:       Primer 4:
3                  7               4567            8765

6                  3               9134            4382
```

6. Funkcija vrši rekurzivnu rotaciju drveta oko svih čvorova, dakle dobija se odraz prvobitnog drveta u ogledalu.

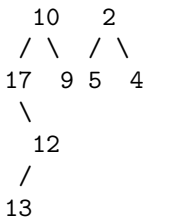
```
Primer 1:

    7
   / \
  2   10
 / \ / \
4  5 9  17
       /
      12
       \
        13

unos: 7 2 4 5 10 9 17 12 13 0
```

```
Primer 2:

    2
   / \
-10   8
        \
         7

unos: 2 -10 8 7 0
```

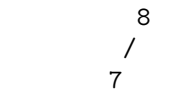
se transformiše u:

```
     7
    / \
```

```
      2
     / \
```

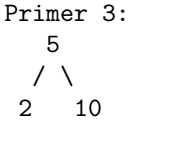


ispis: 7 10 17 12 13 9 2 5 4



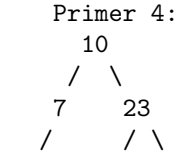
ispis: 2 8 7 -10

Primer 3:



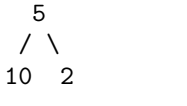
unos: 5 2 10 0

Primer 4:

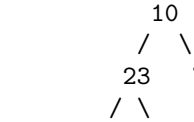


unos: 10 7 2 23 12 40 0

se transformiše u:



ispis: 5 10 2



ispis: 10 23 40 12 7 2

Glava 8

Testovi i ispiti 2012/13

8.1 I smer, Programiranje 2 2012/2013, Drugi test

8.1.1 (I Grupa)

NAPOMENA: Na desktopu napraviti direktorijum `InicijaliAsistenta_ImePrezime_BrojIndeksa_Grupa`, npr. `AK_MarkoMarkovic_mi12201_1`. Zadatke čuvati u ovom direktorijumi pod nazivom **1.c, 2.c**. U slučaju da zadaci budu sačuvani na nekom drugom mestu ili poddirektorijumu neće biti pregledani!

Zadatak 1 *Napisati funkciju koja sortira slova unutar niske karaktera i ispisuje ih na standardni izlaz. Napisati program koji proverava da li su dve niske karaktera koje se zadaju kao argumenti komandne linije anagrami. Dve niske su anagrami ako se sastoje od istog broja istih karaktera. Program treba na standardni izlaz da ispiše true ili false.*

```
Primer 1:    ./anagram miroljub ljubomir          bijlmoru bijlmoru true
Primer 2:    ./anagram vatra trava                aartv aartv true
Primer 3:    ./anagram racunar racun              aacnrru acnru false
Primer 4:    ./anagram program gram               agmoprr agmr false
```

Zadatak 2 *Napisati funkciju* `int count_zero_pairs(unsigned x)` *koja broji koliko se puta kombinacija* 00 *(dve uzastopne nule) pojavljuje u binarnom zapisu celog neoznačenog broja (ako se u bitskoj reprezentaciji nekog broja jave tri uzastopne nule, središnja se broji dva puta, u levom i desnom paru). Napisati program koji za broj iz prvog argumenta komandne linije na standardni izlaz ispisuje rezultat funkcije.*

```
Primer 1:   ./zeros 487                          23
Primer 2:   ./zeros 255                          23
Primer 3:   ./zeros 3567                         19
Primer 4:   ./zeros 1024                         29
```

8.1.2 (II Grupa)

NAPOMENA: Na desktopu napraviti direktorijum `InicijaliAsistenta_ImePrezime_BrojIndeksa_Grupa`, npr. `AK_MarkoMarkovic_mi12201_2`. Zadatke čuvati u ovom direktorijumi pod nazivom **1.c, 2.c**. U slučaju da zadaci budu sačuvani na nekom drugom mestu ili poddirektorijumu neće biti pregledani!

Zadatak 3 *Sa standardnog ulaza unosi se broj reči* `n`*, za kojim sledi* `n` *reči (u svakom redu po jedna). Nakon unosa reči unosi se i broj* `q`*. Reči nisu duže od 20 karaktera. Učitati reči u niz, a zatim ih sortirati. Ispisati reč na poziciji* `q`*. Kriterijum sortiranja je suma* `ascii` *kodova slova svake reči, poredak je rastući.*

```
Primer 1:               Primer 2:               Primer 3:               Primer 4:
3                       4                       4                       2
programiranje           test                    aaaaa                   funkcija
alokacija               kolokvijum              zzzzz                   C
memorija                ispit                   bbbbb                   1
3                       usmeni                  ttttt
                        1                       2
-------------------------------------------------------------------------------------
programiranje           test                    bbbbb                   C
```

Zadatak 4 *Napisati funkciju* `unsigned swap_pair(unsigned x, int i, int j)` *koja razmenjuje vrednosti bitova na pozicijama* `i` *i* `j`*, i vraća rezultat kao povratnu vrednost. Bit sa najmanjom težinom nalazi se na poziciji 0, bit do njega na poziciji 1, itd... Napisati i program koji testira ovu funkciju. Sa standardnog ulaza se unose redom broj kome se invertuju bitovi, i pozicije bitova, a rezultat funkcije se ispisuje na standardni izlaz.*

```
Primer 1:          Primer 2:          Primer 3:          Primer 4:
159 0 1            48 4 0             255 1 9            249 10 5
159                33                 765                1241
```

8.2 Programiranje II, Završni ispit, jun 2013.

1. Trougao je zadat duzinama svoje tri stranice. Napraviti strukturu koja opsuje trougao. Napisati funkciju koja računa površinu trougla. Kao argumenti komadne linije zadati su podaci o dva trougla. Ispisati koji trougao ima veću površinu (na standarni izlaz ispisati `prvi` ili `drugi`).

2. Sa standardnog ulaza se ucitava broj n a zatim i niz od n niski(svaka niska je maksimalne duzine 20 karaktera, duzina niza n nije unapred poznata niti ogranicena). Sortirati taj niz u opadajucem poretku pri cemu je kriterijum poredenja broj samoglasnika u reci. Ispisati sortirani niz na standardni izlaz. U slucaju greske ispisati -1"na standardni izlaz.

3. Napisati funkciju unsigned int fja(unsigned int x) koja vraca broj dobijen od broja x kada se prvih 8 bitova broja (bitovi na najvecim tezinama) postave na 0, poslednjih 4 bita(bitovi na najmanjim tezinama) se postave na 0110, a ostatak broja ostaje nepromenjen. Testirati pozivom u main-u.

4. Napisati funkciju `void umetni(cvor* lista)` koja izmedju svaka dva el- ementa u listi umece element koji predstavlja razliku susedna dva.

5. Napisati funkciju int prebroj(cvor* drvo) koja vraca broj elemenata stabla drvo koji su iste parnosti kao oba svoja sina(sva tri parna ili neparna). Ukoliko je cvor list ili ima samo jednog sina ne ulazi u zbir.

8.3 Programiranje II, Završni ispit, jun2 2013.

NAPOMENA: Na desktopu napraviti direktorijum sa imenom `inicjaliAsistenta_ImePrezime_brojIndeksa`. U tom direktorijumu čuvati zadatke – 1.c, 2.c, 3.c, 4.c. 5.c Na desktopu se nalazi folder `p2_jun` sa datotekama za rad sa listama i drvetom. Date datoteke potrebno je koristiti po pravilima za rad sa više datoteka.

Zadatak 5 *Ime datoteke dato je kao argument komandne linije. U datoteci zameniti svaku cifru sledećom* `0->1, 1->2,....` *Broj 9 zameniti sa* `0`*. U slučaju greške na standardni izlaz ispisati* `-1`*.*

```
Primer 1:                |    Primer 2:                    Primer 3:
./a.out dat.txt          |    ./a.out zad                  ./a.out test.dat
dat.txt: 123abc789       |    zad: 9202jsshsh 120          test.dat:  kazablanka je 1967. bila
                         |         qoqo 2782 20                       kako uhvatiti ziravu 89 puta
dat.txt: 234abc890       |                                            90 puta napisati euklidov algo
                         |     zad: 0313jsshsh 231
______________________   |          qoqo 3893 31
Primer 4:                |                                 test.dat: kazablanka je 2078. bila
./a.out                  |                                           kako uhvatiti ziravu 90 puta
                         |                                           01 puta napisati euklidov algor
 -1                      |
```

Zadatak 6 *Kao argumenti komandne linije dati su celi brojevi a, b, c, d, e. Binarnom pretragom naći ceo broj na intervalu $[d, e]$ nulu funkcije $a \cdot x^2 + b \cdot x + c$ sa tačnošću* `0.0001`*. U slučaju greške na standardni izlaz napisati* `-1`*. Ako su argumenti komandne linije ispravno dati podrazumevati da funkcija ima tačno jednu nulu na datom intervalu. Na standardni izlaz ispisati nađenu vrednost. Broj ispisati sa 3 decimale.*

```
Primer 1:               Primer 2:        Primer 3:                    Primer 4:
./a.out 5 8 -4 0 1      ./a.out 4        ./a.out 10 -2 -15 -2 0       ./a.out -3 6 10 0 4

0.400                   -1               -1.129                       3.082
```

Zadatak 7 *Napisati funkciju* `unsigned int f3(unsigned int x)` *koja vraća broj koji predstavlja odraz u ogledalu polaznog broja* `x`*. Testirati pozivom u main funkciji – sa standardnog ulaza se unosi broj, rezultat ispisati na standarni izlaz. Na primer, ako je ulaz broj čiji je binarni zapis* `00101`*, izlaz je broj čiji je binarni zapis* `10100`*.*

```
Primer 1:          Primer 2:          Primer 3:          Primer 4:
3                  1234               -90                82720000

