Univerzitet u Beogradu
Matematički fakultet

Milena, Jelena, Ana, Mirko, Anđelka, Nina

Zbirka programa

Beograd, 2015.

Predgovor

U okviru kursa *Programiranje 2* na Matematičkom fakultetu vežbaju se zadaci koji imaju za cilj da studente nauče rekurzivnom pristupu rešavanju problema, ispravan rad sa pokazivačima i dinamički alociranom memorijom, osnovne algoritme pretraživanja i sortiranja, kao i rad sa dinamičkim strukturama podataka, poput listi i stabala. Zadaci koji se nalaze u ovoj zbirci predstavljaju objedinjen skup zadataka sa vežbi i praktikuma ovog kursa, kao i primere zadataka sa kolokvijuma i ispita.

Autori velikog broja zadataka ove zbirke su ujedno i autori same zbirke, ali postoje i zadaci za koje se ne može tačno utvrditi ko je originalni autor jer su zadacima davali svoje doprinose različiti asistenti koji su držali vežbe iz ovog kursa u prethodnih desetak godina, pomenimo tu, pre svega, Milana Bankovića i doc dr Filipa Marića. Zbog toga, smatramo da je naš osnovni doprinos što smo objedinili, precizno formulisali i rešili sve najvažnije zadatke koji su potrebni za uspešno savlađivanje koncepata koji se obrađuju u okviru kursa.

Zahvaljujemo se recenzentima na ..., kao i studentima koji su svojim aktivnim učešćem u nastavi pomogli i doprineli u obličavanju ovog materijala.

Sadržaj

Glava 1

Rekurzija

1.1 Zadaci

Zadatak 1 Napisati program Hello.

Zadatak 2 Napisati program Hello.

Zadatak 3 Napisati program Hello.

Zadatak 4 Napisati program Hello.

Zadatak 5 Napisati program Hello.

1.2 Rešenja

Rešenje 1

```c
#include<stdio.h>

int main(){
  printf("Hello!\n");
  return 0;
}
```

Rešenje 2

```c
#include<stdio.h>

int main(){
  printf("Hello!\n");
  return 0;
}
```

Rešenje 3

```c
#include<stdio.h>

int main(){
  printf("Hello!\n");
  return 0;
}
```

Rešenje 4

```c
#include<stdio.h>

int main(){
  printf("Hello!\n");
  return 0;
}
```

Rešenje 5

```c
#include<stdio.h>

int main(){
  printf("Hello!\n");
  return 0;
}
```

Glava 2

Bitovi

2.1 Zadaci

Zadatak 6 Napisati funkciju `int Broj01(unsigned int n)` koja za dati broj `n` vraća `1` ako u njegovom binarnom zapisu ima više jednica nego nula, a inače vraća `0`. Napisati program koji tu funkciju testira za broj koji se zadaje sa standardnog ulaza.

Test 1

```
Ulaz:   10
Izlaz:  0
```

Test 2

```
Ulaz:   1024
Izlaz:  0
```

Test 3

```
Ulaz:   2147377146
Izlaz:  1
```

Test 4

```
Ulaz:   1111111115
Izlaz:  0
```

Zadatak 7 Napisati program Hello bitovi.

Zadatak 8 Napisati program Hello bitovi.

2.2 Rešenja

Rešenje 6

```c
#include<stdio.h>

int main(){
  printf("Hello bitovi!\n"); /* Da li komentari rade čćžš*/
  return 0;
}
```

Rešenje 7

```c
#include<stdio.h>

int main(){
  printf("Hello bitovi!\n"); /* Da li komentari rade čćžš*/
  return 0;
}
```

Rešenje 8

```c
#include<stdio.h>

int main(){
  printf("Hello bitovi!\n"); /* Da li komentari rade čćžš*/
  return 0;
}
```

Glava 3

Pokazivači

3.1 Pokazivači i aritmetika sa pokazivačima

Zadatak 9 Za dati celobrojni niz dimenzije n, napisati funkcije koje obrću njegove elemente:

(a) korišćenjem indeksne sintakse,

(b) korišćenjem pokazivačke sintakse.

Napisati program koji testira napisane funkcije. Sa standardnog ulaza učitati dimenziju niza n ($n \leq 100$), a zatim elemente niza. Prikazati sadržaj niza pre i posle poziva funkcije za obrtanje elemenata niza.

Zadatak 10 Dat je celobrojni niz dimenzije n.

(a) Napisati funkciju `zbir` koja izračunava zbir elemenata niza.

(b) Napisati funkciju `proizvod` koja izračunava proizvod elemenata niza.

(c) Napisati funkciju `min_element` koja izračunava najmanji elemenat niza.

(d) Napisati funkciju `max_element` koja izračunava najveći elemenat niza.

Napisati program koji testira napisane funkcije. Sa standardnog ulaza učitati dimenziju n ($0 < n \leq 100$) celobrojong niza, a zatim i elemente niza. Na standardni izlaz ispisati rezultat primene svake od napisanih funkcija nad učitanim nizom.

Zadatak 11 Korišćenjem pokazivačke sintakse, napisati funkciju koja vrednosti elemenata u prvoj polovini niza povećava za jedan, a u drugoj polovini smanjuje za jedan. Ukoliko niz ima neparan broj elemenata, onda vrednost srednjeg elementa niza ostaviti nepromenjenim. Napisati program koji testira napisanu funkciju. Sa standardnog ulaza učitati dimenziju n ($0 \leq n \leq 100$) celobrojong niza, a zatim i elemente niza. Na standardni izlaz ispisati rezultat primene svake od napisanih funkcija nad učitanim nizom.

