

**Петар Спалевић
Бранимир Јакшић
Стефан Панић**

ЗБИРКА РЕШЕНИХ ЗАДАТАКА ИЗ ПРОГРАМСКОГ ЈЕЗИКА С

I ДЕО

Косовска Митровица, 2011.

Збирка решених задатака из програмског језика **C**, I део

Прво издање

Аутори: Петар Спалевић, ванредни професор Факултета техничких наука у Косовској Митровици
Бранимир Јакшић, асистент на Факултету техничких наука у Косовској Митровици
Стефан Панић, доцент Природно-математичког факултета у Косовској Митровици

Рецензенти: Градимир Миловановић, редовни професор Мегатренд универзитета и
члан Српске академије наука и уметности
Братислав Мирић, редовни професор Државног универзитета у Новом Пазару

Технички уредници: Аутори

Издавач:

Штампа:

Тираж: 100 примерака

НАПОМЕНА: Фотокопирање или умножавање на било који начин или поновно објављивање ове књиге – у целини или деловима – није дозвољено без претходне сагласности и писменог одобрења издавача.

Предговор

Ова збирка задатака је помоћни уџбеник за учење програмирања на језику C. Задаци прате градиво које одговара предмету Програмирање 2 којег слушају студенти у оквиру студијског програма Електротехничко и рачунарско инжењерство на Факултету техничких наука у Косовској Митровици. Збирку могу користити и студенти других факултета који у оквиру својих предмета изучавају језик C, као и ученици средњих школа.

Збирка је тако конципирана да је могу користити и почетници у програмирању. Задаци су у свакој области изложени по тежини, од најлакших ка тежим. Кроз задатке, поред елемената самог језика, приказане су најчешће коришћени поступци у програмирању: претраживање и уређивање низова, обрада знаковних података, рад са показивачима и структурама, као и рад са датотекама.

Решења свих задатака су потпуна у смислу да се дати програмски кодови могу ивршавати на рачунари. Сви задаци су урађени, проверени и тестирани коришћењем програма DEV C/C++. Поред самих програмских кодова дати су изгледи на екрану након тестирања програма. За сложене задатке дата су објашњења у виду текста или пропратног коментара у програмском коду.

Збирка је подељена у два дела. Први део обухвата основне елементе и конструкције језика C: просте линијске структуре, грањање у програму, петље, скокови, улаз-излаз, функције, низови и матрице.

Други део је наставак градива из претходног дела, који обухвата и нешто сложеније задатке: стрингови, показивачи (обухватајући и рад са показивачима кроз све елементе језика из првог дела збирке), динамичка зона меморије, структуре и рад са датотекама.

*У Косовској Митровици,
септембар, 2011.*

Аутори

САДРЖАЈ

1	ОСНОВНЕ КОНСТРУКЦИЈЕ ПРОГРАМСКОГ ЈЕЗИКА С	1
1.1	Типови података, декларације и константе	1
1.2	Оператори	4
2	ПРОГРАМИ СА ПРОСТОМ ЛИНИЈСКОМ СТРУКТУРОМ	8
2.1	Улазна и излазна конверзија	8
2.1	Креирање програма са простом линијском структуром	15
3	ГРАЊАЊЕ У ПРОГРАМУ	27
4	FOR ПЕТЉА	40
5	WHILE ПЕТЉА	61
6	DO...WHILE ПЕТЉА	73
7	СКОКОВИ	79
7.1	BREAK	79
7.2	CONTINUE	83
7.3	GOTO	85
7.4	SWITCH ... CASE	88
8	КАРАКТЕРИ – ЗНАКОВНИ УЛАЗ И ИЗЛАЗ	94
9	ФУНКЦИЈЕ	105
9.1	Рекурзивне функције	123
10	НИЗОВИ	129
10.1	Основне конструкције програма са низовима	129
10.2	Операције са низовима и разврставање елемената	132
10.3	Низови и функције	139
10.4	Претраживање низова	145
10.5	Уређивање и сортирање низова	151
11	МАТРИЦЕ	160
	ЛИТЕРАТУРА	178

1 ОСНОВНЕ КОНСТРУКЦИЈЕ ПРОГРАМСКОГ ЈЕЗИКА **C**

1.1 Типови података, декларације и константе

*Табела 1.1: Знакови који се користе у програмском језику **C***

велика и мала слова енглеске абетеде A – Z, a – z (Ц прави разлику између великих и малих слова)															
цифре: 0 1 2 3 4 5 6 7 8 9															
бели знакови: размак (space), хоризонтални табулатор, вертикални табулатор, нови ред															
специјални знакови:															
+	-	*	/	=	%	&	#	!	?	^	"	'	~	\	
	<	>	()	[]	{	}	:	;	.	,	_		

Табела 1.2: Неки знакови који имају строго дефинисано значење

Знак	Значење
/*	Почетак коментара (коментар се мора завршити са знаком */)
*/	Крај коментара
#	Макро улази (декларација константи, команде превођења, ...)
;	Крај програмске линије
,	Набрајање
\	Специјални знак за контролу излаза
0x	Следи приказ хексадецималног броја
' '	Почетак или крај променљиве типа string
'	Почетак или крај променљиве типа char

*Табела 1.3: Резервисане речи у програмском језику **C***

auto	break	case	char	continue	const	default
do	double	else	enum	extern	float	for
goto	if	int	long	register	return	short
sizeof	static	struct	switch	signed	typedef	union
unsigned	void	volatile	while			

Табела 1.4: Типови података

ознака	значење
int	целобројни податак
char	знаковни податак
float	реални податак једноструке тачности
double	реални податак двоструке тачности
short int	"кратки" целобројни податак; у меморији заузима мање меморијског простора од податка типа int па може приказати мањи распон целих бројева.
long int	"дуги" целобројни податак; у меморији заузима више меморијског простора од податка типа int па може приказати већи и распон целих бројева.
long double	реални податак вишеструке тачности
unsigned int	неозначене (само позитивне) вредности целобројних података
unsigned short int	
unsigned long int	
void	тип податка који не садржи вредност

Декларација променљивих: `tip_podatka ime_promenljive;`
нпр: `int a;`

Иницијализација променљивих: `ime_promenljive = vrednost;`
нпр: `a=5;`

Истовремена декларација и иницијализација променљивих:
`tip_podatka ime_promenljive = vrednost;`
нпр: `int a=5;`

Дефинисање нових типова података: `typedef tip_podatka ime_tipa;`
`typedef float duzina;`

Декларација константи:
`const tip_podatka ime_promenljive = vrednost_konstante;`
нпр: `const double e=2.71828182845905;`

Дефинисање симболичке константе: `#define IME_KONSTANTE vrednost_konstante`
нпр: `#define MAX 50`

Декларација набројаних константи:
`enum ime_nabrajanja { IME_KONSTANTE = vrednost,`
`IME_KONSTANTE = vrednost, ...}`
нпр: `enum podaci {MIN=10, MAX=100, POZELJNO=30}`

1.1. Који су од следећих коментара коректно записани:

- `/*Ovo je programski jezik C.*/`
- `/*Prva oblast /*Tipovi podataka i operatori*/ */`
- `/*Prva oblast`
`Tipovi podataka i operatori*/`
- `*/Programski jezik C.*/`

- а) Коректно.
- б) Некоректно, не могу да се коментари гнезде један унутар другог.
- в) Коректно.
- г) Некоректно, коментар треба започети са `/*` и завршити са `*/`.

1.2. Који су од следећих идентификатора правилно написани:

- | | | |
|------------------|---------|----------|
| а) "x" | г) broj | е) float |
| б) ime_i_prezime | д) a21 | ж) r |
| в) a+b | ђ) 3n | |

- а) Неправилно, недозвољени знак `"`.
- б) Правилно.
- в) Неправилно, оператор није дозвољен.
- г) Прваилно.
- д) Правилно.
- ђ) Неправилно, први знак не сме да буде цифра.
- е) Неправилно, резервисане речи нису дозвољене.
- ж) Правилно.

