

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.

Autori

Sadržaj

1	Rekurzija	3
1.1	Zadaci	3
1.2	Rešenja	6
2	Bitovi	9
2.1	Zadaci	9
2.2	Rešenja	11
3	Pokazivači	13
3.1	Pokazivači i aritmetika sa pokazivačima	13
3.2	Višedimenzioni nizovi	17
3.3	Dinamička alokacija memorije	21
3.4	Pokazivači na funkcije	23
3.5	Rešenja	24
4	Pretraživanje	31
4.1	Zadaci	31
4.2	Rešenja	31
5	Sortiranje	33
5.1	Zadaci	33
5.2	Rešenja	33
6	Liste	35
6.1	Zadaci	35
6.2	Rešenja	35
7	Drveta	37
7.1	Zadaci	37
7.2	Rešenja	37
8	Razno	39
8.1	Zadaci	39
8.2	Rešenja	39
9	Ispitni rokovi	41
9.1	Zadaci	41
9.2	Rešenja	44

Literatura

51

Glava 1

Rekurzija

1.1 Zadaci

Zadatak 1 Napisati rekurzivnu funkciju koja sumira elemente niza celih brojeva. Napisati program koji testira napisane funkcije. Sa standardnog ulaza učitati dimenziju n ($0 < n \leq 100$) celobrojnog niza, a zatim i elemente niza. Na standardni izlaz ispisati rezultat primene napisane funkcije nad učitanim nizom.

Zadatak 2 Napisati rekurzivnu funkciju koja izracunava x^k , za dati realni broj x i prirodan broj k . Napisati program koji testira napisanu funkciju tako što računa 2^{10} .

Zadatak 3 Nina: Ovaj zadatak treba posle poglavlja o bitovima. Možda napraviti podpoglavlje u poglavlju Bitovi da se odvoje oni zadaci koji se odnose i na rekurziju. Napisati rekurzivnu funkciju vraća broj bitova koji su postavljeni na 1, u binarnoj reprezentaciji njenog celobrojnog argumenta. Napisati program koji testira napisanu funkciju za brojeve 127, 128, 0x00FF00FF, 0xFFFFFFFF.

Dodatak za vežbu: Ispratiti stanje na sistemskom steku u toku izvršavanja jednog poziva funkcije.

Zadatak 4 Napisati rekurzivnu funkciju koja proverava da li je broj cifara nekog broja paran ili neparan. Napisati program koji testira napisanu funkciju tako što se za broj učitani sa standardnog ispisuje da li je paran ili neparan. *Napomena: koristiti uzajamnu (posrednu) rekurziju.*

Zadatak 5 Napisati repno-rekurzivnu funkciju koja izracunava $n!$. Napisati program koji testira napisanu funkciju za poizvoljan broj n ($n \leq 12$) unet sa standardnog ulaza.

Zadatak 6 Napisati rekurzivnu funkciju koja računa n -ti element u nizu, čiji se elementi računaju na osnovu sledećih rekurentnih relacija:

$$F(0) = 0$$

$$F(1) = 1$$

$$F(n) = a * F(n - 1) + b * F(n - 2)$$

Napisati funkciju tako da se problemi manje dimenzije rešavaju samo jedan put.

1 Rekurzija

Zadatak 7 Napisati rekurzivnu funkciju koja prikazuje sve permutacije skupa $\{1, 2, \dots, n\}$. Napisati program koji testira napisanu funkciju za poizvoljan prirodan broj n ($n \leq 50$) unet sa standardnog ulaza.

Zadatak 8

- a) Napisati rekurzivnu funkciju koja izračunava d_n kao sumu elemenata n -te hipotenuze Paskalovog trougla.
- b) Napisati rekurzivnu funkciju koja izračunava vrednost polja (i, j) .