3221225472         1260388352         1260388352         11302688
```

Zadatak 8 *Napisati funkciju* `Cvor* izbaci(Cvor *lista1, Cvor *lista2)` *koja iz* `liste1` *izbacuje sve elemente koji se pojavljuju u* `listi2`*. Testirati funkciju pozivom u main-u, sa standarnog ulaza se učitaju elementi prve liste sve dok se ne unese 0. Potom se učitavaju elementi druge liste sve dok se ne učita 0. Elemente liste dodavati na kraj. Potom pozvati funkciju i novodobijenu listu ispisati na standarni izlaz. Dozvoljeno je pravljenje nove liste.*

```
Primer 1:          Primer 2:          Primer 3:          Primer 4:
1 3 5 0            3 -90 2 8 7 0      7 8 0              0
3 4 6 7 0          8 3 -4 0           0                  10 34 5 67 1 2 0

izlaz: 1 5         izlaz: -90 2 7     izlaz: 7 8         izlaz:
```

Zadatak 9 *Napisati funkciju* `int ravnomerno_izbalansirano(Cvor *stablo)` *koja proverava da li je stablo ravnomerno izbalansirano. Stablo je ravnomerno izbalansirano ako za svaki čvor važi da je pozitivna razlika između dubine levog i dubine desnog podstabla najviše* `1`*. Testirati funkciju pozivom u main-u, stablo se učitava sve dok se ne unese 0. Ukoliko jeste izbalansirano ispisati* `1`*, a u suprotnom* `0`*.*

```
Primer 1:

     10
    /  \
   7    20
  / \   / \
 5  8  15  23

ulaz: 10 7 20 5 8 15 23 0
izlaz: 1
```

```
Primer 2:

     10
    /  \
   7    11
  /
 5
/
4

ulaz: 10 7 11 5 4 0
izlaz: 0
```

```
Primer 3:

ulaz: 0
izlaz: 1
```

```
Primer 4:

     10
    /  \
   7    20
  / \
 5   8

ulaz: 10 7 5 8 20
izlaz: 1
```

8.4 Programiranje II, Završni ispit, januar 2014.

Zadatak 10 *Argumenti komadne linije su ime datoteke i jedan ceo broj* `p` *(*`p` > 0*). U datoteku* `izlaz.txt` *prepisati svaki* `p`*-ti karakter iz ulazne datoteke. U slučaju greške ispisati -1.*

```
Primer 1:                                   Primer 2:
./a.out ulaz 3                              ./a.out ulaz 10
ulaz: danas pisemo                          ulaz: Na Sretenje 1804. godine podignut je
      programe razne                              Prvi srpski ustanak, a 1835.
      i mnogo nam je zanimljivo                   donet prvi demokratski ustav Srbije.

izlaz.txt: n sorreae o meami                izlaz.txt: jogr aodib
```

```
Primer 3:                                   Primer 4:
./a.out dat.txt -3                          ./a.out

-1                                          -1
```

Zadatak 11 *U datoteci* `slike.txt` *nalaze se podaci o slikama – ime slike (maksimalne dužine 20 karaktera, ne sadrži praznine, dozvoljeno je korišcenje specifikatora* `%s` *za unos imena) i veličina u kilobajtima (neoznačen ceo broj). Definisati strukturu koja opisuje sliku. Sortirati niz slika u opadajućem poretku pri čemu je kriterijum poređenja veličina slike. Na standarni izlaz ispisati imena slika sortiranog niza. Broj slika u datoteci nije unapred poznat (koristiti dinamičku alokaciju). U slučaju greške ispisati -1.*

```
Primer 1:                   Primer 2:                   Primer 3:           Primer 4:
slike.txt:                  slike.txt:                  slike.txt:          slike.txt:
krug 200                    Klod_Mone 400               p1 300              oblak 200
trougao 450                 Edvard_Munk 550             p2 400              livada.jpg 300
Ojlerov_krug 320            Salvador_Dali 320           p3 320              suma 220
Pashova_teorema 220         Gustav_Klimt 432                                planina 340
paralelogram 330            Rembrant 578                izlaz:
                            Francisko_Goja 349          p2                  izlaz:
izlaz:                      Paja_Jovanovic 490          p3                  planina
trougao                     Uros_Predic 390             p1                  livada.jpg
paralelogram                                                                suma
Ojlerov_krug                izlaz:                                          oblak
Pashova_teorema             Rembrant
krug                        Edvard_Munk
                            Paja_Jovanovic
                            Gustav_Klimt
                            Klod_Mone
                            Uros_Predic
                            Francisko_Goja
                            Salvador_Dali
```

Zadatak 12 *Napisati funkciju —*`unsigned int f3(unsigned int x, int k, int p)`*— koja u broju* `x` *komplementira* `k` *bitova, počevši od pozicije* `p`*. Bitovi se broje sa desna na levo, počevši od 0. Bitovi koji se komplementiraju idu od pozicije* `p` *do pozicije* `k + p - 1`*. Sa standarnog ulaza učitavaju se brojevi* `x`*,* `k` *i* `p`*. Na standarni izlaz ispisati izlaz funkcije* `f3`*. U slučaju greške ispisati -1.*

```
Primer 1:           Primer 2:           Primer 3:           Primer 4:
345 5 6             18903 10 5          456672 4 15         45 15 23

1689                13879               471656              -1
```

Zadatak 13 *Dat je polinom reprezentovan listom, tako da čvorovi liste sadrže koeficijente i stepene polinoma. Dati polinom je nesređen, tj. može sadržati više članova sa istim stepenom. Napisati funkciju* `cvor* f4(cvor* lista)` *koja sređuje polinom tako da:*

- *da se svaki stepen pojavljuje najviše jedanput*
- *da su svi koeficijenti koji su prisutni u listi različiti od 0*

U main-u se unosi broj elemenata liste, a potom i elementi liste (svaki element liste se dodaje na kraj). Ispisati na standarni izlaz polinom koja je dobijena sređivanjem ulazne liste.

```
Primer 1:
4
1 1
2 3
-1 1
1 3

3*x^3
```

```
Primer 2:
10
-1 2
3 3
1 2
-3 3
7 1
2 5
3 1
-1 5
-1 5
4 4

10*x^1 + 4*x^4
```

```
Primer 3:
4
3 1
2 3
5 1
3 3

8*x^1 + 5*x^3
```

```
Primer 4:
5
2 3
1 2
4 5
-2 3
2 2

3*x^2 + 4*x^5
```

Zadatak 14 *Napisati funkciju* `float f5(cvor* drvo)` *koja računa sumu svih elemenata drveta takvih da su veći od sume svojih direktnih potomaka. Ne računati čvorove koji nemaju ni jednog direktog potomka. Sa standarnog ulaza unose se elementi u drvo sve dok se ne unese 0. Elementi su drveta su realni brojevi zapisani u jednostrukoj tačnosti (koristiti* `float`*). Na standarni izlaz ispisati izlaz funkcije* `f5`*. Rezultat ispisati sa 3 decimale. Moguće je koristiti biblioteke za rad sa stablima (*`stbla.c` *i* `stabla.h`*), kompajlirati sa* `gcc 5.c stabla.c`*.*

```
Primer 1:
         7.51
       /      \
      -2        8.1
    /    \      /  \
   -8     5    8    9.1
   / \   / \         /
-10 -3 -1.5 6      8.5

ulaz:
7.51 -2 -8 5 -10 -3 -1.5 6 8.1 8 9.1 8.5 0

izlaz: 11.610
```

```
Primer 2:
         6
        /  \
      -4    8
     / \   /
   -11 5   7
     /
   -20

ulaz:
6 -4 8 -11 5 7 -20 0

izlaz: -1.000
```

```
Primer 3:
       7.82
       /  \
     -4   10
       \
        6

ulaz: 7.82 -4 10 6 0

izlaz: 7.820
```

```
Primer 4:
        -2
       /   \
     -10    4
     / \   /  \
  -13 -4 -1   4.1

ulaz: -2 -10 4 -13 -4 -1 4.1 0

izlaz: -8.000
```

Glava 9

Testovi i ispiti 2013/14

9.1 Programiranje 2, I smer, 2013/14, Test 4 tok 2 grupa 1 (prakticni)

1. Odrediti klasu složenosti narednog programskog koda:

```
int f(int n) {
  int i, a = 1;
  if (n == 0)
    return n;
  for (i = 0; i < n; i++) {
    if (i == 3)
      break;
    a *= i;
  }
  return a * f(n-1) * f(n-2);
}
```

2. Napisati rekurzivnu funkciju koja za za dati prirodan broj vraća broj kome je svaka cifra udvojena. Primer: za broj 123 vraća 112233.

3. U datoteci razdaljine.txt nalaze se podaci o udaljenosti gradova od Beograda. U svakom redu datoteke dato je ime grada (koje se sastoji iz jedne reči maksimalne dužine 20) i razdaljina od Beograda (kao ceo broj). Na standardni izlaz ispisati spisak gradova sa razdaljinama od Beograda u rastućem redosledu rastojanja. Zatim za unetu vrednost ispisati ime grada koji je na toj udaljenosti od Beograda ili -1 ukoliko takav grad ne postoji.

9.2 Programiranje 2, I smer, 2013/14, Test 4 tok 2 grupa 2 (prakticni)

1. Odrediti klasu složenosti narednog programskog koda:

```
int f(unsigned n){
   int i, k=5;
   int a=0;
```

```
    if(n==1)
        return 1;

    for(i=0; i<k; i++)
        a+=k;

    return a+f(n-1)+f(n-2);
}
```

2. Napisati rekurzivnu funkciju koja računa proizvod neparnih cifara datog prirodnog broja.

3. U datoteci koja se zadaje kao argument komandne linije, nalaze se koordinate tačaka u ravni, svaki red sadrži po dva broja. Na standardni izlaz ispisati sve tačke zajedno sa udaljenošću od koordinatnog početka u rastućem redosledu. Zatim se sa standarnog ulaza unosi vrednost razdaljine i ispisati tačku koja je na toj udaljenosti ili -1 ukoliko takava ne postoji.

9.3 Programiranje 2, I smer, 2013/14, Test 4 tok 2 grupa 3 (praktični)

1. Odrediti klasu složenosti narednog programskog koda:

```
int f(unsigned n){
    int i, j;
    int a=0;

    if(n==1)
        return 1;

    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            if (i<>j) a+=i*j;

    return a+f(n/2);
}
```

2. Napisati rekurzivnu funkciju koja brise svaku parnu cifru datog prirodnog broja. Primer: za broj 12345 vraća 135.

3. U datoteci `kvadrati.txt` dati su podaci o državama. U svakom redu nalazi se ime države (jedna reč, maksimalne dužine 20) i njena površina (realan, pozitivan broj). Ispisati imena država sortirana po površinama u opadajućem poretku. Potom sa standarnog ulaza se unosi jedna površina Na standarni izlaz ispisati ime grada koje ima tu površinu ili -1 ukoliko takva država ne postoji.

9.4 I smer, Programiranje 2 2013/2014, završni ispit, septembar, 2014

NAPOMENA: Na desktopu napraviti direktorijum sa imenom `inicijaliAsistenta_ImePrezime_brojIndeksa`. U tom direktorijumu čuvati zadatke – 1.c, 2.c, 3.c, 4.c, 5.c, 6.c

U direktorijumu `g1` nalaze se funkcije za rad sa listama (`liste.c` i `liste.h`) i funkcije za rad sa stablima (`stabla.c` i `stabla.h`).

Kompilacija: `gcc 5.c liste.c`
`gcc 6.c stabla.c`

Zadatak 15 *Napisati program koji ispisuje na standardni izlaz koliko elemenata u njegovoj komandnoj liniji jesu palindromi (reči koje su iste kada se čitaju sa leva na desno, i sa desna na levo).*

```
Primer 1:                              Primer 2:            Primer 3:    Primer 4:
./a.out ana_ana Milan MAMA TegeT W     ./a.out MaMa anka    ./a.out      ./a.out Ana H202H

3                                      0                    0            1
```

Zadatak 16 *U datoteci* `matrica.txt` *nalaze se podaci o kvadratnoj matrici. U prvom redu datoteke data je njena dimenzija, a potom slede elementi matrice. Napisati program koji alocira memorijski prostor za matricu i potom je učitava. Ispisati indekse onih redova matrice u kojima su elementi sortirani neopadajuće (redovi se indeksiraju počevši od nule). U slučaju greške ispisati* `-1` *i prekinuti izvršavanje programa.*

```
Primer 1:          Primer 2:          Primer 3:          Primer 4:
matrica.txt:       matrica.txt:       matrica.txt:       matrica.txt:
2                  3                  -5                 3
-1 3               1 3 2                                 4 4 3
2 1                -5 2 2                                1 0 -3
                   2 4 5                                 5 4 3

0                  1 2                -1
```

Zadatak 17 *U datoteci* `slagalica.txt` *nalaze se podaci o učesnicima u "Slagalici": ime i broj osvojenih poena u svakoj od emisija. Može se desiti da se ime nekog učesnika više puta ponavlja. Na standardni izlaz ispisati imena učesnika sortirana u opadajućem poretku na osnovu ukupnog broja osvojenih poena. Ukupan broj učesnika može biti najviše* `50`*. U slučaju greške ispisati* `-1` *i prekinuti izvršavanje programa.*

```
Primer 1:          Primer 2:          Primer 3:          Primer 4:
slagalica.txt      slagalica.txt      slagalica.txt      slagalica.txt
Marko 50           Isidora 15         Marko 78
Pavle 98           Janko 23           Milan 12
Petar 12           Marko 45           Nebojsa 100
Marko 60                              Milan 15
Pavle 45                              Marko 10

Pavle              Marko              Nebojsa
Marko              Janko              Marko
Petar              Isidora            Milan
```

Zadatak 18 *Sa standardnog ulaza unosi se niska koja sadrži ispravan datum u formatu DD.MM.GGGG (dan i mesec su zapisani sa 2 cifre, a godina sa 4). Napisati funkciju* `unsigned f(char *date)` *koja na osnovu prosledjenog (obavezno ispravnog) datuma formira neoznačen ceo tako što mu 5 bitova najmanje težine postavlja na vrednost dana (broj od 1 do 31), sledećih 4 bita na vrednost meseca (broj od 1 do 12), a u ostale bitove smešta vrednost godine.*

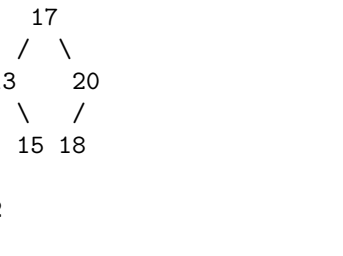
```
Primer 1:          Primer 2:          Primer 3:
03.06.2014         31.12.2000.        01.01.0001.

1031363            1024415            545
```

Zadatak 19 *Lista se učitava sa standardnog ulaza. Elementi liste su celi brojevi i učitavaju se sve dok se ne unese* `0` *i smeštaju se na kraj liste (koristiti datu funkciju* `cvor* ucitaj_listu()`*). Napisati funkciju koja iz date liste briše sledbenika prvog elementa u listi sa zadatom vrednošću, a ukoliko takav element ne postoji u listi, onda se briše prvi član liste. Dobijenu listu ispisati na standardni izlaz. NAPOMENA: Zadatak se mora rešiti korišćenjem listi, u suprotnom broj osvojenih poena je* `0`.

```
Primer 1:           Primer 2:           Primer 3:         Primer 4:
1 2 8 3 8 4 0       6 2 -2 3 4 0        0                 2 6 3 4 5 6 0
8                   4                   3                 7

1 2 8 8 4           6 2 -2 3 4                            6 3 4 5 6
```

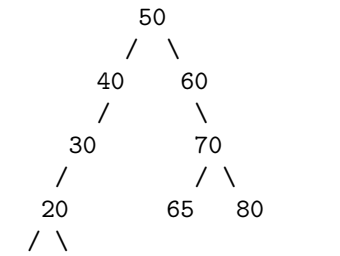
Zadatak 20 *Binarno uredjeno stablo se učitava sa standardnog ulaza. Elementi stabla su celi brojevi i učitavaju se sve dok se ne unese* `0`*. Nakon toga se unosi celobrojna vrednost nivoa stabla* `n`*. Napisati funkciju koja izračunava koliko se čvorova datog binarnog stabla nalazi na* `n`*-tom nivou stabla (koren se nalazi na nultom nivou, njegova deca na prvom nivou i tako redom). Rezultat ispisati na standardni izlaz. NAPOMENA: Zadatak se mora rešiti korišćenjem stabla, u suprotnom broj osvojenih poena je* `0`.

Primer 1:



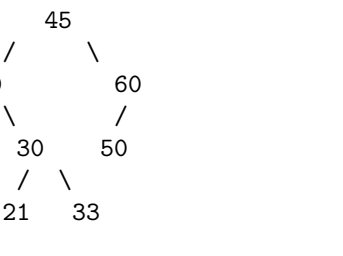

```
ulaz:17 13 20 18 5 15 12 0
     2

izlaz: 3
```

Primer 2:



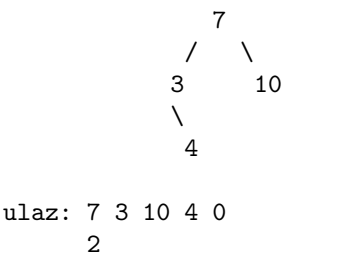

```
ulaz: 50 40 30 20 10 25 60 70 65 80 0
3

izlaz: 3
```

Primer 3:




```
ulaz: 45 20 30 33 21 15 18 60 50 0
      4
izlaz: 0
```

Primer 4:




```
ulaz: 7 3 10 4 0
      2
izlaz: 1
```

9.5 I smer, Programiranje 2 2013/2014, završni ispit, septembar 2014

NAPOMENA: Na desktopu napraviti direktorijum sa imenom `inicijaliAsistenta_ImePrezime_brojIndeksa_1`. U tom direktorijumu čuvati zadatke – 1.c, 2.c, 3.c, 4.c, 5.c, 6.c

U direktorijumu `g1` nalaze se funkcije za rad sa listama (`liste.c` i `liste.h`) i funkcije za rad sa stablima (`stabla.c` i `stabla.h`).
Kompilacija: `gcc 5.c liste.c`
`gcc 6.c stabla.c`

Zadatak 21 *Prvi argument komadne linije je pozitivan ceo broj* `k`*, a ostali argumenti su reči (nije unapred poznato koliko maksimalno reči može biti). Na standarni izlaz ispisati svaku* `k`*-tu reč. Ukoliko nisu zadati argumenti komandne linije, na standarni izlaz ne ispisivati ništa.*
U slučaju greške (`k` $\leq$ `0`*) ispisati* `-1`*.*

```
Primer 1:                                    Primer 2:
./a.out 2 danas je ispit iz p2               ./a.out 3 sunce jutro oktobar festival nemacka pas auto

je iz                                         oktobar auto
```

```
Primer 3:                                   Primer 4:
./a.out -3 autobus prevoz gradski           ./a.out 4 jabuka sljiva banana pomorandza cvekla sargarepa