1.3. Које су од следећих целобројних константи коректно записане:

- | | | |
|-----------|------------|----------|
| а) 543 | г) 650L | е) 03928 |
| б) 0123 | д) 0xFFFFU | ж) 3ab22 |
| в) 0XF7A5 | ђ) 152UL | з) 1.555 |

- а) Коректно, децимална константа типа **int**.
- б) Коректно, октална константа почиње са 0, типа **int**.
- в) Коректно, хексадецимална константа почиње са 0X или 0x, типа **int**.
- г) Коректно, децимална константа типа **long int**, пошто се завршава са L (или l).
- д) Коректно, хексадецимална константа типа **unsigned int**, пошто се завршава са U (или u).
- ђ) Коректно, децимална константа типа **unsigned long int**, пошто се завршава са UL (или ul).
- е) Некоректно, 8 и 9 нису окталне цифре.
- ж) Некоректно, а и b нису децималне цифре.
- з) Тачка означава реалну константу.

1.4. Које су од следећих реалних константи коректно записане:

- | | | |
|-----------|------------|-----------|
| а) 1.24 | г) 211.25F | е) 1ab.34 |
| б) .065 | д) 1.65L | ж) 423. |
| в) 1.34E2 | ђ) 5,45 | з) 2.E11 |

- а) Коректно, реална константа типа **double**.
- б) Коректно, реална константа типа **double**.
- в) Коректно, реална константа типа **double**.
- г) Коректно, реална константа типа **float**.
- д) Коректно, реална константа типа **long double**.
- ђ) Некоректно, не користи се децимални зарез.
- е) Некоректно, слова нису дозвољена.

- ж) Коректно, реална константа типа **double**.
 з) Коректно, реална константа типа **double**.

1.5. Које су од следећих декларација и иницијализација коректно записане:

- а) **int** a, b, c; г) **const double** pi 3.14; ж) **typedef float** duzina;
 б) **int** a=7; д) **#define** MAX =10 duzina a=5.2;
 в) **char** slovo='d'; ђ) **const int** a=5;

Коректно а), б), в), ђ), ж)

1.6. Декларисати константе за следеће вредности:

- а) број дана у недељи; в) сала са 200 места;
 б) особа тешка 80.5 kg; г) број дана у сваком месецу за преступну годину.

- а) **const int** nedelja=7;
 б) **const double** tezina=80.5;
 в) **const int** sala=200;
 г) **enum** meseci {JAN=31, FEB=29, MAR=31, APR=30, MAJ=31, JUN=30,
 JUL=31, AVG=31, SEP=30, OKT=31, NOV=30, DEC=31};

1.2 Оператори

Табела 1.5: Аритметички оператори и оператори доделе

оператор	пример	Значење
=	x=y	Додела
+	x+y	Сабирање
-	x-y	Одузимање
*	x*y	Множење
/	x/y	дељење (уколико се примени над целим бројевима има функцију целобројног дељења)
%	x%y	остатак при целобројном дељењу
++	++x	пре инкрементирање (x=x+1), повећа вредност променљиве x за 1 па се таква променљива користи
++	x++	пост инкрементирање (x=x+1), употреби се променљива x, па се тек онда увећа за 1
--	--x	пре декрементирање (x=x-1), смањи вредност променљиве x за 1 па се таква променљива користи
--	x--	пост декрементирање (x=x-1), употреби се променљива x, па се тек онда смањи за 1
+=	x+=y	(x = x + y) додаје вредност операнда са десне стране оператора вредности операнда са леве стране и то постаје нова вредност левог операнда
-=	x-=y	(x = x - y) одузима вредност операнда са десне стране оператора од вредности операнда са леве стране и то постаје нова вредност левог операнда

<code>*</code>	<code>x*=y</code>	($x = x * y$) вредност операнда са десне стране оператора се множи са вредношћу операнда са леве стране и то постаје нова вредност левог операнда
<code>/</code>	<code>x/=y</code>	($x = x / y$) вредност операнда са леве стране оператора се дели са вредношћу операнда са десне стране и то постаје нова вредност левог операнда
<code>%</code>	<code>x%=y</code>	($x = x \% y$) вредност операнда са леве стране оператора се дели са вредношћу операнда са десне стране и остатак тог дељења постаје нова вредност левог операнда

Табела 1.6: Релацијски оператори

оператор	пример	значење
<code>></code>	<code>x>y</code>	веће
<code><</code>	<code>x<y</code>	мање
<code>>=</code>	<code>x>=y</code>	веће или једнако
<code><=</code>	<code>x<=y</code>	мање или једнако
<code>==</code>	<code>x==y</code>	једнако
<code>!=</code>	<code>x!=y</code>	различито

Табела 1.7: Логички оператори

оператор	пример	значење
<code>&&</code>	<code>x&&у</code>	логичко И (AND)
<code> </code>	<code>x у</code>	логичко ИЛИ (OR)
<code>!</code>	<code>!x</code>	логичка негација (NOT)

Табела 1.8: Оператори над битовима

оператор	пример	значење
<code><<</code>	<code>x<<y</code>	померање у лево, померање x за y битова у лево
<code>>></code>	<code>x>>y</code>	померање у десно, померање x за y битова у десно
<code>&</code>	<code>x&y</code>	логичко И за низ битова
<code> </code>	<code>x y</code>	логичко ИЛИ за низ битова
<code>^</code>	<code>x^y</code>	искључиво логичко ИЛИ за низ битова
<code>~</code>	<code>~x</code>	комплемент
<code><<=</code>	<code>x<<=y</code>	$x=x<<y$
<code>>>=</code>	<code>x>>=y</code>	$x=x>>y$
<code>&=</code>	<code>x&=y</code>	извршава се логичко И над одговарајућим битовима операнда са леве и десне стране оператора и резултат се додељује левом операнду
<code> =</code>	<code>x =y</code>	извршава се логичко ИЛИ над одговарајућим битовима операнда са леве и десне стране оператора и резултат се додељује левом операнду

Табела 1.9: Оператори специјалне намене

оператор	значење
<code>[]</code>	индексирање
<code>()</code>	позив функције
<code>.</code>	приступ пољу (елементу) структуре
<code>-></code>	приступ пољу (елементу) структуре помоћу показивача
<code>*</code> (унарни)	приступ податку помоћу показивача (индексно адресирање)
<code>&</code> (унарни)	адреса податка
<code>?:</code>	условни оператор, $(x < y) ? x : y$
<code>,</code>	ланчање израза
<code>sizeof</code>	величина променљиве, <code>sizeof(x)</code> величина типа, <code>sizeof(int)</code>
<code>(tip)</code>	cast оператор, <code>(double) a</code>

1.7. Који је резултат после извршавања следећих аритметичких операција:

- | | | |
|------------|---------------|---------------|
| а) $2+3*4$ | г) $9\%3$ | е) $7*3/4$ |
| б) $7/4$ | д) $7/4*3$ | ж) $7.*3./4.$ |
| в) $7\%4$ | ђ) $7./4.*3.$ | |
| | | |
| а) 24 | г) 0 | е) 5 |
| б) 1 | д) 3 | ж) 5.25 |
| в) 3 | ђ) 5.25 | |

1.8. Ако је **a=5**, колике ће бити вредности променљивих **a** и **b** након израчунавања израза:

- | | | | |
|---------------|---------------|---------------|---------------|
| а) $b=++a$ | б) $b=a++$ | в) $b=--a$ | г) $b=a--$ |
| | | | |
| а) $a=6, b=6$ | б) $a=6, b=5$ | в) $a=4, b=4$ | г) $a=4, b=5$ |

1.9. Написати следеће изразе у проширеном облику:

- | | | |
|--------------|----------------|------------------|
| а) $a\%=3;$ | б) $a*=10+b;$ | в) $a+=++b+20;$ |
| | | |
| а) $a=a\%3;$ | б) $a=a*10+b;$ | в) $a=a+b+1+20;$ |

1.10. Приказати процес конверзије типа у следећим изразима:

- | | | |
|--------------------------------------------------------------------------------------------------------------|-------------|-------------|
| а) $3+4.$ | б) $5/4*3.$ | в) $3.*5/4$ |
| | | |
| а) $3+4. \rightarrow 3.+4. \rightarrow 7.$ | | |
| б) $5/4*3. \rightarrow (5/4)*3. \rightarrow 1*3. \rightarrow 1.*3. \rightarrow 3.$ | | |
| в) $3.*5/4 \rightarrow (3.*5)/4 \rightarrow (3.*5.)/4 \rightarrow 15./4 \rightarrow 15./4. \rightarrow 3.75$ | | |