Paskalov trougao se dobija tako što mu je svako polje (izuzev 0-te vrste i 0-kolone) zbir jednog polja levo i jednog polja iznad.

```
      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
 1 5 10 10 5 1
```

Zadatak 9 Napisati rekurzivnu funkciju koja određuje maksimum niza celih brojeva. Napisati program koji testira ovu funkciju, za niz koji se učitava sa standardnog ulaza. Niz neće biti duži od 256, i njegovi elementi se unose sve do kraja ulaza.

<i>Test 1</i>	<i>Test 2</i>
<pre> Ulaz: 3 2 1 4 21 Izlaz: 21</pre>	<pre> Ulaz: 2 -1 0 -5 -10 Izlaz: 2</pre>
<i>Test 3</i>	<i>Test 4</i>
<pre> Ulaz: 1 11 3 5 8 1 Izlaz: 11</pre>	<pre> Ulaz: 5 Izlaz: 5</pre>

Zadatak 10 Napisati rekurzivnu funkciju koja izračunava skalarni proizvod dva data vektora. Napisati program koji testira ovu funkciju, za nizove koji se učitavaju sa standardnog ulaza. Dužina neće biti veća od 256. Prvo se unosi dimenzija nizova, a zatim i sami njihovi elementi.

<i>Test 1</i>	<i>Test 2</i>
<pre> Ulaz: 3 1 2 3 1 2 3 Izlaz: 14</pre>	<pre> Ulaz: 2 3 5 2 6 Izlaz: 36</pre>
<i>Test 3</i>	
<pre> Ulaz: 0 Izlaz: 0</pre>	

Zadatak 11 Napisati rekurzivnu funkciju koja sabira dekadne cifre datog celog broja x . Napisati program koji testira ovu funkciju, za broj koji se učitava sa standardnog ulaza.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
Ulaz: 123 Izlaz: 6	Ulaz: 23156 Izlaz: 17	Ulaz: 1432 Izlaz: 10
<i>Test 4</i>	<i>Test 5</i>	
Ulaz: 1 Izlaz: 1	Ulaz: 0 Izlaz: 0	

Zadatak 12 Napisati rekurzivnu funkciju koja računa broj pojavljivanja elementa x u nizu a dužine n . Napisati program koji testira ovu funkciju, za x i niz koji se učitavaju sa standardnog ulaza. Niz neće biti duži od 256. Prvo se unosi x , za zatim elementi niza sve do kraja ulaza.

<i>Test 1</i>	<i>Test 2</i>
Ulaz: 4 1 2 3 4 Izlaz: 1	Ulaz: 11 3 2 11 14 11 43 1 Izlaz: 2
<i>Test 3</i>	
Ulaz: 1 3 21 5 6 Izlaz: 0	

Zadatak 13 Napisati rekurzivnu funkciju koja ispituje da li je data niska palindrom. Napisati program koji testira ovu funkciju. Pretpostaviti da niska neće biti duža od 32, i da se unosi sa standardnog ulaza.

<i>Test 1</i>		<i>Test 2</i>			
Ulaz:	programiranje	Ulaz:	anavolimilovana		
Izlaz:	ne	Izlaz:	da		
<i>Test 3</i>		<i>Test 4</i>		<i>Test</i>	
Ulaz:	a	Ulaz:	aba	Ulaz:	aa
Izlaz:	da	Izlaz:	da	Izlaz:	da

Zadatak 14 Napisati rekurzivnu funkciju kojom se proverava da li su tri zadata broja uzastopni članovi niza. Potom, napisati program koji je testira. Sa standardnog ulaza se unose najpre tri tražena broja, a zatim elementi niza, sve do kraja ulaza. Pretpostaviti da ih neće biti više od 256.

<i>Test 1</i>										<i>Test 2</i>											
	Ulaz:	1	2	3	4	1	2	3	4	5		Ulaz:	1	2	3	11	1	2	4	3	6
	Izlaz:	da										Izlaz:	ne								

<i>Test 3</i>									
	Ulaz:	1	2	3	1	2			
	Izlaz:	ne							

Zadatak 15 Nina: Ovaj bi trebalo da ide u razne zadatke ili u okviru poglavlja sa bitovima.