Zadatak 12 Napisati program koji ispisuje broj prihvaćenih argumenata komandne linije, a zatim i same argumenate kojima prethode njihovi redni brojevi.

Zadatak 13 Korišćenjem pokazivačke sintakse, napisati funkciju koja za datu nisku ispituje da li je palindrom. Napisati program koji vrši prebrojavanje argumenata komandne linije koji su palindromi.

Test 1

```
Poziv:  ./a.out programiranje anavolimilovana topot ana anagram t
Izlaz:  4
```

Zadatak 14 Napisati program koji kao prvi argument komandne linije prihvata putanju do datoteke za koju treba proveriti koliko reči ima n karaktera, gde se n zadaje kao drugi argument komandne linije. U zadatku ne koristiti ugrađene funkcije za rad sa niskama, već implementirati svoje koristeći pokazivačku sintaksu.

Test 1

```
Poziv:    ./a.out fajl.txt 1
Datoteka: Ovo je sadrzaj datoteke i u
          njoj ima reci koje imaju
          1 karakter
Izlaz:    2
```

Zadatak 15 Napisati program koji kao prvi argument komandne linije prihvata putanju do datoteke za koju treba proveriti koliko reči ima zadati sufiks (ili prefiks), koji se zadaje kao drugi argument komandne linije. Program je neophodno pozvati sa jednom od opcija `-s` ili `-p` u zavisnosti od čega treba proveriti koliko reči ima zadati sufiks (ili prefiks). U zadatku ne koristiti ugrađene funkcije za rad sa niskama, već implementirati svoje koristeći pokazivačku sintaksu.

Test 1

```
Poziv:    ./a.out fajl.txt ke -s
Datoteka: Ovo je sadrzaj datoteke
          i u njoj ima reci koje se
          zavrsavaju na ke
Izlaz:    2
```

Zadatak 16 Milena: Ovo bi trebalo da ide u deo o rekurziji? Napisati rekurzivnu funkciju za određivanje najveće cifre u oktalnom zapisu neoznačenog celog broja korišćenjem bitskih operatora. Uputstvo: binarne cifre grupisati u podgrupe od po tri cifre, počev od bitova najmanje težine.

Test 1

```
Ulaz:  5
Izlaz: 5
```

Test 2

```
Ulaz:  125
Izlaz: 7
```

Test 3

```
Ulaz:  8
Izlaz: 1
```

Test 4

```
Ulaz:  10
Izlaz: 2
```

Zadatak 17 Milena: Ovo bi trebalo da ide u deo o rekurziji? Napisati rekurzivnu funkciju za određivanje (dekadne vrednosti) najveće cifre u heksadekadnom zapisu neoznačenog celog broja korišćenjem bitskih operatora. Uputstvo: binarne cifre grupisati u podgrupe od po četiri cifre, počev od bitova najmanje težine.

Test 1

```
Ulaz:  5
Izlaz: 5
```

Test 2

```
Ulaz:  16
Izlaz: 1
```

Test 3

```
Ulaz:  18
Izlaz: 2
```

Test 4

```
Ulaz:  165
Izlaz: 10
```

Zadatak 18 Milena: Ovo bi trebalo da ide u pretraživanje/sortiranje? Napisati funkciju koja u rastuće sortiranom nizu celih brojeva binarnom pretragom pronalazi prvi element veći od nule i kao rezultat vraća njegovu poziciju u nizu. Ukoliko nema elemenata većih od nule, funkcija kao rezultat vraća `-1`. Napisati program koji testira ovu funkciju za niz elemenata koji se učitavaju sa standardnog ulaza. Niz neće biti duži od 256, i njegovi elementi se unose sve do kraja ulaza.

Test 1

```
Ulaz:  -151 -44 5
       12 13 15
Izlaz: 2
```

Test 2

```
Ulaz:  -100 -15 -11
       -8 -7 -5
Izlaz: -1
```

Test 3

```
Ulaz:  -100 -15 0 13
       155 124 258
       315 516 7000
Izlaz: 3
```

Zadatak 19 Milena: Ovo bi trebalo da ide u pretraživanje/sortiranje? Napisati funkciju koja u opadajuće sortiranom nizu celih brojeva binarnom pretragom pronalazi prvi element manji od nule i kao rezultat vraća njegovu poziciju u nizu. Ukoliko nema elemenata manjih od nule, funkcija kao rezultat vraća `-1`. Napisati program koji testira ovu funkciju za niz elemenata koji se učitavaju sa standardnog ulaza. Niz neće biti duži od 256, i njegovi elementi se unose sve do kraja ulaza.

Test 1

```
Ulaz:  151 44 5 -12 -13 -15
Izlaz: 3
```

Test 2

```
Ulaz:  100 55 15 0 -15 -124 -155
       -258 -315 -516 -7000
Izlaz: 4
```

Test 3

```
Ulaz:   100 15 11 8 7 5 4 3 2
Izlaz: -1
```

Zadatak 20 Milena: Ovo bi trebalo da ide u pretraživanje/sortiranje? Struktura `Student` čuva podatke o broju indeksa studenta i poenima sa kolokvijuma, pri čemu su brojevi indeksa i poeni sa kolokvijuma celi brojevi. Napisati program koji učitava podatke o studentima iz datoteke „kolokvijum.txt“ u kojoj se nalazi najviše 500 zapisa o studentima. Sortirati ovaj niz studenata po broju poena opadajuće, a ako više studenata ima isti broj poena, onda po broju indeksa rastuće. Ispisati sortiran niz studenata na standardni izlaz.