1.11. Коју вредност ће имати променљива у после извршавања следећег блока наредби:

- | | | |
|----------------------------|--------------------------|-----------------------------|
| а) <code>int y;</code> | б) <code>float y;</code> | в) <code>float y;</code> |
| <code>float x=3.14;</code> | <code>y=10/4;</code> | <code>y=(float)10/4;</code> |
| <code>y=x;</code> | | |
| | | |
| а) 3 | б) 2 | в) 2.5 |

1.12. Која вредност се добија после извршавања следећих операција:

- | | | |
|-------------------------------------------------------------------------------|--------------|------------|
| а) $5>7$ | в) $8==13>5$ | д) $a<b<5$ |
| б) $10<=20$ | г) $14>5<3$ | |
| | | |
| а) $5>7 \rightarrow 0$ | | |
| б) $10<=20 \rightarrow 1$ | | |
| в) $8==13>5 \rightarrow 8==(13>5) \rightarrow 8==1 \rightarrow 0$ | | |
| г) $14>5<3 \rightarrow (14>5)<3 \rightarrow 1<3 \rightarrow 1$ | | |
| д) $a<b<5 \rightarrow (a<b)<5 \rightarrow (0 \text{ ili } 1)<5 \rightarrow 1$ | | |

1.13. Написати следеће изразе у програмском језику **C**:

- | | | |
|---------------------------------------------|----------------------------|-----------------|
| a) $(a \text{ AND } b) < (c \text{ OR } d)$ | б) $(x \text{ OR } y) = z$ | в) $a \neq b$ |
| а) $(a \ \&\& \ b) < (c \ \ d)$ | б) $(x \ \ y) == z$ | в) $a \ != \ b$ |

1.14. Које се вредности добијају након извршавања следећих операција над битовима за случај када су подаци типа **int** и имају 16 битова:

- | | | |
|-------------------------------|-----------------------------|----------------------------|
| a) $0x1234 \ \& \ 0x5678$ | г) $! \ 0x1234$ | е) $000001 \ << \ 5$ |
| б) $0x1234 \ \ 0x5678$ | д) $022222 \ \& \ 055555$ | ж) $0x3801 \ << \ 4$ |
| в) $0x1234 \ \wedge \ 0x5678$ | ђ) $022222 \ \&\& \ 055555$ | з) $0xff56 \ >> \ 4$ |
| г) $\sim \ 0x1234$ | | |
| а) $0x1230$ | г) 0 | е) 000040 |
| б) $0x567c$ | д) 000000 | ж) $0x8010$ (прекорачење!) |
| в) $0x444c$ | ђ) 1 | з) $0xffff5$ |
| г) $0xedcb$ | | |

2 ПРОГРАМИ СА ПРОСТОМ ЛИНИЈСКОМ СТРУКТУРОМ

2.1 Улазна и излазна конверзија

*Табела 2.1: Специјални карактери након backslash и пре *

карактер	Значење
<code>\n</code>	прелазак у нови ред
<code>\t</code>	хоризонтални табулатор
<code>\v</code>	вертикални табулатор
<code>\b</code>	повратак за један карактер уназад
<code>\r</code>	прелазак на почетак текућег реда
<code>\f</code>	прелазак на нову страницу
<code>\a</code>	звучни сигнал
<code>\'</code>	једноструки наводник
<code>\''</code>	двоструки наводник
<code>\\</code>	коса црта
<code>\?</code>	Упитник
<code>\ddd</code>	исписује знак са кодом ddd октално
<code>\0xdd</code>	исписује знак са кодом dd хексадецимално
<code>\0</code>	нул знак

Табела 2.2: Конверзије за унос података

знак конверзије	тип податка који се читава
<code>%c</code>	један знак (char)
<code>%i</code>	децимални, хексадецимални или октални цели број (int), зависно од облика прочитаног броја према правилима писања целобројних константи
<code>%d</code>	децимални цео број (int)
<code>%u</code>	цео број без предзнака (unsigned int)
<code>%o</code>	октални цео број (int)
<code>%x</code>	хексадецимални цео број (int)
<code>%hi, %hd, %hu, %ho, %hx</code>	кратак цео број (short int) за одговарајући децимални, хексадецимални или октални цели број

%li, %ld, %lu, %lo, %lx	дугачак цео број (long int) за одговарајући децимални, хексадецимални или октални цели број
%e, %f, %g	број типа float
%le, %lf, %lg	број типа double
%Le, %Lf, %Lg	број типа long double
%s	Стринг
%p	Показивач

Табела 2.3: Конверзије за испис података

знак конверзије	тип податка који се исписује
%c	један знак (char)
%d, %i	децимални цео број (int)
%u	цео број без предзнака (unsigned int)
%o	октални цео број без предзнака (unsigned int)
%x, %X	хексадецимални цео број без предзнака (unsigned int)
%hi, %hd, %hu, %ho, %hx, %hX	кратак цео број (short int) за одговарајући децимални, хексадецимални или октални цели број
%li, %ld, %lu, %lo, %lx, %lX	дугачак цео број (long int) за одговарајући децимални, хексадецимални или октални цели број
%e, %f, %g	број типа float
%e, %f, %g	број типа double
%Le, %Lf, %Lg	број типа long double
%s	Стринг
%p	Показивач

Табела 2.4: Допунски параметри конверзије код исписа података

знак конверзије	Значење
%-	податак ће да се равна уз леву ивицу поља ширине n знакова, тј. да се допунски знакови размака додају иза, а не испред података
%+	знак + биће исписан испред позитивног броја
%0	код нумеричких конверзија означава да, у случају равнања уз десну ивицу, број треба да се допуни нулама уместо знакова размака
% (размак)	један размак ће претходити сваком позитивном броју
%#	(уз o или x конверзију) осигурава да ће октални или хексадецимални бројеви бити исписани с водећом 0 или 0x
%#	(уз e, f или g конверзију) осигурава да ће децимална тачка бити исписана и да ће нуле на крајњој десној страни броја бити исписане

- 2.1. Саставити програм који на екрани исписује текст: Od danas programiramo u jeziku C.

```
#include <stdio.h>

main()
{
    printf("Od danas programiramo u jeziku C.");
    getche();
    return 0;
}
```

`#include<stdio.h>` значи да укључујемо стандарну улазно-излазну библиотеку података.

`main()` је почетак главног програма.

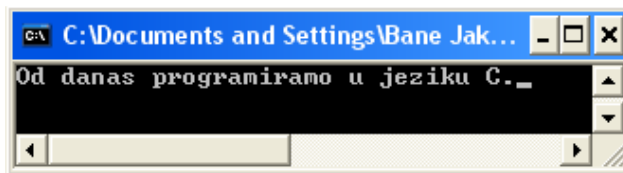
`{` означава почетак блока података.

`printf()` је функција која испишује текст на екрану, тј. конзоли. Текст који се испишује ставља се под наводницима.

`getche()` чека знак са тастатуре, а тек након тога излази из програма. Ово нам користи да видимо шта испишује наш програм.

`return 0;` враћа оперативном систему како је извршавање протекло без проблема. У случају да се врати вредност 1 ОС зна да је дошло до грешке.

`}` означава крај блока наредби. Број отворених заграда мора бити једнак броју затворених.