-1                                          pomorandza tikvice
```

Zadatak 22 *U datoteci* `pravougaonici.txt` *dati su podaci o pravougaonicima – ime (ne duže od 10 karaktera), koordinate donjeg levog temena, koordinate gornjeg desnog temena (strane pravougaonika su paralelne koordinatnim osama, koordinate su dva broja, vrednosti na x-osi i y-osi). Sortirati površine pravouganika opadajući prema njihovoj površini. Na standarni izlaz ispisati imena pravougaonika prema sortiranom rasporedu. Maksimalno može biti 1000 pravougaonika. U slučaju greške ispisati* `-1` *i prekinuti izvršavanje programa.*

Zadatak 23 *U svakom redu datoteke* `saldo.txt` *nalazi se identifikacija (niska maksimalne dužine 20) korisnika banke i iznos novca koji korisnik trenutno ima (ceo broj). Svaki korisnik se pojavljuje tačno jednom i njegov saldo je predstavljen celim brojem (negativan - korisnik je zadužen, pozitivan - korisnik ima pozitivan saldo i nije zadužen). Izračunati koliko su prosečno zaduženi korisnici (pri računanju proseka ne računati one korisnike koji nisu zaduženi). Na standarni izlaz ispisati identifikacioni broj onog korisnika koji su zaduženi više od proseka. Maksimalan broj korisnika nije unapred poznat.*

Zadatak 24 *Napisati funkciju* `unsigned int f2(unsigned int x)` *koja vraća broj dobijen postavljanjem neparnih bitova u broju* `x` *na 0. Parni bitovi ostaju nepromenjeni. Bitovi se broje počev od 1, od bita najmanje težine ka bitu najveće težine (sa desna na levo).*

```
Primer 1:          Primer 2:          Primer 3:          Primer 4:
ulaz: 23           ulaz:  3456        ulaz:  456         ulaz: 778903

izlaz: 2           izlaz: 2176        izlaz: 136         izlaz: 696962
```

Zadatak 25 *Lista se učitava sa standardnog ulaza. Elementi liste su celi brojevi i učitavaju se sve dok se ne unese* `0` *i smeštaju se na kraj liste (koristiti datu funkciju* `cvor* ucitaj_listu()`*). Nakon unosa elemenata liste, unosi se ceo broj* `k`*. Iz date liste izbaciti sve elemente koji su deljivi sa* `k`*. Dobijenu listu ispisati na standardni izlaz. U slučaju greške (pokušaj deljenja sa* `0`*) ispisati -1. NAPOMENA: Zadatak se mora rešiti korišćenjem listi, u suprotnom broj osvojenih poena je* `0`*.*

```
Primer 1:            Primer 2:                Primer 3:                Primer 4:          Primer 5
2 3 6 7 8 91 0       5 7 9 20 24 56 78 0      24 3 7 93 12 4 11 0      9 7 5 3 1 0        6 8 10 0
3                    5                        0                        100                2

2 7 8                7 9 24 56 78             -1                       9 7 5 3 1
```

Zadatak 26 *Binarno uredjeno stablo se učitava sa standarnog ulaza. Elementi stabla su celi brojevi i učitavaju se sve dok se ne unese* `0`*. Odrediti broj čvorova kod kojih je levo podstablo (strogo) dublje od desnog podstabla. NAPOMENA: Zadatak se mora rešiti korišćenjem stabla, u suprotnom broj osvojenih poena je* `0`*.*

```
Primer 1:
          17
         /  \
        13    20
       /  \   /
      5   15 18
       \
       12

ulaz:17 13 20 18 5 15 12 0

izlaz:  3
```

```
Primer 2:
          50
         /  \
        40    60
       /      \
      30       70
      /          \
     20          80
    /              \
   10               90
  /                   \
 5                     95

ulaz: 50 40 30 20 10 5 60 70 80 90 95 0

izlaz: 4
```

```
Primer 3:
            45
          /    \
        20       60
       /  \      /
     15    30   50
      \    /  \
      18  21   33

ulaz: 45 20 30 33 21 15 18 60 50 0

izlaz: 2
```

```
Primer 4:
          7
         /  \
        3    10
        \      \
         4      12

ulaz: 7 3 10 4 12 0

izlaz: 0
```

9.6 Programiranje 2, Završni ispit, januar 2015.

Zadatak 27 *Argumenti komandne linije su celi, pozitivni brojevi. Napisati program koji ispisuje broj elemenata komandne liniji čije su cifre uredjene strogo rastuće.*

```
Primer 1:               Primer 2:              Primer 3:             Primer 4:
./a.out 26 13 468       ./a.out 2 14 41        ./a.out               ./a.out 423 189 243 117 258

3                       2                      0                     2
```

Zadatak 28 *U svakom redu datoteke* `kurs.txt` *dat je datum, u formatu dan mesec godina (celi brojevi odvojeni blanko znakom), a zatim i kurs eura (realan broj) na taj datum. Napisati program koji ispisuje mesec i godinu u kojoj je prosečan kurs eura bio najniži. U slučaju greške ispisati* `-1`*.*

```
Primer 1:               Primer 2:              Primer 3:             Primer 4:
kurs.txt:               kurs.txt:              kurs.txt:             kurs.txt:
12 3 2014 112.34        12 4 2013 102.23                             2 3 2011 110.34
18 3 2014 115.54                                                     18 3 2011 110.54
1 5 2012 108.12         4 2013                                       12 2 2011 107.1
2 2 2014 109.12                                                      15 2 2011 108.1
25 5 2012 105.12                                                     13 2 2012 110
                                                                     15 2 2012 115

5 2015                                                               2 2011
```

Zadatak 29 *Napisati program koji ispisuje broj jedinica koje se nalaze izmedju dve nule u binarnom zapisu neoznačenog celog broja koji se unosi sa standardnog ulaza.*

```
Primer 1:               Primer 2:              Primer 3:             Primer 4:
1024                    320                    2015                  42

1                       2                      0                     2
```

Zadatak 30 *Napisati program koji sa standardnog ulaza učitava indekse studenata, njihova imena i prezimena (svaki student u jednom redu, ne vise od 128 redova), a potom iste ispisuje na standardni izlaz, sortirane rastuće po godini, pa po broju indeksa. Koristiti funkciju standardne biblioteke* `qsort`*. Predpostavka je da su svi redovi zadati u ispravnom formatu.*

```
Primer 1:                               Primer 2:
23/2014 Marko Markovic                  56/1999 Marko Maric
1/2014 Pera Peric                       223/2015 Pera Peric
234/2011 Branko Brankovic               224/2015 Mira Miric
123/2012 Branko Brankovic

                                        56/1999 Marko Maric
234/2011 Branko Brankovic               223/2015 Pera Peric
123/2012 Branko Brankovic               224/2015 Mira Miric
1/2014 Pera Peric
23/2014 Marko Markovic
```

```
Primer 3:                               Primer 4:
Ulaz:                                   Ulaz:
                                        12/2010 Jelena Jovanovic

Izlaz:                                  Izlaz:
                                        12/2010 Jelena Jovanovic
```

Zadatak 31 *Sa standarnog ulaza se unosi lista celih brojeva dok se ne unese 0. Formirati listu, a potom izbaciti sve one elemente koji su jednaki zbiru svojih suseda i ispisati novodobijenu listu. Susedi čvoru su*

njegov prethodnik i njegov sledbenik. Sused prvom elementu liste je njegov sledbenik. Sused poslednjem elementu liste je njegov prethodnik. Ne kreirati novu listu. Smatra se netačnim rešenje u kome se elementi liste samo ispisuju, a lista se pri tome ne menja. NAPOMENA: Zadatak se mora rešiti korišćenjem liste, u suprotnom broj osvojenih poena je `0`.

```
Primer 1:               Primer 2:                 Primer 3:               Primer 4:
7 8 1 3 2 1 0           23 -4 -27 -23 0           1 2 3 4 5 0             7 7 7 0

7 1 2 1 1               23 -23                    1 2 3 4 5               7
```

Zadatak 32 *Uredjeno binarno stablo se učitava sa standarnog ulaza. Elementi stabla su celi brojevi i učitavaju se sve dok se ne unese* `0`. *Odrediti broj čvorova uredjenog binarnog stabla koji su jednaki zbiru svojih suseda. Smatramo da su susedi nekom čvoru njegovi direktni potomci i njegov prvi predak. Koren nema pretka, pa su njegovi susedi samo njegovi direktni potomci. Čvor koji je list nema potomke, pa je njegov sused samo njegov direktan predak. Rezultat ispisati na standardni izlaz. NAPOMENA: Zadatak se mora rešiti korišćenjem stabla, u suprotnom broj osvojenih poena je* `0`.

```
Pored cvora koji zadovoljava trazeno svojstvo stavljena je zvezdica.

Primer 1:
          17*
         /  \
       -4*   21
      /  \   /
    -20   -1 21*
      \
      -8

ulaz:17 -4 -20 -1 -8 21 21 0

izlaz: 3
```

```
Primer 2:
        50
       /  \
     40*  60
    /      \
  -10       70
  /  \       \
-20   20      80
 /              \
-30             90
 /                \
-45                95

ulaz: 50 40 -10 20 -20 -30 -45 60 70 80 90 95 0

izlaz: 1
```

```
Primer 3:
          45*
        /     \
      -20*     65
     /  \      /
   -50* -15*  65*
    \     /  \
    -30  -8  13