Test 1

```
Kolokvijum.txt:   20140015 25
                  20140115 24
                  20130250 3
                  20140001 4
                  20140038 25
Izlaz:            20140015 25
                  20140038 25
                  20140115 24
                  20140001 4
                  20130250 3
```

Test 2

```
Kolokvijum.txt:   20140015 25
                  20110010 12
                  20140105 0
                  20120110 13
Izlaz:            20140015 25
                  20120110 13
                  20110010 12
                  20140105 0
```

Zadatak 21 Napisati strukturu `Student` koja čuva podatke o broju indeksa studenta i broju poena osvojenih na testu. Pretpostaviti da su brojevi indeksa celi brojevi, a poeni sa testa realni brojevi. Napisati program koji učitava podatke o studentima iz datoteke „studenti.txt“ u kojoj se nalazi najviše 100 zapisa o studentima. Sortirati ovaj niz studenata po broju poena rastuće, a ako više studenata ima isti broj poena, onda po broju indeksa opadajuće. Ispisati sortiran niz studenata na standardni izlaz.

Test 1

```
Kolokvijum.txt:    20140015 4.5
                   20130115 4.5
                   20140250 3.4
                   20110304 1.2
Izlaz:             20110304 1.2
                   20140250 3.4
                   20140015 4.5
                   20130115 4.5
```

Test 2

```
Kolokvijum.txt:    20130015 9.5
                   20130010 9.5
                   20090103 0.5
                   20140005 10.0
                   20140120 1.3
                   20140038 2.5
Izlaz:             20090103 0.5
                   20140120 1.3
                   20140038 2.5
                   20130015 9.5
                   20130010 9.5
                   20140005 10.0
```

3.2 Višedimenzioni nizovi

Zadatak 22 Data je kvadratna matrica dimenzije n.

(a) Napisati funkciju koja izračunava trag matrice.

(b) Napisati funkciju koja izračunava euklidsku normu matrice.

(c) Napisati funkciju koja izračunava gornju vandijagonalnu normu matrice.

Napisati program koji testira napisane funkcije. Sa standardnog ulaza učitati dimanziju kvadratne matrice n ($0 < n \leq 100$), a zatim i elemente matrice. Na standardni izlaz ispisati rezultat primene svake od napisanih funkcija nad učitanom matricom.

Zadatak 23 Date su dve kvadratne matrice istih dimenzija n.

(a) Napisati funkciju koja proverava da li su matrice jednake.

(b) Napisati funkciju koja izračunava zbir matrica.

(c) Napisati funkciju koja izračunava proizvod matrica.

Napisati program koji testira napisane funkcije. Sa standardnog ulaza učitati dimanziju kvadratnih matrica n ($0 < n \leq 100$), a zatim i elemente matrica. Na standardni izlaz ispisati rezultat primene svake od napisanih funkcija nad učitanim matricama.

Test 1

```
Ulaz:   3
        1 2 3
        1 2 3
        1 2 3
        1 2 3
        1 2 3
        1 2 3
Izlaz:  da
        2 4 6
        2 4 6
        2 4 6
        6 12 18
        6 12 18
        6 12 18
```

Zadatak 24 Relacija se može predstaviti kvadratnom matricom nula i jedinica na sledeći način: dva elementa i i j su u relaciji ukoliko se u preseku i-te vrste i j-te kolone matrice nalazi broj 1, a nisu u relaciji ukoliko se tu nalazi broj 0.

(a) Napisati funkciju koja proverava da li je relacija zadata matricom refleksivna.

(b) Napisati funkciju koja proverava da li je relacija zadata matricom simetrična.

(c) Napisati funkciju koja proverava da li je relacija zadata matricom tranzitivna.

(d) Napisati funkciju koja određuje refleksivno zatvorenje relacije (najmanju refleksivnu relaciju koja sadrži datu).

(e) Napisati funkciju koja određuje simetrično zatvorenje relacije (najmanju simetričnu relaciju koja sadrži datu).

(f) Napisati funkciju koja određuje refleksivno-tranzitivno zatvorenje relacije (najmanju refleksivnu i tranzitivnu relaciju koja sadrži datu)(Napomena: koristiti Varšalov algoritam).

Napisati program koji učitava matricu iz datoteke čije se ime zadaje kao prvi argument komandne linije. U prvoj liniji datoteke nalazi se dimenzija matrice n ($0 < n \leq 64$), a potom i sami elementi matrice. Na standardni izlaz ispisati rezultat testiranja napisanih funkcija.

Test 1

```
Datoteka:  4
           1 0 0 0
           0 1 1 0
           0 0 1 0
           0 0 0 0
Izlaz:     refleksivnost: ne
           simetricnost: ne
           tranzitivnost: da
           refleksivno zatvorenje:
           1 0 0 0
           0 1 1 0
           0 0 1 0
           0 0 0 1
           simetricno zatvorenje:
           1 0 0 0
           0 1 1 0
           0 1 1 0
           0 0 0 0
           refleksivno-tranzitivno zatvorenje:
           1 0 0 0
           0 1 1 0
           0 0 1 0
           0 0 0 1
```

Zadatak 25 Data je kvadratna matrica dimenzije n.

(a) Napisati funkciju koja određuje najveći element matrice na sporednoj dijagonali.