Napisati rekurzivnu funkciju koja štampa bitovsku reprezentaciju neoznačenog celog broja, i program koji je testira za ulaz koji se zadaje sa standardnog ulaza.

[illegible]

Zadatak 16 Napisati rekurzivnu funkciju koja prikazuje sve varijacije sa ponavljanjem dužine n skupa $\{a, b\}$, i program koji je testira, za n koje se učitava sa standardnog ulaza.

Test 1

Ulaz:	3
Izlaz:	a a a
	a a b
	a b a
	a b b
	b a a
	b a b
	b b a
	b b b

Zadatak 17 Modifikacija problema "Hanojskih kula": Data su četiri vertikalna štapa, na jednom se nalazi n diskova poluprečnika $1, 2, 3, \dots$ do n , tako da se najveći nalazi na dnu, a najmanji na vrhu. Ostala tri štapa su prazna. Potrebno je premestiti diskove na drugi štap tako da budu u istom redosledu, premestajući jedan po jedan disk, pri čemu se ni u jednom trenutku ne sme staviti veći disk preko manjeg, pri čemu se preostala dva štapa koriste kao pomoćni štapovi prilikom premeštanja.

1.2 Rešenja

Rešenje 1

```
1 #include <stdio.h>
```

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

Rešenje 2

```
#include<stdio.h>  
  
2 int main(){  
4     printf("Hello bitovi!\n"); /* Da li komentari rade ččžš*/  
    return 0;  
6 }
```

Rešenje 3

```
#include<stdio.h>  
  
2 int main(){  
4     printf("Hello bitovi!\n"); /* Da li komentari rade ččžš*/  
    return 0;  
6 }
```

Rešenje 4

Rešenje 5

Rešenje 6

Rešenje 7

Rešenje 8

Rešenje 9

Rešenje 10

Rešenje 11

Rešenje 12

Rešenje 13

Rešenje 14

Glava 2

Bitovi

2.1 Zadaci

Zadatak 18 Napisati:

- (a) funkciju `print_bits` koja štampa bitove u binarnom zapisu celog broja x .
- (b) program koja testira funkciju `print_bits` za proizvoljne brojeve.

Zadatak 19 Napisati funkciju koja broji bitove postavljene na 1 u zapisu broja x . Napisati program koji testira tu funkciju za broj `0xABCDE123`.

Zadatak 20 Napisati:

- (a) funkciju koja određuje najveći broj koji se može zapisati istim binarnim ciframa kao dati broj.
- (b) funkciju koja određuje najmanji broj koji se može zapisati istim binarnim ciframa kao dati broj.
- (c) program koji testira prethodno napisane funkcije.

Zadatak 21 Napisati:

- (a) funkciju koja određuje broj koji se dobija kada se n bitova datog broja, počevši od pozicije p postave na 0.
- (b) funkciju koja određuje broj koji se dobija kada se n bitova datog broja, počevši od pozicije p postave na 1.
- (c) funkciju koja određuje broj koji se dobija kada se n bitova datog broja, počevši od pozicije p i vraća ih kao bitove najmanje težine rezultata.
- (d) funkciju koja vraća broj koji se dobija upisivanjem poslednjih n bitova broja y u broj x , počevši od pozicije p .

- (e) funkciju koja vraća broj koji se dobija invertovanjem n bitova broja x počevši od pozicije p .
- (f) program koji testira prethodno napisane funkcije.

Zadatak 22 Napisati:

- (a) funkciju funkciju koja određuje broj koji se dobija rotiranjem u levo datog celog broja.
- (b) funkciju koja određuje broj koji se dobija rotiranjem u desno datog celog neoznačenog broja.
- (c) funkciju koja određuje broj koji se dobija rotiranjem u desno datog celog broja.
- (d) program koji testira prethodno napisane funkcije.

Zadatak 23 Napisati funkciju koja određuje ceo broj čiji binarni zapis je slika u ogledalu binarnog zapisa argumenta funkcije, a potom i program koji testira datu funkciju za argument `0xABCDE123`.