Испис на екрану

2.2. Шта се испишује након извршавања следећих програмских кодова:

a)

```
#include <stdio.h>

main()
{
    printf("Pozdrav svima!");
    getche();
    return 0;
}
```

б)

```
#include <stdio.h>

main()
{
    printf("\nPozdrav svima!\n");
    getche();
    return 0;
}
```

в)

```
#include <stdio.h>

main()
{
    printf("\nPozdrav\nsvima!\n");
    getche();
    return 0;
}
```

г)

```
#include <stdio.h>

main()
{
    printf("\nDobrodosli ");
    printf("u jezik C\n\n");
    getche();
    return 0;
}
```

д)

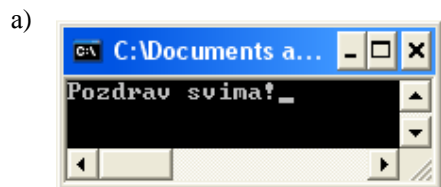
```
#include <stdio.h>

main()
{
    printf("\nDobrodosli \n\n\tu jezik C\n\n");
    getche();
    return 0;
}
```

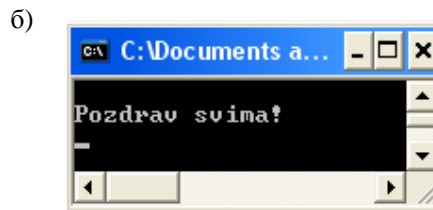
ђ)

```
#include <stdio.h>

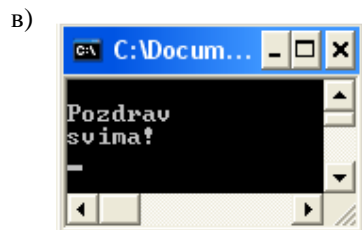
main()
{
    printf("\nABCDEF\n");
    printf("A");
    printf("BC");
    printf("DEF");
    printf("\nA\nBC\nDEF\n");
    getche();
    return 0;
}
```



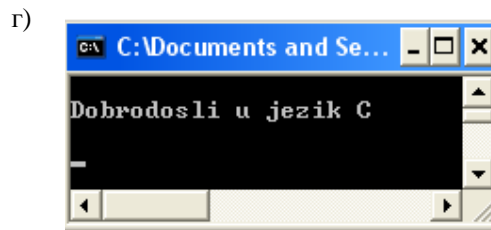
Испис на екрану



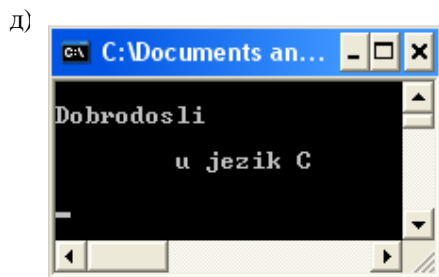
Испис на екрану



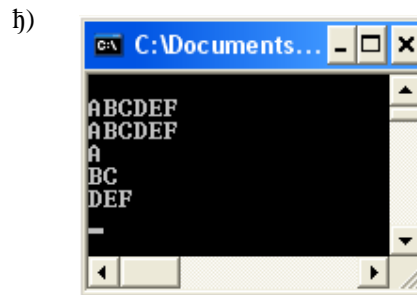
Испис на екрану



Испис на екрану



Испис на екрану



Испис на екрану

2.3. Ако су декларисане следеће променљиве

int a, b;

long int i, j;

double x, y;

char f;

написати функцију која учитава вредности са тастатуре за следеће променљиве:

а) a, b, x

б) i, f, j

в) x, y, f

г) i, f

а) `scanf("%d %d %lf", &a, &b, &x);`

б) `scanf("%ld %c %ld", &i, &f, &j);`

в) `scanf("%lf %lf %c", &x, &y, &f);`

г) `scanf("%ld %c", &i, &f);`

2.4. Написати наредбу којом се штампају:

а) променљива x у декадном систему;

б) реалне променљиве x и y;

в) променљива x у декадном, окталном и хексадецималном систему;

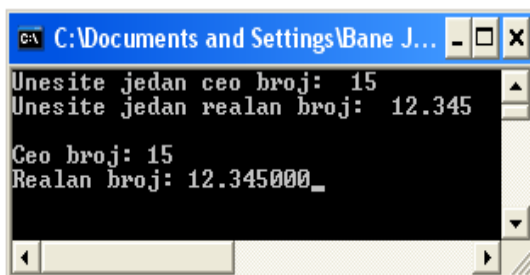
г) штампа вредност променљиве x типа char.

- a) `printf("%d", x);`
- б) `printf("%f %f", x, y);`
- в) `printf("%d %o %x", x, x, x);`
- г) `printf("%c", x);`

2.5. Саставити програм којим се учитавају и приказују један цео и један реалан број.

```
#include <stdio.h>

main()
{
    int ceo;
    float realan;
    printf("Unesite jedan ceo broj: ");
    scanf("%d",&ceo);
    printf("Unesite jedan realan broj: ");
    scanf("%f",&realan);
    printf("\nCeo broj: %d", ceo);
    printf("\nRealan broj: %f", realan);
    getche();
    return 0;
}
```

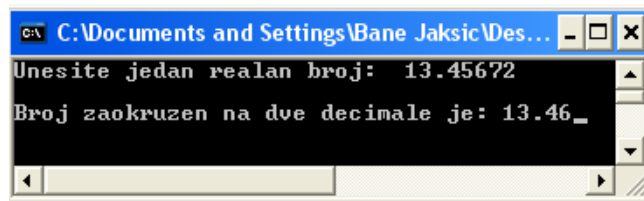


Испис на екрану

2.6. Саставити програм којим се реални број унет са тастатуре заокружује на две децимале.

```
#include <stdio.h>

main()
{
    float a;
    printf("Unesite jedan realan broj: ");
    scanf("%f",&a);
    printf("\nBroj zaokruzen na dve decimale je: %.2f", a);
    getche();
    return 0;
}
```

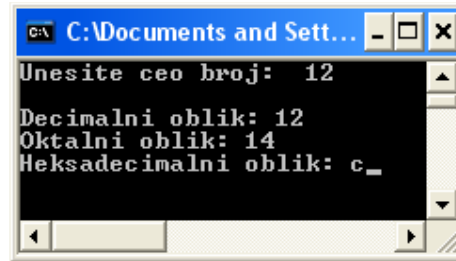


Испис на екрану

2.7. Саставити програм који омогућује унос целог броја са тастатуре и његов приказ у децималном, окталном и хексадецималном облику.

```
#include <stdio.h>

main()
{
    int a;
    printf("Unesite ceo broj: ");
    scanf("%d",&a);
    printf("\nDecimalni oblik: %d", a);
    printf("\nOktalni oblik: %o", a);
    printf("\nHeksadecimalni oblik: %x", a);
    getche();
    return 0;
}
```

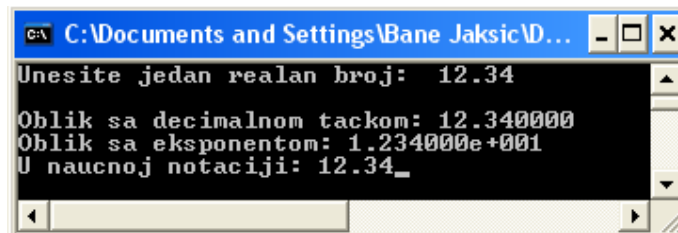


Испис на екрану

2.8. Саставити програм који омогућује унос реалног броја са тастатуре и његов приказ у различитим облицима.

```
#include <stdio.h>

main()
{
    double a;
    printf("Unesite jedan realan broj: ");
    scanf("%lf",&a);
    printf("\nOblik sa decimalnom tackom: %f", a);
    printf("\nOblik sa eksponentom: %e", a);
    printf("\nU naucnoj notaciji: %g", a);
    getche();
    return 0;
}
```



Испис на екрану

2.9. Како изгледа испис на екрану после извршавања следећег програмског кода:

```
#include <stdio.h>

main()
{
    const int a=987;
    const double b=1.2345;
    printf("%10d\n", a);
    printf("%-10d\n\n", a);
    printf("%10f\n", b);
    printf("%-10f\n", b);
    printf("%10.2f\n", b);
    printf("%.2f\n", b);
    getche();
    return 0;
}
```

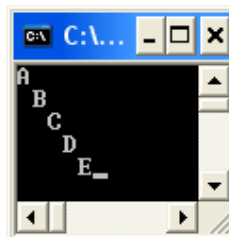


Испис на екрану

2.10. Саставити програм којим се на екрану исписује распоред слова као што је приказано на слици.

```
#include <stdio.h>

main()
{
    printf("%c\n%2c\n%3c\n%4c\n%5c", 'A', 'B', 'C', 'D', 'E');
    getche();
    return 0;
}
```