(b) Napisati funkciju koja određuje indeks kolone koja sadrži najmanji element matrice.

(c) Napisati funkciju koja određuje indeks vrste koja sadrži najveći element matrice.

(d) Napisati funkciju koja određuje broj negativnih elemenata matrice.

Napisati program koji testira napisane funkcije. Sa standardnog ulaza učitati elemente celobrojne kvadratne matrice čija se dimenzija n ($0 < n \leq 32$) zadaje kao argument komandne linije.

Test 1

```
Poziv:  ./a.out 3
Ulaz:   1 2 3
        -4 -5 -6
        7 8 9
Izlaz:  7 2 2 3
```

Zadatak 26 Napisati funkciju kojom se proverava da li je zadata kvadratna matrica dimenzije n ortonormirana. Matrica je ortonormirana ako je skalarni proizvod svakog para različitih vrsta jednak nuli, a skalarni proizvod vrste sa samom sobom jednak jedinici. Napisati program koji testira napisanu funkciju. Sa standardnog ulaza učitati dimenziju celobrojne kvadratne matrice n ($n \leq 32$), a zatim i njene elemente. Na standardni izlaz ispisati rezultat primene napisane funkcije na učitanu matricu.

Test 1

```
Ulaz:  4
       1 0 0 0
       0 1 0 0
       0 0 1 0
       0 0 0 1
Izlaz: da
```

Test 2

```
Ulaz:  3
       1 2 3
       5 6 7
       1 4 2
Izlaz: ne
```

Zadatak 27 Data je matrica dimenzije $n \times m$.

(a) Napsiati funkciju koja učitava elemente matrice sa standardnog ulaza

(b) Napsiati funkciju koja na standardni izlaz spiralno ispisuje elemente matrice.

Napisati program koji testira napisane funkcije. Sa standardnog ulaza učitati dimenzije matrice n ($0 < n \leq 10$) i m ($0 < n \leq 10$), a zatim i elemente matrice (pozivom gore napisane funkcije). Na standardni izlaz spiralno ispisati elemente učitane matrice.

Zadatak 28 Milena: Ovo bi trebalo da ide u pretraživanje/sortiranje? Napisati funkciju koja vrši leksikografsko opadajuće sortiranje niza niski (pretpostaviti da ima najvise 1000 niski, od kojih svaka ima najvise 30 karaktera). Napisati program koji testira rad napisane funkcije. Niske učitati iz datoteke „niske.txt“ (prva linija datoteke sadrži broj niski, a svaka sledeća po jednu nisku). Na standardni izlaz ispisati leksikografski opadajuće sortirane niske.

Zadatak 29 Napisati funkciju koja izračunava k-ti stepen kvadratne matrice dimenzije n ($n \leq 32$). Napisati program koji testira napisanu funkciju. Sa standardnog ulaza učitati dimenziju celobrojne matrice n, elemente matrice i stepen k ($0 < k \leq 10$). Na standardni izlaz ispisati rezultat primene napisane funkcije. Napomena: voditi računa da se prilikom stepenovanja matrice izvrši što manji broj množenja.

Test 1

```
Ulaz:  3
       1 2 3
       4 5 6
       7 8 9
       8
Izlaz: 510008400 626654232 743300064
       1154967822 1419124617 1683281412
       1799927244 2211595002 2623262760
```

3.3 Dinamička alokacija memorije

Zadatak 30 Napisati program koji sa standardnog ulaza učitava niz celih brojeva a zatim na standardni izlaz ispisuje ove brojeve u obrnutom poretku. Ne praviti nikakavo ograničenje za dimenziju niza.

Zadatak 31 Napisati funkciju koja kao rezultat vraća nisku koja se dobija nadovezivanjem dve niske, bez promene njihovog sadržaja. Napisati program koji testira rad napisane funkcije. Sa standardnog ulaza učitati dve niske karaktera. Na standardni izlaz ispisati nisku koja se dobija njihovim nadovezivanjem.

Zadatak 32 Napisati program koji sa standardnog ulaza učitava niz celih brojeva. Brojevi se unose sve dok se ne unese nula. Ne praviti nikakve pretpostavke o dimenziji niza. Na standardni izlaz ispisati ovaj niz brojeva u obrnutom poretku. Zadatak uraditi na dva načina:

(a) realokaciju memorije niza vršiti korišćenjem `malloc()` funkcije,

(b) realokaciju memorije niza vršiti korišćenjem `realloc()` funkcije.

Zadatak 33 Napisati program koji sa standardnog ulaza učitava matricu celih brojeva. Prvo se učitavaju dimenzije matrice n i m (ne praviti nikakve pretpostavke o njihovoj veličini), a zatim i elementi matrice. Na standardni izlaz ispisati trag matrice.

Zadatak 34 Data je celobrojna matrica dimenzije $n \times m$ napisati:

(a) Napisati funkciju koja vrši učitavanje matrice sa standardnog ulaza.

(b) Napisati funkciju koja ispisuje elemente ispod glavne dijagonale matrice.

Napisati program koji testira napisane funkcije. Sa standardnog ulaza učitati n i m (ne praviti nikakve pretpostavke o njihovoj veličini), zatim učitati elemente matrice i na standardni izlaz ispisati elemente ispod glavne dijagonale matrice.