ulaz: 45 -20 -50 -15 -30 -8 13 65 65 0

izlaz: 5
```

```
Primer 4:
        7*
       /  \
     -3    10
       \   /
        4  10*

ulaz: 7 -3 4 10 10 0

izlaz: 2
```

Glava 10

Testovi i ispiti 2014/15

10.1 Programiranje 2, I smer, 2014/15, prvi prakticni test

10.1.1 Grupa 1

1. Sa standardnog ulaza se učitava ceo broj x, ceo broj n ($n \leq 100$), a potom i niz od n celih brojeva. Napisati rekurzivnu funkciju *void f(int a[], int n, int x)*, koja u nizu a poslavlja na nulu sve parove susednih elementa čiji je zbir (u početnom nizu) jednak x. Rezultujući niz ispisati na standardni izlaz.

Ulaz	3 9 1 2 7 6 0 3 2 1 4	2 2 1 1	5 5 8 2 3 2 9	7 8 1 5 2 6 3 4 3 9
Izlaz	0 0 7 6 0 0 0 0 4	0 0	8 0 0 0 9	1 0 0 6 0 0 0 9

2. Sa standardnog ulaza se učitava ceo broj n ($n \leq 100$), a potom i niz od n celih brojeva, uredjenih rastuće. Binarnom pretragom pronaći indeks prvog elementa koji je veći od prosečne vrednosti u nizu (element sa najmanjim indeksom koji ispunjava dato svojstvo). Ispisati dobijeni indeks na standardni izlaz (u slučaju da takav broj ne postoji ne pisati ništa). U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	10 5 10 12 20 100 101 102 596 703 1001	8 1 2 8 10 65 102 104 500	0	3 100 100 100
Izlaz	7	5		

3. U datoteci "kompleksni.txtše nalaze kompleksni brojevi (njihov tačan broj nije unapred poznat). Svaki kompleksan broj je zadat sa dva broja tipa *float*. Sortirati kompleksne

   brojeve nerastuće prema veličini modula. Dobijeni sortiran niz upisati u datoteku šortirani_kompleksni.txt". Maksimalan

   broj kompleksnih brojeva je 1000. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	kompleksni.txt: 19.89 3.56 6.87 19.05 10.21 11.32 10.23 2.78 6.04 17.19 19.55 15.53	kompleksni.txt: 4.22 12.63 14.49 18.18 5.76 6.91 15.39 4.44		kompleksni.txt:
Izlaz	sortirani_kompleksni.txt: 19.55 15.53 6.87 19.05 19.89 3.56 6.04 17.19 10.21 11.32 10.23 2.78	sortirani_kompleksni.txt: 14.49 18.18 15.39 4.44 4.22 12.63 5.76 6.91	-1	sortirani_kompleksni.txt:

10.1.2 Grupa 2

1. Sa standardnog ulaza se učitava ceo broj n ($n \leq 100$), a potom i niz od n celih brojeva. Napisati rekurzivnu funkciju *void f(int a[], int n)* koja u nizu a postavlja na nulu sve elemente koji su (u početnom nizu) jednaki zbiru svojih suseda. Rezultujući niz ispisati na standardni izlaz.

Ulaz	6 1 4 3 6 3 3	5 1 0 3 3 0	2 1 1	6 1 2 1 1 3 2
Izlaz	1 0 3 0 3 3	1 0 0 0 0	1	1 0 1 1 0 2

2. Sa standardnog ulaza se učitava ceo broj k, ceo broj n ($n \leq 100$), a potom i niz od n celih brojeva, uređenih rastuće. Napisati funkciju koja binarnom pretragom nalazi indeks prvog k-tocifrenog elementa (element sa najmanjim indeksom koji ispunjava dato svojstvo). Ispisati dobijeni indeks na standardni izlaz (u slučaju da takav broj ne postoji ne pisati ništa). U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	4 10 5 10 12 20 100 101 102 596 703 1001	3 8 1 2 8 10 65 102 104 500	5 0	4 3 100 100 100
Izlaz	9	5		

3. Datoteka "artikli.txtšadrži informacije o artiklima.

   Format datoteke je takav da je najpre dat broj artikala, a potom u svakom

   sledećem redu su date informacije o artiklu: *naziv* (najviše 20 karaktera), *cena* (ceo broj), *komada* (ceo broj).

   Artikala nikad nema više od 200. Učitati datoteku u niz struktura ARTIKAL,

   a potom sortirati niz prema ukupnoj vrednosti artikala ($vrednost = cena \cdot komada$) u opadajućem

   poretku i ispisati ga na standardni izlaz. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	artikli.txt: 3 frizider 23450 2 pegla 4500 15 usisivac 7000 4	artikli.txt: 5 a1 162 130 a2 160 136 a3 172 182 a4 173 183 a5 101 116		artikli.txt: 0
Izlaz	pegla 4500 15 frizider 23450 2 usisivac 7000 4	a4 173 183 a3 172 182 a2 160 136 a1 162 130 a5 101 116	-1	

10.1.3 Grupa 4

1. Ceo broj x se učitava sa standardnog ulaza. Napisati rekurzivnu funkciju *int f(int x)* koja u datom broju x uklanja sve cifre koje su (u početnom broju) jednake zbiru svojih suseda. Rezultat funkcije ispisati na standardni izlaz.

Ulaz	14363	11	13216	10330
Izlaz	133	11	1216	100

2. Sa standardnog ulaza se učitava ceo broj x, ceo broj n ($n \leq 100$), a potom i niz od n celih brojeva, uredjenih rastuće. Napisati funkciju koja u rastuće uredjenom nizu celih brojeva pronalazi broj koji je najbliži datom broju x. Ukoliko ima više takvih brojeva pronalazi onaj sa najmanjim indeksom. Funkcija vraća vrednost

   pronadjenog broja i treba da radi u vremenu $O(log(n))$. Ispisati dobijeni broj na standardni izlaz (u slučaju da takav broj ne postoji ispisati 0). U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	11 10 5 10 12 20 100 101 102 596 703 1001	200 8 1 2 8 10 65 102 104 500	5 0	54 3 100 100 100
Izlaz	10	104	0	100

3. U datoteci čije se ime zadaje kao prvi argument komandne linije se nalaze pozitivni razlomci, u svakom redu po jedan, ne više od 256 (njihov tačan broj nije unapred poznat). Jedan razlomak je zadat kao par brojeva tipa *float*.

   Koristeći qsort, sortirati ih rastuće, i tako sortirane ih upisati u datoteku čije se ime zadaje kao drugi argument komenandne linije. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	a.out u.txt i.txt u.txt: 14.49 18.18 15.39 4.44 4.22 12.63 5.76 6.91	a.out u.txt i.txt u.txt: 1 2 3 4 5 0 6 7	a.out u.txt i.txt	a.out u.txt i.txt u.txt:
Izlaz	i.txt: 4.22 12.63 14.49 18.18 5.76 6.91 15.39 4.44	-1	-1	sortirani_kompleksni.txt:

10.1.4 Grupa 3

1. Sa standardnog ulaza se učitava ceo broj n ($n \leq 100$), a potom i niz od n celih brojeva. Napisati rekurzivnu funkciju *int f(int a[], int n)* koja računa proizvod svih neparnih brojeva prosledjenog celobrojnog niza. Rezultat funkcije ispisati na standardni izlaz.

Ulaz	6 1 4 3 6 3 3	4 2 4 6 8	3 5 0 2	0
Izlaz	27	1	5	1

2. Kao argumenti komandne linije zadaju se dva broja tipa *float* a i b ($a \leq b$). Sa standarnog ulaza se unosi 11 brojeva tipa *float* (redom $a_0, a_1, a_2, ..., a_{10}$) koji predstavljaju koeficijente polinoma $a_{10}x^{10} + a_9x^9 + a_8x^8 + ... + a_0$. Napisati funkciju koja traži nulu polinoma na intervalu $[a, b]$. Pretpostaviti da će na intervalu $[a, b]$ uvek postojati tačno jedna nula funkcije, i da su vrednosti polinoma u tačkama a i b različitog znaka. Koristiti metod polovljenja intervala. Rezultat ispisati na standardni izlaz zaokružen na dve decimale. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	a.out 0 2 -1 0 1 0 0 0 0 0 0 0 0	a.out -7.5 1.3 1 0 -2 3.2 0 0 0 0 0 0 0	a.out -1 0 -4 0 5 -0.2 0 0 0 0 0 0 0	a.out 20 30 -4 0 5 -0.2 0 0 0 0 0 0 0
Izlaz	1.00	-0.52	-0.88	24.97

3. U prvom redu datoteke "proizvodi.txt" dat je broj n ($n \leq 1000$), a zatim u svakom od narednih rarednih n redova naziv proizvoda i količina. Proizvod sa istim nazivom se može pojaviti više puta u više različitih redova u datoteci. Potrebno je učitati sve proizvode u niz (bez ponavljanja proizvoda), u kome će se uz svaki proizvod čuvati njegova ukupna količina pročitana iz datoteke. Niz sortirati na osnovu ukupne količine rastuće i ispisati ga na standarndi izlaz. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	proizvodi.txt: 5 p1 20 p2 50 p1 40 p3 10 p2 5	proizvodi.txt: 8 p1 18 p2 2 p3 17 p4 12 p1 9 p2 9 p3 13 p4 14		proizvodi.txt: 0
Izlaz	p3 10 p2 55 p1 60	p2 11 p4 26 p1 27 p3 30	-1	

10.2 Programiranje 2, I smer, 2013/14, Drugi praktični test

10.2.1 Prva grupa

NAPOMENA: Na desktopu napraviti direktorijum sa imenom `inicijaliAsistenta_ImePrezime_brojIndeksa_1`. U tom direktorijumu čuvati zadatke – 1.c, 2.c, 3.c. U drugom i trećem zadatku korisiti pomoćne funkcije iz liste.[hc], a u fajlovima 2.c i 3.c napisati samo traženu i main funkciju.

Zadatak 1 *Napisati funkciju* `unsigned int f1(unsigned int x)` *koja u datom broju invertuje svaki treći bit. Prvi bit koji se invertuje je bit najmanje težine. Sa standardnog ulaza se unosi ceo pozitivan broj. Ispisati rezultat funkcije na standardni izlaz.*

```
Ulaz:     0           345         1024        1
Izlaz:    1227133513  1227133712  1227134537  1227133512
```

Zadatak 2 *Napisati funkciju* `int f2(cvor* lista)` *koja za elemente liste $a_1, a_2, ..., a_n$ računa $a_1 - a_2 + a_3 - ... + (-1)^{n+1} a_n$. Dozvoljeno je dodati još jedan argument u funkciju f2. Lista se učitava sa standardnog ulaza, sve dok se ne unese nula (koja se ne treba nalaziti u listi), elemenati se dodaju na kraj liste, a rezultat funkcije se ispisuje na standardni izlaz.*