Zadatak 24 Napisati funkciju `int Broj01(unsigned int n)` koja za dati broj n vraća 1 ako u njegovom binarnom zapisu ima više jedinica nego nula, a inače vraća 0. Napisati program koji tu funkciju testira za broj koji se zadaje sa standardnog ulaza.

	<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
Ulaz:	10	1024	2147377146
Izlaz:	0	0	1

	<i>Test 4</i>
Ulaz:	1111111115
Izlaz:	0

Zadatak 25 Napisati funkciju koja broji koliko se puta kombinacija 11 (dve uzastopne jedinice) pojavljuje u binarnom zapisu celog neoznačenog broja x . Tri uzastopne jedinice se broje dva puta. Napisati program koji tu funkciju testira za broj koji se zadaje sa standardnog ulaza.

	<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
Ulaz:	11	1024	2147377146
Izlaz:	1	0	22

	<i>Test 4</i>
Ulaz:	1111111115
Izlaz:	6

Zadatak 26 Napisati program koji sa standardnog ulaza učitava pozitivan ceo broj, a na standardni izlaz ispisuje vrednost tog broja sa razmenjenim vrednostima bitova na pozicijama i , j . Pozicije i , j se učitavaju kao parametri komandne linije. Smatrati da je krajnji desni bit binarne reprezentacije 0-ti bit. Pri rešavanju nije dozvoljeno koristiti pomoćni niz niti aritmetičke operatore $+$, $-$, $/$, $*$, $\%$.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
<code>Poziv: ./a.out 1 2</code>	<code>Poziv: ./a.out 1 2</code>	<code>Poziv: ./a.out 12 12</code>
<code>Ulaz: 11</code>	<code>Ulaz: 1024</code>	<code>Ulaz: 12345</code>
<code>Izlaz: 13</code>	<code>Izlaz: 1024</code>	<code>Izlaz: 12345</code>

Zadatak 27 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.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
<code>Ulaz: 11</code>	<code>Ulaz: 1024</code>	<code>Ulaz: 12345</code>
<code>Izlaz: 0000000B</code>	<code>Izlaz: 00000400</code>	<code>Izlaz: 00003039</code>

Zadatak 28 Napisati funkciju koja za dva data neoznačena broja x i y invertuje u podatku x one bitove koji se poklapaju sa odgovarajućim bitovima u broju y . Ostali bitovi ostaju nepromenjeni. Napisati program koji tu funkciju testira za brojeve koji se zadaju sa standardnog ulaza.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
<code>Ulaz: 123 10</code>	<code>Ulaz: 3251 0</code>	<code>Ulaz: 12541 1024</code>
<code>Izlaz: 4294967285</code>	<code>Izlaz: 4294967295</code>	<code>Izlaz: 4294966271</code>

Zadatak 29 Napisati funkciju koja računa koliko petica bi imao ceo neoznačen broj x u oktalnom zapisu. Napisati program koji tu funkciju testira za broj koji se zadaje sa standardnog ulaza.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
<code>Ulaz: 123</code>	<code>Ulaz: 3245</code>	<code>Ulaz: 100328</code>
<code>Izlaz: 0</code>	<code>Izlaz: 2</code>	<code>Izlaz: 1</code>

2.2 Rešenja

Rešenje 18

```
1 #include <stdio.h>
```


2 Bitovi

```
3 int main(){  
    int a, b;  
5    printf("Hello bitovi!\n"); /* Da li komentari rade ččžšđ*/  
    if(a>b) a++;  
7    return 0;  
}
```

Rešenje [19](#)

Rešenje [20](#)

Rešenje [21](#)

Rešenje [22](#)

Rešenje [23](#)

Rešenje [24](#)

Rešenje [25](#)

Rešenje [26](#)

Rešenje [27](#)

Rešenje [28](#)

Rešenje [29](#)

Glava 3

Pokazivači

3.1 Pokazivači i aritmetika sa pokazivačima

Zadatak 30 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 31 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 32 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 33 Napisati program koji ispisuje broj prihvaćenih argumenata komandne linije, a zatim i same argumente kojima prethode njihovi redni brojevi.