Zadatak 35 Milena: Ovo bi trebalo da ide u pretraživanje/sortiranje? U datoteci „pesme.txt“ nalaze se informacije o gledanosti pesama na Youtube-u. Format datoteke sa informacijama je sledeći:

- U prvoj liniji datoteke može i ne mora da se nalazi ukupan broj pesama prisutnih u datoteci.
- Svaki naredni red datoteke sadrži informacije o gledanosti pesama u formatu `izvodjac - naslov, brojGledanja`.

Napisati program koji učitava informacije o pesmama i vrši sortiranje pesama u zavisnosti od argumenata komandne linije na sledeći način:

- nema opcija, sortiranje se vrši po broju gledanja;
- prisutna je opcija `-i`, sortiranje se vrši po imenima izvođača;
- prisutna je opcija `-n`, sortiranje se vrši po naslovu pesama.

Na standardni izlaz ispisati informacije o pesmama sortirane na opisan način.

(a) Uraditi zadatak uz pretpostavku da je maksimalna dužina imena izvođača 20 karaktera, a imena naslova pesme 50 kraraktera.

(b) Uraditi zadatak bez pravljenja pretpostavki o maksimalnoj dužini imena izvođača i naslova pesme.

Test 1

```
Poziv: ./a.out
Datoteka:  5
           Ana - Nebo, 2342
           Laza - Oblaci, 29
           Pera - Ptice, 327
           Jelena - Sunce, 92321
           Mika - Kisa, 5341
Izlaz:     Jelena - Sunce, 92321
           Mika - Kisa, 5341
           Ana - Nebo, 2342
           Pera - Ptice, 327
           Laza - Oblaci, 29
```

Zadatak 36 Za zadatu matricu dimenzije $n \times m$ napisati funkciju koja izračunava redni broj kolone matrice čiji je zbir maksimalan. Napisati program koji testira ovu funkciju. Sa standardnog ulaza učitati dimenzije matrice n i m, a zatim elemente matrice.

Zadatak 37 Za zadatu kvadratnu matricu dimenzije n napisati funkciju koja menja njen sadržaj tako što polovi elemente iznad glavne dijagonale, duplira elemente ispod glavne dijagonale, dok elemente na glavnoj dijagonali ostavlja nepromenjene. Napisati program koji testira ovu funkciju za vrednosti koje se učitavaju iz datoteke „matrica.txt“. U datoteci se nalazi prvo dimenzija matrice, a zatim redom elementi matrice.

Zadatak 38 Za zadatu kvadratnu matricu dimenzije n napisati funkciju koja izračunava zbir apsolutnih vrednosti matrice ispod sporedne dijagonale. Napisati program koji testira ovu funkciju za vrednosti koje se učitavaju iz datoteke čije se ime zadaje kao argument komandne linije. U datoteci se nalazi prvo dimenzija matrice, a zatim redom elementi matrice.

Zadatak 39 Za zadatu kvadratnu matricu dimenzije n napisati funkciju koja vrši sortiranje vrsta matrice, rastuće na osnovu sume elemenata u vrsti. Napisati program koji testira ovu funkciju. Sa standardnog ulaza se prvo unosi dimenzija matrice, a zatim redom elementi matrice. Rezultujuću matricu ispisati na standardni izlaz.

Zadatak 40 Petar sakuplja sličice igrača za predstojeće Svetsko prvenstvo u fudbalu. U datoteci „slicice.txt“ se nalaze informacije o sličicama koje mu nedostaju u formatu: `redni_broj_sličice ime_reprezentacije_kojoj_sličica_pripada`. Pomozite Petru da otkrije koliko mu sličica ukupno nedostaje, kao i da pronađe ime reprezentacije čijih sličica ima najmanje. Dobijene podatke ispisati na standardni izlaz. Napomena: za realokaciju memorije koristiti `realloc()` funkciju.

Zadatak 41 U datoteci „temena.txt“ se nalaze tačke koje predstavljaju temena nekog n-tougla. Napisati program koji na osnovu sadržaja datoteke na standardni izlaz ispisuje o kom n-touglu je reč, a zatim i vrednosti njegovog obima i površine. Pretpostavka je da će mnogougao biti konveksan.

Zadatak 42 Napisati program koji na osnovu dve matrice dimenzija $m \times n$ formira matricu dimenzije $2 \cdot m \times n$ tako što naizmenično kombinuje jednu vrstu prve matrice i jednu vrstu druge matrice. Matirce su zapisane u datoteci „matrice.txt“. U prvom redu se nalaze dimenzije matrica m i n, u narednih m redova se nalaze vrste prve matrice, a u narednih m redova vrste druge matrice. Rezultujuću matricu ispisati na standardni izlaz.

Zadatak 43 Za zadatu kvadratnu matricu dimenzije n napisati funkciju koja sortira kolone matrice, opadajuće, na osnovu vrednosti prvog elementa u koloni.

Napisati program koji testira ovu funkciju. Sa standardnog ulaza se prvo unosi dimenzija matrice, a zatim redom elementi matrice. Rezultujuću matricu ispisati na standardni izlaz.

Zadatak 44 Na ulazu se zadaje niz celih brojeva čiji se unos završava nulom. Napisati funkciju koja od zadatog niza formira matricu tako da prva vrsta odgovara unetom nizu, a svaka naredna se dobija cikličkim pomeranjem elemenata niza za jednu poziciju ulevo.

Napisati program koji testira ovu funkciju. Sa standardnog ulaza se prvo unosi dimenzija matrice, a zatim redom elementi matrice. Rezultujuću matricu ispisati na standardni izlaz.