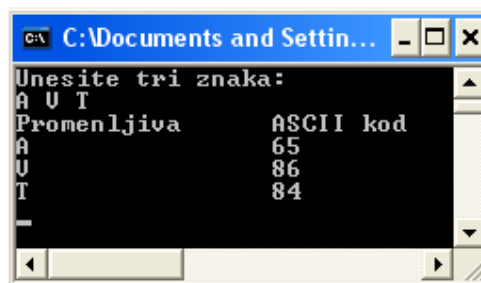


Испис на екрану

2.11. Саставити програм који омогућује унос три знаковне променљиве са тастатуре, а затим приказује њихове вредности и њихов одговарајући ASCII код.

```
#include <stdio.h>

main()
{
    char x, y, z;
    printf("Unesite tri znaka:\n");
    scanf("%c %c %c",&x,&y,&z);
    printf("Promenljiva\t ASCII kod\n");
    printf("%c\t\t %d\n",x, x);
    printf("%c\t\t %d\n",y, y);
    printf("%c\t\t %d\n",z, z);
    getche();
    return 0;
}
```

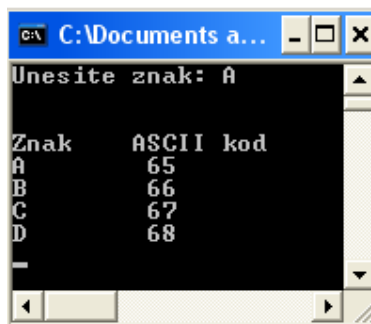


Испис на екрану

2.12. Саставити програм који омогућује унос једне знаковне променљиве, приказује њену вредност и њен ASCII код, а затим у следећа три реда приказује променљиве чији је код за један већи од претходне.

```
#include <stdio.h>

main()
{
    char x;
    char ch;
    printf("Unesite znak: ");
    scanf("%c",&x);
    printf("\n\nZnak\t ASCII kod\n");
    printf("%c \t %d \n", x, x);
    printf("%c \t %d \n", x+1, x+1);
    printf("%c \t %d \n", x+2, x+2);
    printf("%c \t %d \n", x+3, x+3);
    getche();
    return 0;
}
```

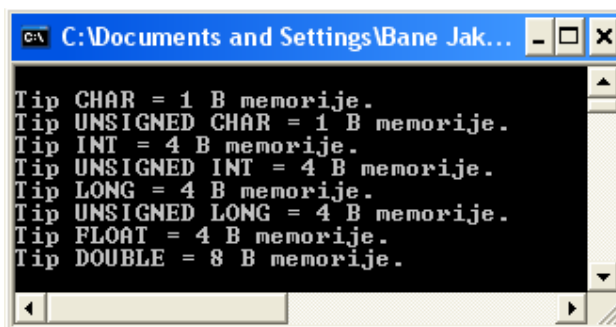


Испис на екрану

2.13. Саставити програм који приказује величину у бајтовима следећих типова података: **char**, **unsigned char**, **int**, **unsigned int**, **long**, **unsigned long**, **float**, **double**.

```
#include <stdio.h>

main()
{
    printf("\nTip CHAR = %d B memorije.", sizeof(char));
    printf("\nTip UNSIGNED CHAR = %d B memorije.", sizeof(unsigned char));
    printf("\nTip INT = %d B memorije.", sizeof(int));
    printf("\nTip UNSIGNED INT = %d B memorije.", sizeof(unsigned int));
    printf("\nTip LONG = %d B memorije.", sizeof(long));
    printf("\nTip UNSIGNED LONG = %d B memorije.", sizeof(unsigned long));
    printf("\nTip FLOAT = %d B memorije.", sizeof(float));
    printf("\nTip DOUBLE = %d B memorije.", sizeof(double));
    getch();
    return 0;
}
```



Испис на екрану

2.1 Креирање програма са простом линијском структуром

Табела 2.5: Неке од функција за рад са целобројним променљивима дефинисане у библиотеци `<stdlib.h>`

функција	вредност функције
<code>abs(n)</code>	Вредност функције типа int , једнака је апсолутној вредности аргумента n типа int .
<code>labs(n)</code>	Вредност функције типа long int , једнака је апсолутној вредности аргумента n типа long int .
<code>rand()</code>	Вредност функције, типа int , је псеудослучајан број са равномерном расподелом у опсегу <code>[0, RAND_MAX]</code> . <code>RAND_MAX</code> је симболичка константа чија се вредност мења од рачунара до рачунара, али која не сме да буде мања од 32767.
<code>srand(n)</code>	Ова функција поставља почетне вредности секвенце псеудослучајних бројева, коју даје функција <code>rand()</code> , на вредност аргумента n . Подразумевана почетна вредност секвенце је 1. Тип аргумента n је unsigned int . Функција не даје никакав резултат.

* **n** означава целобројну променљиву

Табела 2.6: Неке од математичких функција дефинисане у библиотеци <math.h>

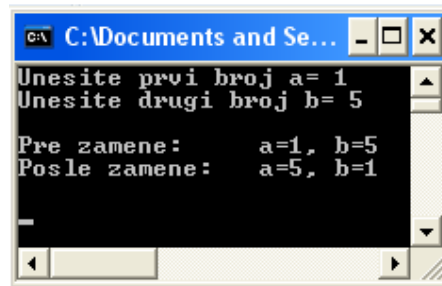
функција	вредност функције
$\sin(x)$	$\sin x$
$\cos(x)$	$\cos x$
$\tan(x)$	$\operatorname{tg} x$
$\operatorname{asin}(x)$	$\arcsin x, x \in [-1, 1]$
$\operatorname{acos}(x)$	$\arccos x, x \in [-1, 1]$
$\operatorname{atan}(x)$	$\operatorname{arc} \operatorname{tg} x, y$ опсегу $[-\pi/2, \pi/2]$
$\operatorname{atan2}(x, y)$	$\operatorname{arc} \operatorname{tg} x/y, y$ опсегу $[-\pi, \pi]$
$\sinh(x)$	$\operatorname{sh} x$
$\cosh(x)$	$\operatorname{ch} x$
$\tanh(x)$	$\operatorname{th} x$
$\exp(x)$	e^x
$\log(x)$	$\log_e x, x > 0$
$\log_{10}(x)$	$\log_{10} x, x > 0$
$\operatorname{pow}(x, y)$	x^y , ако је $x=0$, мора да буде $y > 0$; ако је $x < 0$, y мора да буде цео број
$\operatorname{sqrt}(x)$	$\sqrt{x}, x \geq 0$
$\operatorname{ceil}(x)$	вредност функције је најмања целобројна вредност која није мања од x
$\operatorname{floor}(x)$	вредност функције је највећа целобројна вредност која није већа од x
$\operatorname{fabs}(x)$	$ x $
$\operatorname{ldexp}(x, n)$	$x \cdot 2^n$

* x, y и n означавају променљиве

2.14. Саставити програм којим се замењују вредности два унета цела броја.

```
#include <stdio.h>

main()
{
    int a, b, pomocna;
    printf("Unesite prvi broj a= ");
    scanf("%d", &a);
    printf("Unesite drugi broj b= ");
    scanf("%d", &b);
    printf("\nPre zamene:\ta=%d, b=%d", a, b);
    pomocna = a;
    a = b;
    b = pomocna;
    printf("\nPosle zamene:\ta=%d, b=%d\n\n", a, b);
    getch();
    return 0;
}
```



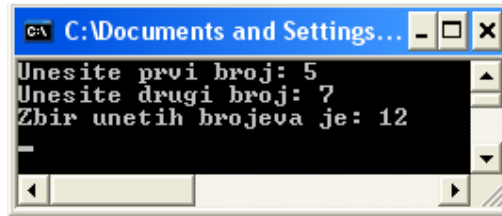
Испис на екрану

2.15. Саставити програм који учитава два цела броја са тастатуре и исписује њихов збир.