Zadatak 34 Korišćenjem pokazivačke sintakse, napisati funkciju koja za datu nisku ispituje da li je palindrom. Napisati program koji vrši prebrojavanje argumentata komandne linije koji su palindromi.

Test 1

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

Zadatak 35 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 sadržaj datoteke i u
||          njoj ima reci koje imaju
||          1 karakter
|| Izlaz:  2
```

Zadatak 36 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 sadržaj datoteke
||          i u njoj ima reci koje se
||          završavaju na ke
|| Izlaz:  2
```

Zadatak 37 **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 38 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 39 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 40 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 41 *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 sortirani 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 42 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 sortirani 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 43 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 44 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
      1 2 3
Izlaz:  da
      2 4 6
      2 4 6
      2 4 6
      6 12 18
      6 12 18
      6 12 18
```

Zadatak 45 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 46 Data je kvadratna matrica dimenzije n .

- Napisati funkciju koja određuje najveći element matrice na sporednoj dijagonali.
- Napisati funkciju koja određuje indeks kolone koja sadrži najmanji element matrice.
- Napisati funkciju koja određuje indeks vrste koja sadrži najveći element matrice.
- 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 47 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 učitanoj matrici.

<i>Test 1</i>	<i>Test 2</i>
<pre> Ulaz: 4 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 Izlaz: da </pre>	<pre> Ulaz: 3 1 2 3 5 6 7 1 4 2 Izlaz: ne </pre>

Zadatak 48 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 ($0 < m \leq 10$), a zatim i elemente matrice (pozivom gore napisane funkcije). Na standardni izlaz spiralno ispisati elemente učitane matrice.

Zadatak 49 Milena: Ovo bi trebalo da ide u pretraživanje/sortiranje? Napisati funkciju koja vrši leksikografsko opadajuće sortiranje niza niski (pretpostaviti da ima najviše 1000 niski, od kojih svaka ima najviše 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 50 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 51 Napisati program koji sa standardnog ulaza učitava niz celih brojeva a zatim na standardni izlaz ispisuje ove brojeve u obrnutom poretku. Ne praviti nikakvo ograničenje za dimenziju niza.

Zadatak 52 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 53 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 54 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 55 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 56 **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.

- Uradi zadatak uz pretpostavku da je maksimalna dužina imena izvođača 20 karaktera, a imena naslova pesme 50 karaktera.
- Uradi 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 57 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 58 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 59 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 60 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 61 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 pronade ime reprezentacije čijih sličica ima najmanje. Dobijene podatke ispisati na standardni izlaz. Napomena: za realokaciju memorije koristiti `realloc()` funkciju.

Zadatak 62 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 63 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. Matrice 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 64 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 65 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 66 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 30

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello bitovi!\n");
5     return 0;
6 }
```

Rešenje 31

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello bitovi!\n");
5     return 0;
6 }
```

Rešenje 32

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello bitovi!\n");
5     return 0;
6 }
```

Rešenje 33

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello bitovi!\n");
5     return 0;
6 }
```

Rešenje 34

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 35

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 36

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 37

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 38

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 39

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 40

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 41

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 42

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 43

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 44

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 45

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5 }
```

```
5   return 0;
}
```

Rešenje 46

```
1  #include <stdio.h>

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

Rešenje 47

```
1  #include <stdio.h>

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

Rešenje 48

```
1  #include <stdio.h>
2
3  int main(){
4   printf("Hello pokazivaci!\n");
   return 0;
6 }
```

Rešenje 49

```
1  #include <stdio.h>
2
3  int main(){
4   printf("Hello pokazivaci!\n");
   return 0;
6 }
```

Rešenje 50

```
1  #include <stdio.h>
2
3  int main(){
4   printf("Hello pokazivaci!\n");
   return 0;
6 }
```

Rešenje 51


```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 52

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 53

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 54

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 55

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 56

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 57

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 58

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 59

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 60

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 61

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
5     return 0;
6 }
```