3.4 Pokazivači na funkcije

Milena: ovde nedostaju razni zadaci

Zadatak 45 Napisati program koji tabelarno štampa vrednosti proizvoljne realne funkcije sa jednim realnim argumentom, odnosno izračunava i ispisuje vrednosti date funkcije na diskretnoj ekvidistantnoj mreži od n tačaka intervala $[a, b]$. Realni brojevi a i b $(a < b)$ kao i ceo broj n $(n \geq 2)$ se učitavaju sa standardnog ulaza. Ime funkcije se zadaje kao argument komandne linije (`sin`, `cos`, `tan`, `atan`, `acos`, `asin`, `exp`, `log`, `log10`, `sqrt`, `floor`, `ceil`, `sqr`).

3.5 Rešenja

Rešenje 9

```
#include<stdio.h>

int main(){
  printf("Hello bitovi!\n");
  return 0;
}
```

Rešenje 10

```
#include<stdio.h>

int main(){
  printf("Hello bitovi!\n");
  return 0;
}
```

Rešenje 11

```
#include<stdio.h>

int main(){
  printf("Hello bitovi!\n");
  return 0;
}
```

Rešenje 12

```
#include<stdio.h>

int main(){
  printf("Hello bitovi!\n");
  return 0;
}
```

Rešenje 13

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 14

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 15

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 16

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 17

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 18

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 19

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 20

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 21

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 22

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 23

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 24

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
```

```
5   return 0;
}
```

Rešenje 25

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 26

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 27

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 28

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 29

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 30

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 31

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 32

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 33

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 34

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 35

```c
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 36

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 37

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 38

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 39

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 40

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 41

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
```

```
  return 0;
}
```

Rešenje 42

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 43

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 44

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Rešenje 45

```
#include<stdio.h>

int main(){
  printf("Hello pokazivaci!\n");
  return 0;
}
```

Glava 4

Pretraživanje

4.1 Zadaci

4.2 Rešenja

Glava 5

Sortiranje

5.1 Zadaci

5.2 Rešenja

Glava 6

Liste

6.1 Zadaci

6.2 Rešenja

Glava 7

Drveta

7.1 Zadaci

7.2 Rešenja

Glava 8

Razno

8.1 Zadaci

8.2 Rešenja

Glava 9

Ispitni rokovi

9.1 Zadaci

Programiranje 2, praktični deo ispita, jun 2015.

Zadatak 46

Kao argument komandne linije zadaje se ime ulazne datoteke u kojoj se nalaze niske. U prvoj liniji datoteke nalazi se informacija o broju niski, a zatim u narednim linijama po jedna niska ne duža od `50` karaktera.

Napisati program u kojem se dinamički alocira memorija za zadati niz niski, a zatim se na standardnom izlazu u redosledu suprotnom od redosleda čitanja ispisuju sve niske koje počinju velikim slovom.

U slučaju pojave bilo kakve greške na standardnom izlazu ispisati vrednost `-1` i prekinuti izvršavanje programa.

Test 1

```
Sadržaj datoteke:
 5
 Programiranje
 Matematika
 12345
 dInAmiCnArEc
 Ispit
Izlaz:
 Ispit
 Matematika
 Programiranje
```

Test 2

```
Sadržaj datoteke:
  2
  maksimalano
  poena
Izlaz:
```

Test 3

```
Problem:
  datoteka
   ne postoji
Izlaz:
  -1
```

Zadatak 47

Data je biblioteka za rad sa binarnim pretrazivačkim stablima čiji čvorovi sadrže cele brojeve. Napisati funkciju `int sumirajN (Cvor * koren, int n)` koja izračunava zbir svih čvorova koji se nalaze na n-tom nivou stabla (koren se nalazi na nultom nivou, njegova deca na prvom nivou i tako redom). Ispravnost napisane funkcije testirati na osnovu zadate `main` funkcije i biblioteke za rad sa pretraživačkim stablima.

Napisati program koji sa standardnog ulaza učitava najpre prirodan broj n, a potom i brojeve sve do pojave nule koje smešta u stablo i ispisuje rezultat pozivanja funkcije `prebrojN` za broj n i tako kreirano stablo. U slučaju greške na standardni izlaz za grešku ispisati `-1`.

Test 1

```
Ulaz:
 2 8 10 3 6 14 13 7
    4 0
Izlaz:
 20
```

Test 2

```
Ulaz:
 0 50 14 5 2 4 56 8
    52 7 1 0
Izlaz:
 50
```

Zadatak 48 Sa standardnog ulaza učitava se broj vrsta i broj kolona celobrojne matrice A, a zatim i elementi matrice A. Napisati program koji će ispisati indeks kolone u kojoj se nalazi najviše negativnih elemenata. Ukoliko postoji više takvih kolona, ispisati indeks prve kolone. Može se pretpostaviti da je broj vrsta i broj kolona manji od `50`. U slučaju greške ispisati vrednost `-1` na standardni izlaz za greške.

Test 1

```
Ulaz:
  4
  5
  1  2  3  4  5
 -1  2 -3  4 -5
 -5 -4 -3 -2  1
 -1  0  0  0  0
Izlaz:
 0
```

Test 2