Први начин:

```
#include <stdio.h>

main()
{
    int a, b, c;
    printf("Unesite prvi broj: ");
    scanf("%d", &a);
    printf("Unesite drugi broj: ");
    scanf("%d", &b);
    c = a + b;
    printf("Zbir unetih brojeva je: %d\n", c);
    getch();
    return 0;
}
```



Испис на екрану

Други начин:

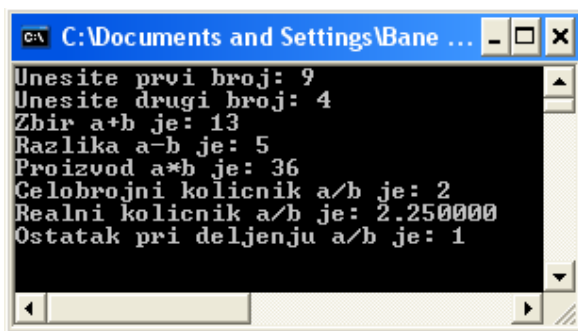
```
#include <stdio.h>

main()
{
    int a, b;
    printf("Unesite prvi broj: ");
    scanf("%d", &a);
    printf("Unesite drugi broj: ");
    scanf("%d", &b);
    printf("Zbir unetih brojeva je: %d\n", a+b);
    getch();
    return 0;
}
```

2.16. Саставити програм који учитава два цела броја и исписује њихов збир, разлику, производ, целобројни количник, реални количник и остатак при целобројном дељењу.

```
#include <stdio.h>

main()
{
    int a, b;
    printf("Unesite prvi broj: ");
    scanf("%d", &a);
    printf("Unesite drugi broj: ");
    scanf("%d", &b);
    printf("Zbir a+b je: %d\n", a+b);
    printf("Razlika a-b je: %d\n", a-b);
    printf("Proizvod a*b je: %d\n", a*b);
    printf("Celobrojni kolicnik a/b je: %d\n", a/b);
    printf("Realni kolicnik a/b je: %f\n", (float)a/(float)b);
    printf("Ostatak pri deljenju a/b je: %d\n", a%b);
    getch();
    return 0;
}
```



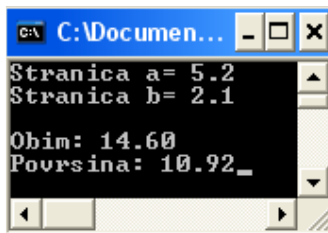
```
C:\Documents and Settings\Bane ...
Unesite prvi broj: 9
Unesite drugi broj: 4
Zbir a+b je: 13
Razlika a-b je: 5
Proizvod a*b je: 36
Celobrojni kolicnik a/b je: 2
Realni kolicnik a/b je: 2.250000
Ostatak pri deljenju a/b je: 1
```

Испис на екрану

2.17. Саставити програм који за унете странице правоугаоника исписује његов обим и површину.

```
#include <stdio.h>

main()
{
    float a, b, o, p;
    printf("Stranica a= ");
    scanf("%f", &a);
    printf("Stranica b= ");
    scanf("%f", &b);
    o=2*a+2*b;
    p=a*b;
    printf("\nObim: %.2f",o);
    printf("\nPovrsina: %.2f",p);
    getch();
    return 0;
}
```



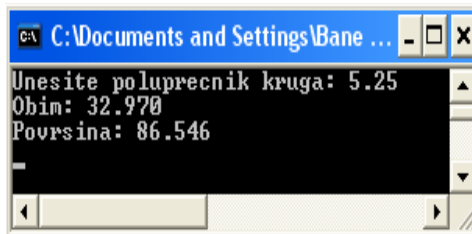
```
C:\Documen...
Stranica a= 5.2
Stranica b= 2.1
Obim: 14.60
Povrsina: 10.92
```

Испис на екрану

2.18. Саставити програм који за унети полупречник круга исписује његов обим и површину.

```
#include <stdio.h>
#define PI 3.14

main()
{
    double r;
    printf("Unesite poluprecnik kruga: ");
    scanf("%lf", &r);
    printf("Obim: %.3f\n", 2*r*PI);
    printf("Povrsina: %.3f\n", r*r*PI);
    getch();
    return 0;
}
```



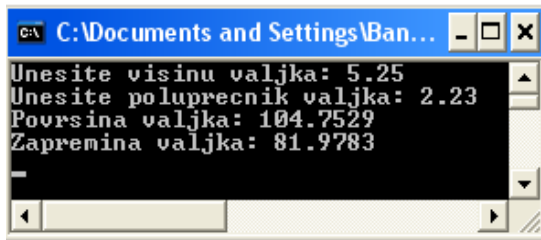
```
C:\Documents and Settings\Bane ...
Unesite poluprecnik kruga: 5.25
Obim: 32.970
Povrsina: 86.546
```

Испис на екрану

2.19. Саставити програм који за унети полупречник основице и висине ваљка испишује његову површину и запремину. (Површина: $P=2*r*\pi*(r+h)$, Запремина: $V=r^2*\pi*h$)

```
#include <stdio.h>
#define PI 3.14

main()
{
    double p,r,h,v;
    printf("Unesite visinu valjka: ");
    scanf("%lf",&h);
    printf("Unesite poluprecnik valjka: ");
    scanf("%lf",&r);
    p = 2*r*PI*(r+h);
    v = r*r*PI*h;
    printf("Povrsina valjka: %.4f\n", p);
    printf("Zapremina valjka: %.4f\n",v);
    getche();
    return 0;
}
```

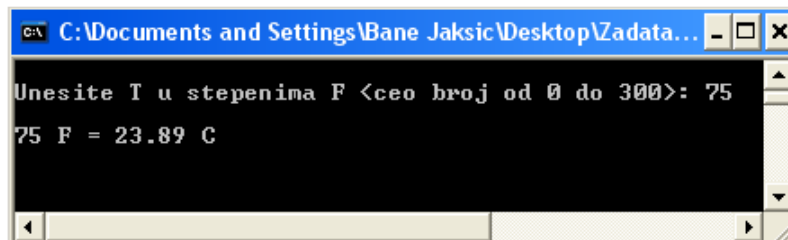


Испис на екрану

2.20. Саставити програм којим се вредност температуре унете у Фаренхајтима приказује у Целзијусима.

```
#include <stdio.h>

main()
{
    int temp;
    double fahr, celsius;
    printf("\nUnesite T u stepenima F <ceo broj od 0 do 300>: ");
    scanf("%d",&temp);
    fahr=(double)temp;
    celsius=(5.0/9.0)*(fahr-32.0);
    printf("\n%d F = %.2f C\n\n", temp, celsius);
    getche();
    return 0;
}
```



Испис на екрану

2.21. Саставити програм за решавање линеарне једначине $AX+B=0$, где се коефицијенти **A** и **B** уносе са тастатуре (**A**≠0).

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

```
main()
```

```
{
```

```
    float X, A, B;
```

```
    printf("A = ");
```

```
    scanf("%f", &A);
```

```
    printf("B = ");
```

```
    scanf("%f",&B);
```

```
    X=-B/A;
```

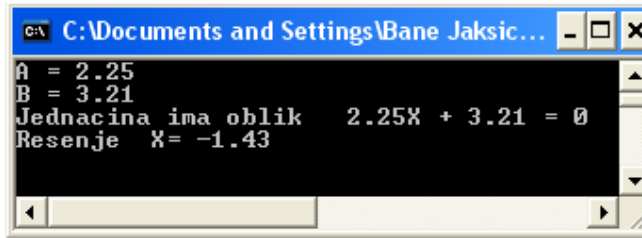
```
    printf("Jednacina ima oblik  %.2fX + %.2f = 0\n", A, B);
```

```
    printf("Resenje X= %.2f\n", X);
```

```
    getch();
```

```
    return 0;
```

```
}
```



Испис на екрану

2.22. Саставити програм за рачунање израза $y = \sqrt{x + x^2 + x^3}$ за унету вредност **x**.