Rešenje 62

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello pokazivaci!\n");
```

3 Pokazivači

```
    return 0;
6 }
```

Rešenje 63

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 64

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 65

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

Rešenje 66

```
#include <stdio.h>

2
int main(){
4     printf("Hello pokazivaci!\n");
    return 0;
6 }
```

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 67

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.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
<code>Sadržaj datoteke:</code> 5 Programiranje Matematika 12345 dInAmiCnArEc Ispit Izlaz: Ispit Matematika Programiranje	<code>Sadržaj datoteke:</code> 2 maksimalano poena Izlaz:	<code>Problem:</code> datoteka ne postoji Izlaz: -1

Zadatak 68

Data je biblioteka za rad sa binarnim pretraživačkim 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.

9 Ispitni rokovi

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`.

<i>Test 1</i>	<i>Test 2</i>
<pre>Ulaz: 2 8 10 3 6 14 13 7 4 0 Izlaz: 20</pre>	<pre>Ulaz: 0 50 14 5 2 4 56 8 52 7 1 0 Izlaz: 50</pre>

Zadatak 69 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.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
<pre>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</pre>	<pre>Ulaz: 2 3 0 0 -5 1 2 -4 Izlaz:</pre>	<pre>Ulaz: -2 Izlaz (na stderr): -1</pre>

Programiranje 2, praktični deo ispita, jul 2015.

Zadatak 70

Napisati program koji kao prvi argument 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 foo
Datoteka:  (ne postoji)
Izlaz (na stderr):  -1

```

Test 4

```

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

```

Zadatak 71

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 72 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.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>
<code>Ulaz:</code> 1 5 3 6 1 4 7 9 <code>Izlaz:</code> 1	<code>Ulaz:</code> 2 5 3 6 1 0 4 7 9 <code>Izlaz:</code> 2	<code>Ulaz:</code> 0 4 2 5 <code>Izlaz:</code> 0
<i>Test 4</i>	<i>Test 5</i>	
<code>Ulaz:</code> 3 <code>Izlaz:</code> 0	<code>Ulaz:</code> -1 4 5 1 7 <code>Izlaz:</code> 0	

9.2 Rešenja

Rešenje 67

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4 #define MAX 50
5
6 void greska(){
7     printf("-1\n");
8     exit(EXIT_FAILURE);
9 }
10
11 int main(int argc, char* argv[]){
12
13     FILE* ulaz;
14     char** linije;
15     int i, j, n;
16
17     /* Proveravamo argumente komandne linije.
18     */
19     if(argc!=2){
20         greska();
21     }
22
23     /* Otvaramo datoteku čije ime je navedeno kao argument
24     komandne linije neposredno nakon imena programa koji se
25     poziva. */
26     ulaz=fopen(argv[1], "r");
27     if(ulaz==NULL){
28         greska();
29     }
30
31     /* Učitavamo broj linija. */
32     fscanf(ulaz, "%d", &n);
33
34     /* Alociramo memoriju na osnovu učitanoog broja linija.*/
```

```

33     linije=(char**)malloc(n*sizeof(char));
34     if(linije==NULL){
35         greska();
36     }
37     for(i=0; i<n; i++){
38         linije[i]=malloc(MAX*sizeof(char));
39         if(linije[i]==NULL){
40             for(j=0; j<i; j++){
41                 free(linije[j]);
42             }
43             free(linije);
44             greska();
45         }
46     }
47
48     /* Učitavamo svih n linija iz datoteke. */
49     for(i=0; i<n; i++){
50         fscanf(ulaz, "%s", linije[i]);
51     }
52
53     /* Ispisujemo u odgovarajućem poretku učitane linije koje
54     zadovoljavaju kriterijum. */
55     for(i=n-1; i>=0; i--){
56         if(isupper(linije[i][0])){
57             printf("%s\n", linije[i]);
58         }
59     }
60
61     /* Oslobađamo memoriju koju smo dinamički alocirali. */
62     for(i=0; i<n; i++){
63         free(linije[i]);
64     }
65
66     free(linije);
67
68     /* Zatvaramo datoteku. */
69     fclose(ulaz);
70
71     /* Završavamo sa programom. */
72     return 0;
73 }

```

Rešenje 68