```
Ulaz:
 2
 3
 0 0 -5
 1 2 -4
Izlaz:
```

Test 3

```
Ulaz:
 -2
Izlaz (na stderr):
 -1
```

Programiranje 2, praktični deo ispita, jul 2015.

Zadatak 49

Napisati program koji kao prvi arugment komandne linije prima ime dokumenta u kome treba prebrojati sva pojavljivanja tražene niske (bez preklapanja) koja se navodi kao drugi argument komandne linije (iskoristiti funkciju standardne biblioteke `strstr`). U slučaju bilo kakve greške ispisati `-1` na standardni izlaz za greške. Pretpostaviti da linije datoteke neće biti duže od `127` karaktera.
Potpis funkcije strstr:
`char *strstr(const char *haystack, const char *needle);`
Funkcija traži prvo pojavljivanje podniske needle u nisci haystack, i vraća pokazivač na početak podniske, ili NULL ako podniska nije pronađena.

Test 1

```
Poziv:  ./a.out fajl
   .txt test
Datoteka:   Ovo je
   test primer.
        U njemu se
   rec test javlja
        vise puta.
   testtesttest
Izlaz:  5
```

Test 2

```
Poziv:   ./a.out
Izlaz  (na stderr):
    -1
```

Test 3

```
Poziv:  ./a.out fajl
   .txt .
Datoteka:   (prazna)
Izlaz:    0
```

Test 4

```
Poziv:   ./a.out
   fajl.txt foo
Datoteka:   (ne
   postoji)
Izlaz (na stderr):
     -1
```

Zadatak 50

Na početku datoteke "trouglovi.txt" nalazi se broj trouglova čije su koordinate temena zapisane u nastavku datoteke. Napisati program koji učitva trouglove, i ispisuje ih na standardni izlaz sortirane po površini opadajuće (koristiti Heronov obrazac: $P = \sqrt{s * (s-a) * (s-b) * (s-c)}$, gde je s poluobim trougla). U slučaju bilo kakve greške ispisati -1 na standardni izlaz za greške. Ne praviti nikave pretpostavke o broju trouglova u datoteci, i proveriti da li je datoteka ispravno zadata.

Test 1

```
Datoteka:  4
           0 0 0 1.2
    1 0
           0.3 0.3
   0.5 0.5 0.9 1
           -2 0 0 0
   0 1
           2 0 2 2
   -1 -1

Izlaz:     2 0 2 2
   -1 -1
           -2 0 0 0
   0 1
           0 0 0 1.2
    1 0
           0.3 0.3
   0.5 0.5 0.9 1
```

Test 2

```
Datoteka:  3
           1.2 3.2
   1.1 4.3

Izlaz:     -1
```

Test 3

```
Datoteka:    (nema
   datoteke)

Izlaz:      -1
```

Test 4

```
Datoteka:    0

Izlaz:
```

Zadatak 51 Data je biblioteka za rad sa binarnim pretraživačkim stablima celih brojeba. Napisati funkciju
`int f3(Cvor *koren, int n)`
koja u datom stablu prebrojava čvorove na n-tom nivou, koji imaju tačno jednog potomka. Pretpostaviti da se koren nalazi na nivou `0`. Ispravnost napisane funkcije testirati na osnovu zadate `main` funkcije i biblioteke za rad sa stablima.

Test 1

```
Ulaz:
 1 5 3 6 1 4 7 9
Izlaz:
 1
```

Test 2

```
Ulaz:
 2 5 3 6 1 0 4 7 9
Izlaz:
 2
```

Test 3

```
Ulaz:
 0 4 2 5
Izlaz:
 0
```

Test 4

```
Ulaz:
  3
Izlaz:
 0
```

Test 5

```
Ulaz:
  -1 4 5 1 7
Izlaz:
 0
```

9.2 Rešenja

Rešenje 46

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#define MAX 50

void greska(){
    printf("-1\n");
    exit(EXIT_FAILURE);
}

int main(int argc, char* argv[]){

    FILE* ulaz;
    char** linije;
    int i, j, n;

    /* Proveravamo argumente komandne linije.
  */
```

```
19      if(argc!=2){
            greska();
21      }

23      /* Otvaramo datoteku čije ime je navedeno kao argument
        komandne linije neposredno nakon imena programa koji se
        poziva. */
        ulaz=fopen(argv[1], "r");
25      if(ulaz==NULL){
            greska();
27      }

29      /* Učitavamo broj linija. */
        fscanf(ulaz, "%d", &n);
31
        /* Alociramo memoriju na osnovu učitanog broja linija.*/
33      linije=(char**)malloc(n*sizeof(char*));
        if(linije==NULL){
35          greska();
        }
37      for(i=0; i<n; i++){
            linije[i]=malloc(MAX*sizeof(char));
39          if(linije[i]==NULL){
                for(j=0; j<i; j++){
41                  free(linije[j]);
                }
43              free(linije);
                greska();
45          }
        }
47
        /* Učitavamo svih n linija iz datoteke. */
49      for(i=0; i<n; i++){
            fscanf(ulaz, "%s", linije[i]);
51      }

53      /* Ispisujemo u odgovarajućem poretku učitane linije koje
        zadovoljavaju kriterijum. */
        for(i=n-1; i>=0; i--){
55          if(isupper(linije[i][0])){
                printf("%s\n", linije[i]);
57          }
        }
59
        /* Oslobadjamo memoriju koju smo dinamički alocirali. */
61      for(i=0; i<n; i++){
            free(linije[i]);
63      }

65      free(linije);

67      /* Zatvaramo datoteku. */
        fclose(ulaz);
69
        /* Završavamo sa programom. */
71      return 0;
```

```
}
```