```
Ulaz:     4 2 5 3 8 0     12 3 3 24 25 0     4 6 2 1 2 4 0    0
Izlaz:    12              13                 -3               0
```

Zadatak 3 *Napisati funkciju* `void f3(cvor* lista, int k)` *koja modifikuje zadatu listu tako što iza svakog broja deljivog sa* `k` *umeće -1. Lista se učitava sa standardnog ulaza, sve dok se ne unese nula, potom se učitava k, a izmenjenu listu ispisati na standardni izlaz.*

```
Ulaz:     4 2 5 3 8 0 2       12 3 3 24 25 0 3          4 6 2 1 2 4 0 1                0 5
Izlaz:    4 -1 2 -1 5 3 8 -1  12 -1 3 -1 3 -1 24 -1 25  4 -1 6 -1 2 -1 1 -1 2 -1 4 -1
```

10.2.2 Drugi praktični test - Druga grupa

NAPOMENA: Na desktopu napraviti direktorijum sa imenom `inicijaliAsistenta_ImePrezime_brojIndeksa_2`. U tom direktorijumu čuvati zadatke – 1.c, 2.c, 3.c. U drugom i trećem zadatku korisiti pomoćne funkcije iz liste.[hc], a u fajlovima 2.c i 3.c napisati samo traženu i main funkciju.

Zadatak 1 *Napisati funkciju* `unsigned int f1(unsigned int x, unsigned int k, unsigned int p)` *koja u datom broju invertuje prvih k i poslednjih p bitova. U slučaju da su k i p u zbiru dovoljno veliki, može se desiti da neki bitovi budu dva puta invertovani. Bitovi broja se čitaju sa desna na levo. Sa standardnog ulaza se unose celi pozitivni brojevi x, k i p. Ispisati rezultat funkcije na standardni izlaz.*

```
Ulaz:     0 2 3         23345 2 1         1024 1 4        1 3 2
Izlaz:    3758096387    2147506994        4026532865      3221225478
```

Zadatak 2 *Napisati funkciju* `int f2(cvor* lista, int k)` *koja vraća zbir elemenata u listi deljivih sa k. Lista se učitava sa standardnog ulaza, sve dok se ne unese nula (koja se ne treba nalaziti u listi), elemenati se dodaju na kraj liste, potom se učitava k, a rezultat funkcije se ispisuje na standardni izlaz.*

```
Ulaz:     4 2 5 3 8 0 2    12 3 3 24 25 0 3     4 6 2 1 2 4 0 5     0 8
Izlaz:    14               42                   0                   0
```

Zadatak 3 *Napisati funkciju* `void f3(cvor* lista)` *koja modifikuje zadatu listu tako što iza svakog broja umeće broj njegovih cifara. Lista se učitava sa standardnog ulaza, sve dok se ne unese nula, a izmenjenu listu ispisati na standardni izlaz.*

```
Ulaz:     4 2 5 3 8 0           -12 312 3 24 25 0             1024 0      0
Izlaz:    4 1 2 1 5 1 3 1 8 1   -12 2 312 3 3 1 24 2 25 2     1024 4
```

10.2.3 Drugi praktični test - Treća grupa

NAPOMENA: Na desktopu napraviti direktorijum sa imenom `inicijaliAsistenta_ImePrezime_brojIndeksa_3`. U tom direktorijumu čuvati zadatke – 1.c, 2.c, 3.c. U drugom i trećem zadatku korisiti pomoćne funkcije iz liste.[hc], a u fajlovima 2.c i 3.c napisati samo traženu i main funkciju.

Zadatak 1 *Napisati funkciju* `unsigned int f1(unsigned int x, unsigned int k)` *koja u datom broju invertuje svako k-to pojavljivanje jedinice. Bitovi broja se čitaju sa desna na levo. Sa standardnog ulaza se unosi ceo pozitivan broj x i k. Ispisati rezultat funkcije na standardni izlaz.*

```
Ulaz:     0 2         23345 2         1024 1         1 3
Izlaz:    0           4641            0              1
```

Zadatak 2 *Napisati funkciju* `int f2(cvor* lista1, cvor* lista2)` *koja vraća skalarni proizvod dve liste. Pretpostaviti da su liste iste dužine. Liste se učitavaju sa standardnog ulaza, sve dok se ne unese nula (koja se ne treba nalaziti u listi), elemenati se dodaju na kraj liste, a rezultat funkcije se ispisuje na standardni izlaz.*

```
Ulaz:     1 2 3 0 4 5 6 0    10 9 0 4 3 0      4 0 5 0     0 0
Izlaz:    32                 67                20          0
```

Zadatak 3 *Napisati funkciju* `void f3(cvor* lista)` *koja modifikuje zadatu listu tako što ukljanja svaki broj koji je veći od svog predhodnika. Lista se učitava sa standardnog ulaza, sve dok se ne unese nula, a izmenjenu listu ispisati na standardni izlaz.*