Први начин:

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

```
#include <math.h>
```

```
main()
```

```
{
```

```
    float x, y;
```

```
    printf("Unesite x: ");
```

```
    scanf("%f", &x);
```

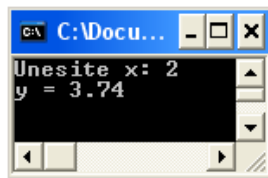
```
    y=sqrt(x+ x*x + x*x*x);
```

```
    printf("\ny = %.2f", y);
```

```
    getch();
```

```
    return 0;
```

```
}
```



Испис на екрану

Други начин:

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

```
#include <math.h>
```

```
main()
```

```
{
```

```
    float x, y;
```

```
    printf("Unesite x: ");
```

```
    scanf("%f", &x);
```

```
    y=sqrt(x+ pow(x,2) + pow(x,3));
```

```
    printf("\ny = %.2f", y);
```

```
    getch();
```

```
    return 0;
```

```
}
```

2.23. Саставити програм који исписује вредност модула комплексног броја $z = a + bi$ за унете вредности **a** и **b**. Модул се рачуна по формули $|z| = \sqrt{a^2 + b^2}$.

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

```
#include <math.h>
```

```
main()
```

```
{
```

```
    float a, b, m;
```

```
    printf("a= "); scanf("%f",&a);
```

```
    printf("b= "); scanf("%f",&b);
```

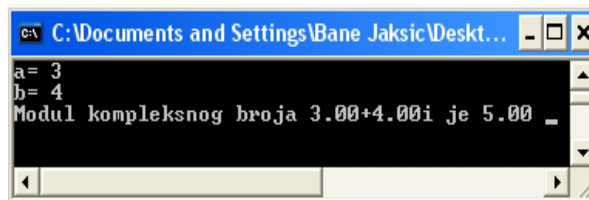
```
    m=sqrt(a*a+b*b);
```

```
    printf("Modul kompleksnog broja %.2f+%.2fi je %.2f ", a, b, m);
```

```
    getch();
```

```
    return 0;
```

```
}
```



Испис на екрану

2.24. Саставити програм који за унете реалне и имагинарне делове два комплексна броја испишује та да два броја у облику $z_1 = a + bi$ и $z_2 = c + di$, а затим рачуна њихов збир и разлику $z_1 \pm z_2 = (a + c) \pm i(b + d)$.

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

main()
{
    float a1,b1,a2,b2,rz,rr,iz,ir;
    printf("a1 = "); scanf("%f",&a1);
    printf("b1 = "); scanf("%f",&b1);
    printf("a2 = "); scanf("%f",&a2);
    printf("b2 = "); scanf("%f",&b2);
    rz=a1+a2;
    rr=a1-a2;
    iz=b1+b2;
    ir=b1-b2;
    printf("\nz1 = %.2f + %.2fi \nz2 = %.2f + %.2fi", a1, b1, a2, b2);
    printf("\nz1+z2 = %.2f + %.2fi", rz, iz);
    printf("\nz1-z2 = %.2f + %.2fi", rr, ir);
    getch();
    return 0;
}
```

```
C:\Documents an...
a1 = 5
b1 = 7
a2 = 2.11
b2 = 3.2
z1 = 5.00 + 7.00i
z2 = 2.11 + 3.20i
z1+z2 = 7.11 + 10.20i
z1-z2 = 2.89 + 3.80i
```

Испис на екрану

2.25. Саставити програм за исписивање растојања између две тачке у тродимензионалном простору на основу унетих координата тачака.

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

main()
{
    double x1, x2, y1, y2, z1, z2, d;
    printf("Unesite koordinate prve tacke <x1,y1,z1>: \n");
    scanf("%lf %lf %lf",&x1,&y1,&z1);
    printf("Unesite koordinate druge tacke <x2,y2,z2>: \n");
    scanf("%lf %lf %lf",&x2,&y2,&z2);
    d=sqrt(pow(x2-x1,2)+pow(y2-y1,2)+pow(z2-z1,2));
    printf("\nRastojanje d = %.2f\n", d);
    getch();
    return 0;
}
```

```
C:\Documents and Settings\Bane Jaksic\Desktop...
Unesite koordinate prve tacke <x1,y1,z1>:
1
2
3
Unesite koordinate druge tacke <x2,y2,z2>:
11
12
13
Rastojanje d = 17.32
```

Испис на екрану

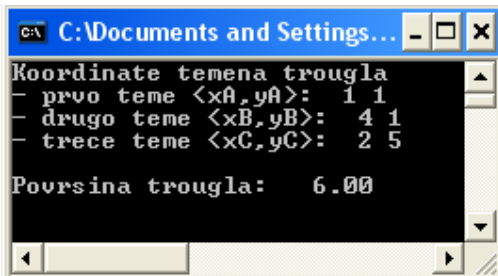
2.26. Саставити програм за исписивање површине троугла ако су задате координате његових темена. Површину троугла рачунати помоћу следећих формула:

$$a = \sqrt{(x_B - x_C)^2 - (y_B - y_C)^2} \quad , \quad b = \sqrt{(x_C - x_A)^2 - (y_C - y_A)^2} \quad , \quad c = \sqrt{(x_A - x_B)^2 - (y_A - y_B)^2}$$

$$S = \frac{a+b+c}{2} \quad , \quad P = \sqrt{S(S-a)(S-b)(S-c)}$$

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

main()
{
    double xA, yA, xB, yB, xC, yC, a, b, c, s, P;
    printf("Koordinate temena trougla\n");
    printf("- prvo teme <xA,yA>:  "); scanf("%lf%lf",&xA,&yA);
    printf("- drugo teme <xB,yB>:  "); scanf("%lf%lf",&xB,&yB);
    printf("- trece teme <xC,yC>:  "); scanf("%lf%lf",&xC,&yC);
    a=sqrt(pow(xB-xC,2)+pow(yB-yC,2));
    b=sqrt(pow(xC-xA,2)+pow(yC-yA,2));
    c=sqrt(pow(xA-xB,2)+pow(yA-yB,2));
    s=(a+b+c)/2;
    P=sqrt(s*(s-a)*(s-b)*(s-c));
    printf("\nPovrsina trougla:    %.2f\n", P);
    getch();
    return 0;
}
```

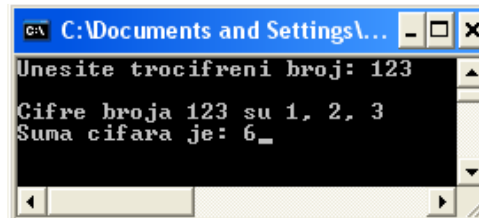


Испис на екрану

2.27. Саставити програм који за унети троцифрени број исписује његове цифре и суму цифара.

```
#include <stdio.h>

main()
{
    int xyz, x, y, z;
    printf("Unesite trocifreni broj: ");
    scanf("%d",&xyz);
    x=xyz/100;
    y=(xyz/10)%10;
    z=xyz%10;
    printf("\nCifre broja %d su %d, %d, %d", xyz,x,y,z);
    printf("\nSuma cifara je: %d", x+y+z);
    getch();
    return 0;
}
```

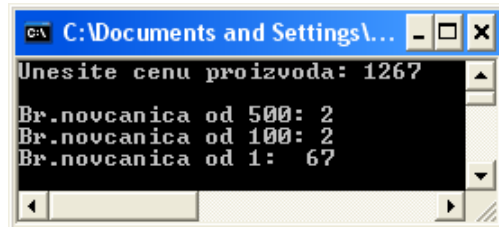


Испис на екрану

2.28. Саставити програм који учитава вредност производа у динарима, а затим израчунава и приказује колико је потребно новчаница од 500 дин., 100 дин. и 1 дин. за плаћање тог производа.

```
#include<stdio.h>

main()
{
    int n, n500, n100, n1;
    printf("Unesite cenu proizvoda: ");
    scanf("%d",&n);
    n500=n/500;
    n100=(n%500)/100;
    n1=(n%500)%100;
    printf("\nBr.novcanica od 500: %d",n500);
    printf("\nBr.novcanica od 100: %d",n100);
    printf("\nBr.novcanica od 1:  %d",n1);
    getch();
    return 0;
}
```

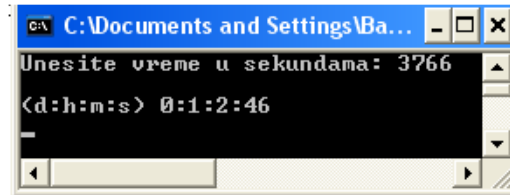


Испис на екрану

2.29. Саставити програм који за унети временски интервал у секундама и исписује га у облику дани : часови : минуте : секунде.