```

1 #include <stdio.h>
2 #include "stabla.h"
3
4
5 int sumirajN (Cvor * koren, int n){
6     if(koren==NULL){
7         return 0;
8     }
9
10    if(n==0){

```

```
12         return koren->broj;
13     }
14     return sumirajN(koren->levo, n-1) + sumirajN(koren->desno, n-1);
15 }
16
17
18 int main(){
19     Cvor* koren=NULL;
20     int n;
21     int nivo;
22
23     /* Čitamo vrednost nivoa */
24     scanf("%d", &nivo);
25
26
27     while(1){
28
29         /* Čitamo broj sa standardnog ulaza */
30         scanf("%d", &n);
31
32         /* Ukoliko je korisnik uneo 0, prekidamo dalje čitanje.
33          */
34         if(n==0){
35             break;
36         }
37
38         /* A ako nije, dodajemo procitani broj u stablo. */
39         dodaj_u_stablo(&koren, n);
40     }
41
42     /* Ispisujemo rezultat rada tražene funkcije */
43     printf("%d\n", sumirajN(koren,nivo));
44
45     /* Oslobađamo memoriju */
46     oslobodi_stablo(&koren);
47
48
49     /* Prekidamo izvršavanje programa */
50     return 0;
51 }
```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include "stabla.h"
4
5 Cvor* napravi_cvor(int b ) {
6     Cvor* novi = (Cvor*) malloc(sizeof(Cvor));
7     if( novi == NULL)
8         return NULL;
9
10    /* Inicijalizacija polja novog čvora */
11    novi->broj = b;
12    novi->levo = NULL;
```

```

13     novi->desno = NULL;
15     return novi;
16 }
17
19 void oslobodi_stablo(Cvor** adresa_korena) {
20     /* Prazno stablo i nema šta da se oslobađa */
21     if( *adresa_korena == NULL)
22         return;
23
24     /* Rekurzivno oslobađamo najpre levo, a onda i desno
25     podstablo*/
26     if( (*adresa_korena)->levo )
27         oslobodi_stablo(&(*adresa_korena)->levo);
28     if( (*adresa_korena)->desno)
29         oslobodi_stablo(&(*adresa_korena)->desno);
30
31     free(*adresa_korena);
32     *adresa_korena = NULL;
33 }
34
35 void prover_i_alokaciju( Cvor* novi) {
36     if( novi == NULL) {
37         fprintf(stderr, "Malloc greska za nov cvor!\n");
38         exit(EXIT_FAILURE);
39     }
40 }
41
42 void dodaj_u_stablo(Cvor** adresa_korena, int broj) {
43     /* Postojeće stablo je prazno*/
44     if( *adresa_korena == NULL){
45         Cvor* novi = napravi_cvor(broj);
46         prover_i_alokaciju(novi);
47         *adresa_korena = novi; /* Kreirani čvor novi će biti
48         od sada koren stabla*/
49         return;
50     }
51
52     /* Brojeve smeštamo u uređeno binarno stablo, pa
53     ako je broj koji ubacujemo manji od broja koji je u korenu
54     */
55     if( broj < (*adresa_korena)->broj) /* dodajemo u
56     levo podstablo */
57         dodaj_u_stablo(&(*adresa_korena)->levo, broj);
58     /* ako je broj manji ili jednak od broja koji je u korenu
59     stabla, dodajemo nov čvor desno od korena */
60     else
61         dodaj_u_stablo(&(*adresa_korena)->desno, broj);
62 }
63
64 #ifndef __STABLA_H__
65 #define __STABLA_H__ 1
66
67 /* 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
/* Oslobađamo dinamički alociran prostor za stablo
15 * Nakon oslobađanja se u pozivajućoj funkciji koren
   * postavlja 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 prover_i_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 69