```
Ulaz:     4 2 5 3 8 0    12 3 3 24 25 0   4 6 2 1 2 4 3 0    3 0
Izlaz:    4 2 3          12 3 3           4 2 1 3           3
```

10.3 Programiranje 2, 2014/2015, I smer, završni ispit, jun 1

10.3.1 Grupa 1

NAPOMENA: Na desktopu napraviti direktorijum sa imenom `InicijaliAsistenta_ImePrezime_BrojIndeksa_1`. U tom direktorijumu čuvati zadatke – 1.c, 2.c, 3.c, 4.c, 5.c
U direktorijumu `g1` nalaze se funkcije za rad sa listama (`liste.c` i `liste.h`) i funkcije za rad sa stablima (`stabla.c` i `stabla.h`).

1. Kao argumenti komandne linije su zadata dva pravougaonika sa svoim dimenzijama, redom širinom i visinom: *s1 v1 s2 v2* (tipa *float*). Ispisati na standardni izlaz koliko najviše prvih pravougaonika može da stane u drugi, tako da su odgovarajuće stranice paralelne (svaka stranica koja označava širinu prvog pravougaoniku je paralelna sa stranicom koja označava širinu drugog pravougaonika). U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	./a.out 3.2 4.1 2.1 16.2	./a.out 2.1 3.2 9.8 9.1	./a.out 2 2 4 4	./a.out
Izlaz	0	8	4	-1

2. U datoteci *duzi.txt* se nalazi spisak duži zadat tačkama. Format datoteke je takav da je najpre zadat broj duži, a pitom u svakom narednom redu duž u vidu četiri koordinate: *Ax Ay Bx By* (tipa *float*). Potrebno je učitati duži iz datoteke, sortirati ih opadajuće prema njihovoj dužini i ispisati tako sortirani niz na standardni izlaz. U svakom redu se ispisuju *Ax Ay Bx By d*, gde je *d* dužina duži. Sve podatke ispisati zaokružene na dve decimale. Koristiti dinamičku alokaciju memorije. U slučaju greške ispisati -1 na standardni izlaz. Za koren broja tipa *float* koristiti funkciju *sqrtf*.

Ulaz	*duzi.txt*: 4 2.09 7.33 9.12 1.58 5.67 4.01 1.25 0.62 6.73 8.61 1.88 8.49 3.77 8.82 9.93 6.99	*duzi.txt*: 4 4 2 4 4 6 6 2 7 5 5 1 9 7 0 4 5	*duzi.txt*: 4 0 4 7 9 1 7 7 0 2 8 4 4 4 1 6 8	
Izlaz	2.09 7.33 9.12 1.58 9.08 3.77 8.82 9.93 6.99 6.43 5.67 4.01 1.25 0.62 5.57 6.73 8.61 1.88 8.49 4.85	7.00 0.00 4.00 5.00 5.83 5.00 5.00 1.00 9.00 5.66 6.00 6.00 2.00 7.00 4.12 4.00 2.00 4.00 4.00 2.00	1.00 7.00 7.00 0.00 9.22 0.00 4.00 7.00 9.00 8.60 4.00 1.00 6.00 8.00 7.28 2.00 8.00 4.00 4.00 4.47	-1

3. Sa standardnog ulaza se unosi ceo broj n ($n \leq 32$), a zatim i niz od n neoznačenih celih brojeva. Formirati neoznačeni ceo broj x tako što se na i-ti bit broja x postavlja vrednost i-tog bita i-tog broja niza. Broj x ispisati na standardni izlaz. Bitove broja čitati od pozicija manje težine ka pozicijama veće težine. Podrazumevana vrednost bitova broja x je 0. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	7 12 45 72 415 146 333 85	5 1024 64 128 31 511	5 127 0 0 63 128	41
Izlaz	88	24	9	-1

4. Napisati funkcuju koja briše svaki element liste koji je manji od sume svih prethodnih elemenata u listi. Prilikom računanja sume uzeti u obzir i prethodno obrisane elemente. Kreirati glavni program koji učitava listu, poziva funkciju *f4* i ispisuje dobijenu listu na izlaz. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	1 2 3 1 2 3 9 30 0	1 2 4 8 16 32 0	51 27 84 28 62 3 28 0	5 2 4 1 4 20 100 84 21 200 0
Izlaz	1 2 3 30	1 2 4 8 16 32	51 84	5 20 100

5. Sa standardnog ulaza se učitavaju dva stabla, $s1$ i $s2$. Ispitati da li $s1$ i $s2$ imaju istu strukturu (dva stabla imaju istu strukturu ako se jedno može u potpunosti preklopiti preko drugog i obrnuto). U slučaju da nemaju istu strukturu ispisati -1 na standardni izlaz. U slučaju da imaju istu strukturu

potrebno je izmeniti vrednosti u čvorovima stabla $s1$ tako što se vrednost svakog čvora stabla $s1$ uveća za vrednost odgovarajućeg čvora stabla $s2$. Izmenjeno stablo $s1$ ispisati na standardni izlaz. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	10 5 15 12 13 0 12 7 20 15 17 0	10 5 15 12 11 0 12 7 20 15 17 0	30 15 46 11 0 20 13 81 9 0	10 8 0 10 8 12 0
Izlaz	12 22 27 30 35	-1	20 28 50 127	-1

10.3.2 Grupa 2

NAPOMENA: Na desktopu napraviti direktorijum sa imenom `InicijaliAsistenta_ImePrezime_BrojIndeksa_2`. U tom direktorijumu čuvati zadatke – 1.c, 2.c, 3.c, 4.c, 5.c
U direktorijumu `g2` nalaze se funkcije za rad sa listama (`liste.c` i `liste.h`) i funkcije za rad sa stablima (`stabla.c` i `stabla.h`).

1. Kao argumenti komandne linije su zadate tri tačke sa svoim koordinatama: *x1 y1 x2 y2 x3 y3* (tipa *float*). Izračunati dužinu puta koji počinje u tački (0,0), prolazi kroz sve tri tačke redom i završava se u tački (0,0). Rezultat ispisati na standardni izlaz zaokružen na dve decimale. U slučaju greške ispisati -1 na standardni izlaz. Za koren broja tipa *float* koristiti funkciju *sqrtf*.

Ulaz	./a.out 1 0 1 1 0 1	./a.out 3 4 -2 1 2 8	./a.out 2 -1 4 3 -2 1	./a.out 1 2 3
Izlaz	4.00	27.14	15.27	-1

2. U datoteci *proizvodi.txt* se nalazi spisak proizvoda. Format datoteke je takav da je najpre zadat broj proizvoda, a zatim u svakom narednom redu naziv proizvoda (maksimalno 20 karaktera), cena i količina (tipa *float*). Potrebno je učitati proizvode iz datoteke, sortirati ih opadajuće prema ukupnoj vrednosti (*cena * količina*) i ispisati tako sortirani niz na standardni izlaz. U svakom redu se ispisuju naziv, cena, količina i ukupna vrednost. Sve podatke ispisati zaokružene na dve decimale. Koristiti dinamičku alokaciju memorije. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	*proizvodi.txt*: 4 p1 2.09 7.33 p2 5.67 4.01 p3 6.73 8.61 p4 3.77 8.82	*proizvodi.txt*: 4 p1 4 2 p2 6 6 p3 5 5 p4 7 0	*proizvodi.txt*: 4 p1 0 4 p2 1 7 p3 2 8 p4 4 1	
Izlaz	p3 6.73 8.61 57.95 p4 3.77 8.82 33.25 p2 5.67 4.01 22.74 p1 2.09 7.33 15.32	p2 6.00 6.00 36.00 p3 5.00 5.00 25.00 p1 4.00 2.00 8.00 p4 7.00 0.00 0.00	p3 2.00 8.00 16.00 p2 1.00 7.00 7.00 p4 4.00 1.00 4.00 p1 0.00 4.00 0.00	-1

3. Napisati program koji sa standardnog ulaza učitava neoznačen ceo broj x i tri broja i, j i k. U broju x razmeniti vrednosti dva bloka bitova dužine k, gde prvi blok počinje bitom na poziciji i, a drugi bitom na poziciji j. Dobijeni broj ispisati na standardni izlaz. Bitove broja čitati od pozicija manje težine ka pozicijama veće težine. U slučaju greške ispisati -1 na standardni izlaz. Greškom smatrati preklapanja blokova, kao i ako neki blok iskoči van granica neoznačenog celog broja.

Ulaz	1234567 0 4 3	341 2 17 5	2047 20 3 7	32567536 10 5 4
Izlaz	1234672	2752769	133170183	32562576

4. Napisati funkcuju *void f4(cvor *lista)* koja briše svaki element liste koji je manji od prethodnog elemenata u listi, a veći od sledećeg. Prilikom brisanja uzeti u obzir i prethodno obrisane elemente. Prvi i poslednji element se ne brišu. Kreirati glavni program koji učitava listu, poziva funkciju *f4* i ispisuje dobijenu listu na izlaz. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	1 4 2 6 3 1 4 2 1 0	5 4 3 2 1 0	8 1 3 2 7 4 3 1 4 6 2 1 0	0
Izlaz	1 4 2 6 1 4 1	5 1	8 1 3 2 7 1 4 6 1	

5. Napisati funkciju *cvor* f5(cvor* s)* koja za svaki čvor u stablu menja redosled njegovog levog i desnog direktnog potomka ukoliko levo podstablo ima veću dubinu od desnog podstabla. Ispisati dobijeno stablo na izlazu. Dubina predstavlja najduži put od korena do lista. Kreirati glavni program koji učitava stablo, poziva funkciju *f5* i ispisuje dobijeno stablo na izlaz. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	20 10 30 5 12 7 0	10 5 15 3 1 0	20 10 30 9 15 25 12 17 0	10 5 3 8 7 0
Izlaz	30 20 12 10 5 7	15 10 5 3 1	30 25 20 9 10 12 15 17	10 3 5 8 7

10.4 I smer, Programiranje 2 2014/2015, završni ispit, jun2 2015

Na *Desktop*-u napraviti direktorijum čije je ime u formatu **InicijaliAsistenta_ImeIPrezime_BrojIndeksa_1**. Na primer, **AZ_PeraPeric_mi14231_1**. Sve zadatke sačuvati u ovom direktorijumu. Zadatke imenovati sa **1.c**, **2.c**, **3.c**, **4.c** i **5.c**.

1. Kao argument komadne linije zadaje se jedna reč. Ispisati na standarni izlaz reč koja se dobije od zadate reči tako što se prvo slovo ponovi jednom, drugo dva puta, ..., n-to n puta. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	./a.out petar	./a.out 12345	./a.out p2ispit	./a.out
Izlaz	peetttaaaarrrrr	122333444455555	p22iiisssspppppiiiiiitttttttt	-1

2. U datoteci **polinomi.txt** se nalaze polinomi zadati svojim koeficijentima. Prvo se zadaje ukupan broj polinoma, a zatim u svakom narednom redu po jedan polinom. Svaki od tih redova sadrži ime polinoma (maksimalne dužine 20 karaktera), broj koeficijenata polinoma (ceo neoznačen broj n) i koeficijente (tipa *float*, ukupno n njih). Sortirati polinome opadajuće prema vrednosti u tački x koja je prosleđena programu kao argument komandne linije (tipa *float*). Ispisati imena i vrednosti sortiranih polinoma na standarni izlaz. Sve podatke ispisati zaokružene na dve decimale. Koristiti dinamičku alokaciju memorije. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	a.exe 1 *polinomi.txt*: 5 x^2+3x+5.1 3 5.1 3 1 x^3+3x+5 4 5 3 0 1 x^2-8 3 -8 0 1 8.5x 2 0 8.5 12x-4 2 -4 12	a.exe 3 *polinomi.txt*: 5 x^2+3x+5.1 3 5.1 3 1 x^3+3x+5 4 5 3 0 1 x^2-8 3 -8 0 1 8.5x 2 0 8.5 12x-4 2 -4 12	a.exe -3 *polinomi.txt*: 5 x^2+3x+5.1 3 5.1 3 1 x^3+3x+5 4 5 3 0 1 x^2-8 3 -8 0 1 8.5x 2 0 8.5 12x-4 2 -4 12	a.exe
Izlaz	x^2+3x+5.1 9.10 x^3+3x+5 9.00 8.5x 8.50 12x-4 8.00 x^2-8 -7.00	x^3+3x+5 41.00 12x-4 32.00 8.5x 25.50 x^2+3x+5.1 23.10 x^2-8 1.00	x^2+3x+5.1 5.10 x^2-8 1.00 8.5x -25.50 x^3+3x+5 -31.00 12x-4 -40.00	-1

3. Sa standarnog ulaza učitava se neoznačen ceo broj x, neoznačen ceo broj n i niz od n celih neoznačenih brojeva ($n \leq 32$). Odrediti neoznačen ceo broj y koji se dobija na sledeći način: porede se i-ti bit broja x i i-ti bit i-tog broja niza. Ukoliko su jednaki na i-to mesto broja y se postavlja bit 1, a inače se postavlja 0 (ukoliko i-ti broj niza ne postoji, podrazumevati da je vrednost odgovarajućeg bita 0). Ispisati broj y na standardni izlaz. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	1023 7 0 0 1023 1023 0 0 0	348712 4 1235 964914 24214 4212	12345 0	726431 2 4967 349672
Izlaz	4294966284	4294618576	4294954950	4294240865

4. Napisati funkciju *void f4(cvor* lista, int k)* koja u datoj listi između svaka dva elementa čiji su zbir ili razlika jednaki datom broju k umeće -1. Glavni program učitava listu i ceo broj k. Potrebno je ispisati

rezultujuću listu na stadardni izlaz. Nije dozvoljeno korišćenje pomoćne liste. U slučaju greške ispisati -1 na standardni izlaz. Ne analizirati prvi i poslednji element liste jer oni nemaju oba suseda.

Ulaz	1 2 3 1 2 3 0 3	4 2 1 5 6 2 4 0 2	1 3 1 1 3 1 0 2	0 5
Izlaz	1 -1 2 3 1 -1 2 3	4 -1 2 1 5 6 2 -1 4	1 -1 3 -1 1 -1 1 -1 3 -1 1	

5. Napisati funkciju *int f5(cvor* s, int k)* koja računa zbir svih parnih elemenata stabla s na nivou k, umanjen za zbir svih neparnih elemenata stabla s na nivou k. Glavni program učitava stablo i ceo broj k. Potrebno je ispisati rezultat funkcije *f5* na standardni izlaz. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	20 10 30 5 12 7 0 3	10 5 15 3 1 0 2	20 10 30 9 15 25 12 17 0 3	10 5 3 8 7 0 10
Izlaz	7	-20	-49	0

10.5 I smer, Programiranje 2 2014/2015, završni ispit, septembar 2015

Na *Desktop*-u napraviti direktorijum čije je ime u formatu **InicijaliAsistenta_ImePrezime_BrojIndeksa_1**. Na primer, **AZ_PeraPeric_mi14231_1**. Sve zadatke sačuvati u ovom direktorijumu. Zadatke imenovati sa **1.c**, **2.c**, **3.c**, **4.c** i **5.c**. U poslednja dva zadatka koristiti priložene biblioteke za rad sa listama (liste.[hc]) i stablima (stabla.[hc]) i kompilirati ih iz dve C datoteke.

1. Kao argument komadne linije zadaju se tri parametra – reč, slovo, broj. Izmeniti reč tako da se između prva dva pojavljivanja datog slova u reči svaki karakter uveća za dati broj. U slučaju greške ispisati `-1`.