Rešenje 47

```
#include <stdio.h>
#include "stabla.h"


int sumirajN (Cvor * koren, int n){
    if(koren==NULL){
            return 0;
    }

    if(n==0){
        return koren->broj;
    }

    return sumirajN(koren->levo, n-1) + sumirajN(koren->desno, n
   -1);
}


int main(){
    Cvor* koren=NULL;
    int n;
    int nivo;

    /* Čitamo vrednost nivoa */
    scanf("%d", &nivo);


    while(1){

        /* Čitamo broj sa standardnog ulaza */
        scanf("%d", &n);

        /* Ukoliko je korisnik uneo 0, prekidamo dalje čitanje.
   */
        if(n==0){
            break;
        }

        /* A ako nije, dodajemo procitani broj u stablo. */
        dodaj_u_stablo(&koren, n);

    }

    /* Ispisujemo rezultat rada tražene funkcije */
    printf("%d\n", sumirajN(koren,nivo));

    /* Oslobadjamo memoriju */
    oslobodi_stablo(&koren);


```

```
    /* Prekidamo izvršavanje programa */
    return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include "stabla.h"

Cvor* napravi_cvor(int b ) {
    Cvor* novi = (Cvor*) malloc(sizeof(Cvor));
    if( novi == NULL)
        return NULL;

    /* Inicijalizacija polja novog čvora */
    novi->broj = b;
    novi->levo = NULL;
    novi->desno = NULL;

    return novi;
}


void oslobodi_stablo(Cvor** adresa_korena)  {
      /* Prazno stablo i nema šta da se oslobadja */
    if( *adresa_korena == NULL)
        return;

    /* Rekurzivno oslobadjamo najpre levo, a onda i desno
   podstablo*/
    if( (*adresa_korena)->levo )
        oslobodi_stablo(&(*adresa_korena)->levo);
    if( (*adresa_korena)->desno)
        oslobodi_stablo(&(*adresa_korena)->desno);

    free(*adresa_korena);
    *adresa_korena =NULL;
}


void proveri_alokaciju( Cvor* novi) {
    if( novi == NULL) {
        fprintf(stderr, "Malloc greska za nov cvor!\n");
        exit(EXIT_FAILURE);
    }
}

void dodaj_u_stablo(Cvor** adresa_korena, int broj) {
    /* Postojece stablo je prazno*/
    if( *adresa_korena == NULL){
        Cvor* novi = napravi_cvor(broj);
        proveri_alokaciju(novi);
        *adresa_korena = novi;  /*  Kreirani čvor novi će biti
   od sada koren stabla*/
        return;
    }
```

```
51      /* Brojeve smeštamo u uredjeno binarno stablo, pa
        ako je broj koji ubacujemo manji od broja koji je u korenu
       */
53      if( broj < (*adresa_korena)->broj)          /* dodajemo u
       levo podstablo */
            dodaj_u_stablo(&(*adresa_korena)->levo, broj);
55      /* ako je broj manji ili jednak od broja koji je u korenu
       stabla, dodajemo nov čvor desno od korena */
        else
57          dodaj_u_stablo(&(*adresa_korena)->desno, broj);
}
```

```
1  #ifndef __STABLA_H__
   #define __STABLA_H__ 1
3
   /* Struktura kojom se predstavlja čvor drveta */
5  typedef struct dcvor{
       int broj;
7      struct dcvor* levo, *desno;
   } Cvor;
9
   /* Funkcija alocira prostor za novi čvor drveta, inicijalizuje
       polja
11      strukture i vraća pokazivač na nov čvor */
   Cvor* napravi_cvor(int b );
13
   /* Oslobadjamo dinamički alociran prostor za stablo
15  * Nakon oslobadjanja se u pozivajućoj funkciji koren
    * postavljana NULL, jer je stablo prazno  */
17 void oslobodi_stablo(Cvor** adresa_korena);
19
   /* Funkcija proverava da li je novi čvor ispravno alociran,
21  * i nakon toga prekida program */
   void proveri_alokaciju( Cvor* novi);
23
25 /* Funkcija dodaje nov čvor u stablo i
    * ažurira vrednost korena stabla u pozivajućoj funkciji.
27  */
   void dodaj_u_stablo(Cvor** adresa_korena, int broj);
29
   #endif
```

Rešenje 48

```
#include <stdio.h>
2  #define MAX 50
4
   int main(){
6      int m[MAX][MAX];
       int v, k;
8      int i, j;
       int max_broj_negativnih, max_indeks_kolone;
```

```
    int broj_negativnih;

    /* Učitavamo dimenzije matrice */
    scanf("%d", &v);
    scanf("%d", &k);

    if(v<0 || v>MAX || k<0 || k>MAX){
        fprintf(stderr, "-1\n");
        return 0;
    }

    /* Učitavamo elemente matrice */
    for(i=0; i<v; i++){
        for(j=0; j<k; j++){
            scanf("%d", &m[i][j]);
        }
    }

    /*Pronalazimo kolonu koja sadrži najveći broj negativnih
    elemenata */
    max_indeks_kolone=0;

    max_broj_negativnih=0;
    for(i=0; i<v; i++){
        if(m[i][0]<0){
            max_broj_negativnih++;
        }

    }

    for(j=0; j<k; j++){
        broj_negativnih=0;
        for(i=0; i<v; i++){
            if(m[i][j]<0){
                broj_negativnih++;
            }
            if(broj_negativnih>max_broj_negativnih){
                max_indeks_kolone=j;
            }
        }

    }

    /* Ispisujemo traženi rezultat */
    printf("%d\n", max_indeks_kolone);

    /* Završavamo program */
    return 0;
}
```

Rešenje 49

Rešenje 50

Rešenje 51

Listings