```
#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;
10    int broj_negativnih;

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

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

20
    /* Učitavamo elemente matrice */
22    for(i=0; i<v; i++){
        for(j=0; j<k; j++){
24            scanf("%d", &m[i][j]);
        }
    }
```

```

26     }

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

30
    max_broj_negativnih=0;
32    for(i=0; i<v; i++){
        if(m[i][0]<0){
34            max_broj_negativnih++;
        }

36    }

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

48        }

50    }

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

54
    /* Završavamo program */
56    return 0;
}

```

Rešenje 70

```

1  #include <stdio.h>
   #include <stdlib.h>
3  #include <string.h>
   #define MAX 128

5
   int main(int argc, char **argv) {
7       FILE *f;
       int brojac = 0;
9       char linija[MAX], *p;

11      if (argc != 3) {
          fprintf(stderr, "-1\n");
13          exit(EXIT_FAILURE);
      }

15
      if ((f = fopen(argv[1], "r")) == NULL) {
17          fprintf(stderr, "-1\n");
          exit(EXIT_FAILURE);
19      }

```

```
21 while (fgets(linija, MAX, f) != NULL) {  
    p = linija;  
23     while (1) {  
        p = strstr(p, argv[2]);  
25         if (p == NULL)  
            break;  
27         brojac++;  
        p = p + strlen(argv[2]);  
29     }  
    }  
31  
    fclose(f);  
33  
    printf("%d\n", brojac);  
35  
    return 0;  
37 }
```

Rešenje [71](#)

Rešenje [72](#)

Bibliografija

Listings

resenja/01_Rekurzija/101.c	6
resenja/01_Rekurzija/102.c	7
resenja/01_Rekurzija/103.c	7
resenja/02_Bitovi/201.c	11
resenja/03_Pokazivaci/301.c	24
resenja/03_Pokazivaci/302.c	24
resenja/03_Pokazivaci/303.c	24
resenja/03_Pokazivaci/304.c	24
resenja/03_Pokazivaci/305.c	25
resenja/03_Pokazivaci/306.c	25
resenja/03_Pokazivaci/307.c	25
resenja/03_Pokazivaci/308.c	25
resenja/03_Pokazivaci/309.c	25
resenja/03_Pokazivaci/310.c	25
resenja/03_Pokazivaci/311.c	26
resenja/03_Pokazivaci/312.c	26
resenja/03_Pokazivaci/313.c	26
resenja/03_Pokazivaci/314.c	26
resenja/03_Pokazivaci/315.c	26
resenja/03_Pokazivaci/322.c	26
resenja/03_Pokazivaci/323.c	27
resenja/03_Pokazivaci/324.c	27
resenja/03_Pokazivaci/325.c	27
resenja/03_Pokazivaci/326.c	27
resenja/03_Pokazivaci/327.c	27
resenja/03_Pokazivaci/328.c	28
resenja/03_Pokazivaci/329.c	28
resenja/03_Pokazivaci/330.c	28
resenja/03_Pokazivaci/331.c	28
resenja/03_Pokazivaci/332.c	28
resenja/03_Pokazivaci/333.c	28
resenja/03_Pokazivaci/336.c	29
resenja/03_Pokazivaci/337.c	29
resenja/03_Pokazivaci/338.c	29
resenja/03_Pokazivaci/339.c	29
resenja/03_Pokazivaci/340.c	29
resenja/03_Pokazivaci/341.c	29

LISTINGS

resenja/03_Pokazivaci/342.c	30
resenja/03_Pokazivaci/343.c	30
resenja/03_Pokazivaci/344.c	30
resenja/03_Pokazivaci/345.c	30
resenja/09_IspitniRokovi/901.c	44
resenja/09_IspitniRokovi/902.c	45
resenja/09_IspitniRokovi/902stabla.c	46
resenja/09_IspitniRokovi/902stabla.h	47
resenja/09_IspitniRokovi/903.c	48
resenja/09_IspitniRokovi/904.c	49