```
Primer 1:              Primer 2:             Primer 3:             Primer 4:
./a.out danas a 3      ./a.out danas n 3     ./a.out oktobar 50    ./a.out proGramiraNjer r 5

daqas                  danas                 -1                    prtLramiraNjer
```

2. U datoteci `bioskop.txt` se nalaze podaci o filmovima koji se prikazuju u bioskopu. Ukupan broj filmova nije unapred poznat. Podaci su zapisani na sledeći način: `ime_filma` (jedna reč, ne duža od 50 kraktera), `vreme_prikazivanja` (vreme je oblika `HH:MM`). Pretpostaviti da su podaci u datoteci ispravno zadati. Sortirati podatke rastuće prema vremenu prikazivanja i na standarni izlaz ispisati imena filmova tako sortiranog niza. Potom ispisati onaj sat (`HH:00`) u kome ima najviše projekcija. U slučaju greške ispisati `-1`.

```
Primer 1:                          Primer 2:          Primer 3:          Primer 4:
Fantasticna_cetvorka 22:40         (nema datoteke)    (prazna datoteka)  Poseta 20:30
Noc_u_muzeju_2 14:00
Bekstvo_iz_Sosenka 20:20
Gradovi_na_papiru 19:00
Haos_u_najavi 19:20
Malci 19:10

Noc_u_muzeju_2                     -1                 -1                 Poseta
Gradovi_na_papiru                                                        20:00
Malci
Haos_u_najavi
Bekstvo_iz_Sosenka
Fantasticna_cetvorka
19:00
```

3. Napisati funkciju koja na osnovu neoznačenog broja x formira nisku s koja sadrži heksadekadni zapis broja x, koristeći algoritam za brzo prevođenje binarnog u heksadekadni zapis (svake 4 binarne cifre se zamenjuju jednom odgovarajućom heksadekadnom cifrom). Napisati program koji tu funkciju testira za broj koji se zadaje sa standardnog ulaza.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
11	1024	12345	123456789
0000000B	00000400	00003039	075BCD15

4. Napisati funkciju `cvor* f4(cvor* lista, int k)` koja u datoj listi izbacuje susedne elemente čiji zbir je jednak datom broju `k`. Potrebno je ispisati tako dobijenu listu na stadardni izlaz. Nije dozvoljeno korišćenje pomoćne liste. Nije dovoljno samo ispisati traženu listu već je potrebno elemente zaista izbaciti i konstruisati novu listu. Elementi liste su celi brojevi, lista se unosi sa standarnog ulaza sve dok se ne unese `0`. Nakon unosa elemenata liste unosi se broj `k`.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
13 4 5 10 0 9	13 4 5 9 9 0 9	13 4 5 4 3 0 9	4 5 3 -2 11 -2 11 -2 0 9
13 10	13 9 9	13 3	3

5. Napisati funkciju `int f5(cvor* stablo)` koja u datom stablu određuje broj onih elemenata kod kojih je zbir cifara svih elemenata levog podstabla strogo veći od zbira cifara svih elemenata desnog podstabla. Testirati funkciju pozivom u main-u. Stablo se unosi sa standardnog ulaza sve dok se ne unese 0. Elementi stabla su celi pozitivni brojevi.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
52 38 64 21 40 55 103 88 0	304 0	104 88 110 78 99 105 120 55 0	1111 -100 0
1	0	4	1

Glava 11

Testovi i ispiti 2015/2016

11.1 I smer, Programiranje 2 2015/2016, prvi praktični test

11.1.1 Grupa I

1. Kao argumenti komandne linije prosledjena su dva cela broja k i l ($2 \leq k \leq l \leq 10000$). Ispisati na standarni izlaz sve proste brojeve p takve da vazi $k \leq p \leq l$. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	./a.out 2 10	./a.out 3 6	./a.out 10 20	./a.out 52 10
Izlaz	2 3 5 7	3 5	11 13 17 19	-1

2. U datoteci "r.txt" nalazi se broj n, za kojim sledi n reci ($n \geq 0$, maksimalna dužina reči je 50 karaktera). U datoteku "b.txt" ispisati sve reči koje predstavljaju cele brojeve u dekadnom zapisu, a u datoteku "o.txt" sve one koje to nisu. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	r.txt: 5 baba deda 123 sd3f45 ff543g	r.txt: 10 1 a 22 b 333 c 4 d 5 ff	r.txt: 0	
Izlaz	b.txt: 123 o.txt: baba deda sd3f45 ff543g	b.txt: 1 22 333 4 5 o.txt: a b c d ff	b.txt: o.txt:	-1

3. Kreirati rekurzivnu funkciju *int f3(int x)* koja u početnom zapisu broja x izbacuje svaku neparnu cifru, ukoliko se ispred te cifre nalazi cifra 2. Kreirati program koji testira ovu funkciju, tako što sa standardnog ulaza učitava ceo broj x, i na standardni izlaz ispisuje vrednost funkcije *f3(x)*. U slučaju greške ispisati -1 na standardni izlaz.

   Napomena. Zadatak mora biti uradjen rekurzivno. Nije dozvoljeno korišćenje statičkih i globalnih promenljivih, menjanje zaglavlja funkcije i pisanje pomoćnih funkcija.

Ulaz	2325	23523	12345	0
Izlaz	22	252	1245	0

11.1.2 Grupa II

1. Kao argumenti komandne linije prosledjena su dva cela broja k i l ($0 \leq k \leq l \leq 10000$). Ispisati na standardni izlaz sumu cifara svih brojeva x, takvih da je $k \leq x \leq l$. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	./a.out 0 9	./a.out 10 13	./a.out 93 2145	./a.out
Izlaz	45	10	28722	-1

2. U datoteci "u1.txt" nalazi se broj n_1, za kojim sledi n_1 reči, dok se u datoteci "u2.txt" nalazi broj n_2, za kojim sledi n_2 reči ($n_1 \geq 0$, $n_2 \geq 0$, maksimalna dužina reči je 50 karaktera). U datoteku "i.txt" upisati naizmenično reči iz prve dve datoteke. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	u1.txt: 5 a b c d e u2.txt: 5 1 2 3 4 5	u1.txt: 2 a b u2.txt: 5 1 2 3 4 5	u1.txt: 2 abc 123 u2.txt: 0	
Izlaz	i.txt: a 1 b 2 c 3 d 4 e 5	i.txt: a 1 b 2 3 4 5	i.txt: abc 123	-1

3. Kreirati rekurzivnu funkciju *void f3(int *a, int na)* koja u početnom nizu a, duzine na, smanjuje vrednost broja za 1 ukoliko je on paran, i nakon njega se nalazi paran broj. Kreirati program koji testira ovu funkciju, tako što sa standardnog ulaza učitava broj na ($0 \leq na \leq 1000$), zatim vrednosti niza a, i na standardni izlaz ispisuje vrednosti izmenjenog niza.

   Napomena. Zadatak mora biti uradjen rekurzivno. Nije dozvoljeno korišćenje statičkih i globalnih promenljivih, menjanje zaglavlja funkcije i pisanje pomoćnih funkcija.

Ulaz	10 2 2 5 7 6 8 4 3 2 1	5 1 2 2 2 1	10 4 4 6 6 6 6 8 8 10 10	-1
Izlaz	1 2 5 7 5 7 4 3 2 1	1 1 1 2 1	3 3 5 5 5 5 7 7 9 10	-1

11.1.3 Grupa III

1. Preko argumenata komandne linije prosledjen je niz celih brojeva. Neka su a, b, c redom minimum, maksimum i prosečna vrednost niza. Ispisati brojeve niza strogo veće od (a + b + c)/3. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	./a.out 1 2 3 4 5	./a.out 4 11 10 5 6 15 8 9 19 8	./a.out 5 4 6 6 5 17 11 10 6 2	./a.out
Izlaz	4 5	11 15 19	17 11 10	-1

2. U datoteci "dat.txt" nalazi se reč s, zatim broj n i n reci ($n \geq 0$, maksimalna dužina reči je 50 karaktera). Napisati program koji na standardni izlaz ispisuje sve reči kojima je reč s sufiks. U slučaju greške ispisati -1 na standardni izlaz.

Ulaz	dat.txt: ab 6 vjab ab a abcd feb egaab	dat.txt: aa 7 a ab ba baa baaa b cba	dat.txt: abc 0	
Izlaz	vjab ab egaab	baa baaa		-1

3. Kreirati rekurzivnu funkciju *void f3(int *a, int na, int suma_prethodnih)* koja u u nizu a, dužine na, postavlja vrednost svakog broja na 0 ukoliko je veći od sume prethodnih brojeva u nizu. Kreirati program koji testira ovu funkciju, tako što sa standardnog ulaza učitava broj na, zatim vrednosti niza a, i na standardni izlaz ispisuje izmenjeni niz. U slučaju greške ispisati -1 na standardni izlaz.

   Napomena. Zadatak mora biti uradjen rekurzivno. Nije dozvoljeno korišćenje statičkih i globalnih promenljivih, menjanje zaglavlja funkcije i pisanje pomoćnih funkcija.

Ulaz	10 2 2 5 7 6 25 4 3 2 1	5 1 2 4 12 9	2 1 1	-1
Izlaz	0 2 0 7 6 0 4 3 2 1	0 0 0 0 9	0 1	-1