```
#include<stdio.h>

main()
{
    long sec, d, h, m, s;
    printf("Unesite vreme u sekundama: ");
    scanf("%ld",&sec);
    s=sec%60;
    m=sec/60;
    h=m/60;
    m=m%60;
    d=h/24;
    h=h%24;
    printf("\n(d:h:m:s) %ld:%ld:%ld:%ld\n",d,h,m,s);
    getch();
    return 0;
}
```



Испис на екрану

2.30. Саставити програм који одређује и исписује број степени, минута и секунди у углу који је задат у радијанима.

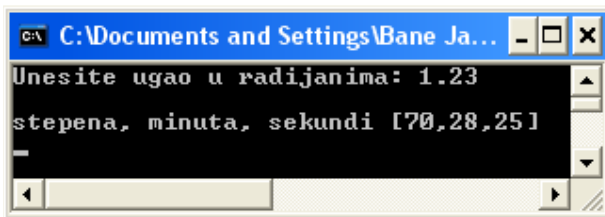
```
#include<stdio.h>
#define PI 3.141592

main()
{
    double x;
    int stepen,minut,sekund;
    printf("Unesite ugao u radijanima: ");
    scanf("%lf",&x);
    x*=180/PI; /*konverzija u stepene*/
```

```

    stepen=(int)x; /*zaokruzivanje broja stepeni*/
    x-=stepen; /*izracunavanje koliko delova stepena je preostalo*/
    x*=60; /*izracunavanje broja minuta*/
    minut=(int)x; /*zaokruzivanje broja minuta*/
    x-=minut; /*koliko delova minuta je ostalo*/
    x*=60; /*izracunavanje broja sekundi*/
    sekund=(int)x; /*zaokruzivanje broja sekundi*/
    printf("\nstepena, minuta, sekundi [%d,%d,%d]\n", stepen, minut, sekund);
    getch();
    return 0;
}

```



Испис на екрану

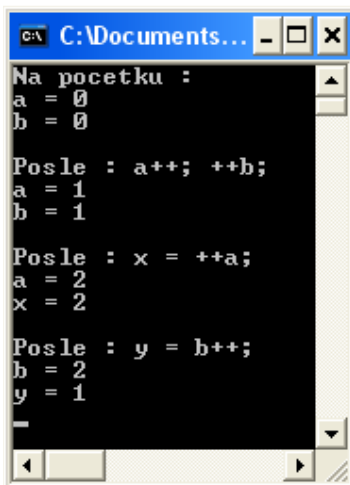
2.31. Шта се испишује на екрану након извршавања следећег програмског кода:

```

#include<stdio.h>

main()
{
    int x, y;
    int a=0, b=0;
    printf("Na pocetku : \na = %d\nb = %d\n", a, b);
    a++;
    ++b;
    printf("\nPosle : a++; ++b; \na = %d\nb = %d\n", a, b);
    x = ++a;
    y = b++;
    printf("\nPosle : x = ++a; \na = %d\nx = %d\n", a, x);
    printf("\nPosle : y = b++; \nb = %d\nny = %d\n", b, y);
    getch();
    return 0;
}

```

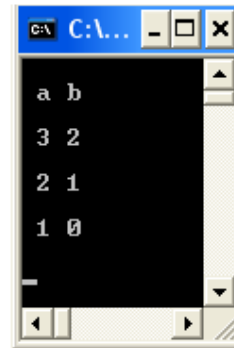


Испис на екрану

2.32. Шта се исписује на екрану након извршавања следећег програмског кода:

```
#include<stdio.h>

main()
{
    int a = 3, b = 3;
    printf("\n a b \n");
    printf("\n %d %d\n", a--, --b);
    printf("\n %d% d\n", a--, --b);
    printf("\n %d% d\n\n", a--, --b);
    getch();
    return 0;
}
```

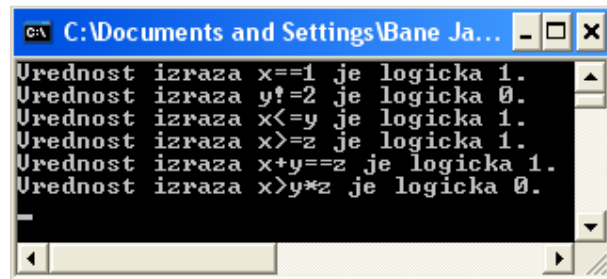


Испис на екрану

2.33. Који је резултат извршавања следећег програмског кода:

```
#include<stdio.h>

main()
{
    int x=1, y=2, z=3;
    printf("Vrednost izraza x==1 je logicka %d.\n", x==1);
    printf("Vrednost izraza y!=2 je logicka %d.\n", y!=2);
    printf("Vrednost izraza x<=y je logicka %d.\n", x<=1);
    printf("Vrednost izraza x>=z je logicka %d.\n", x>=1);
    printf("Vrednost izraza x+y==z je logicka %d.\n", x+y==z);
    printf("Vrednost izraza x>y*z je logicka %d.\n", x>y*z);
    getch();
    return 0;
}
```

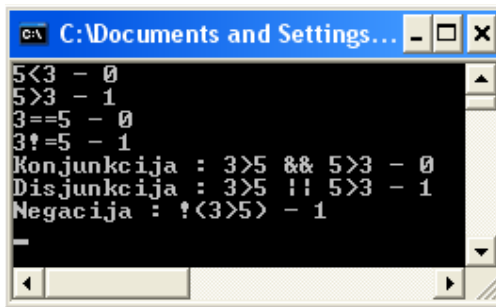


Испис на екрану

2.34. Који је резултат извршавања следећег програмског кода:

```
#include<stdio.h>

main()
{
    int a = 5<3, b = 5>3, c = 3==5, d = 3!=5;
    printf("5<3 - %d\n5>3 - %d\n3==5 - %d\n3!=5 - %d\n", a, b, c, d);
    printf("Konjunkcija : 3>5 && 5>3 - %d\n", a && b);
    printf("Disjunkcija : 3>5 || 5>3 - %d\n", a || b);
    printf("Negacija : !(3>5) - %d\n", !a);
    getch();
    return 0;
}
```



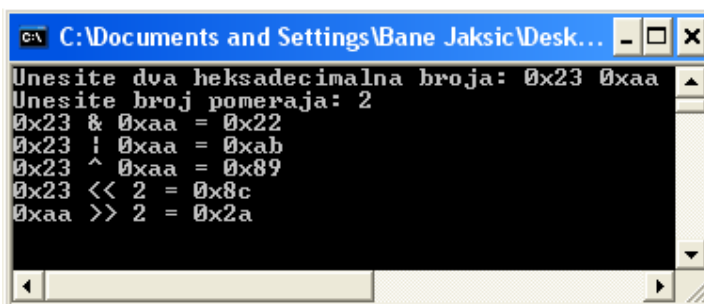
```
C:\Documents and Settings...
5<3 - 0
5>3 - 1
3==5 - 0
3!=5 - 1
Konjunkcija : 3>5 && 5>3 - 0
Disjunkcija : 3>5 || 5>3 - 1
Negacija : !(3>5) - 1
```

Испис на екрану

2.35. Саставити програм који тестира операторе над битовима за два унета броја у хексадецималном облику и броја помераја.

```
#include<stdio.h>

main()
{
    int x, y, n;
    printf("Unesite dva heksadecimalna broja: ");
    scanf("%i %i", &x, &y);
    printf("Unesite broj pomeraja: ");
    scanf("%d", &n);
    printf("%#x & %#x = %#x\n", x, y, x&y);
    printf("%#x | %#x = %#x\n", x, y, x|y);
    printf("%#x ^ %#x = %#x\n", x, y, x^y);
    printf("%#x << %d = %#x\n", x, n, x<<n);
    printf("%#x >> %d = %#x\n", y, n, y>>n);
    getch();
    return 0;
}
```



```
C:\Documents and Settings\Bane Jaksic\Desk...
Unesite dva heksadecimalna broja: 0x23 0xaa
Unesite broj pomeraja: 2
0x23 & 0xaa = 0x22
0x23 | 0xaa = 0xab
0x23 ^ 0xaa = 0x89
0x23 << 2 = 0x8c
0xaa >> 2 = 0x2a
```

Испис на екрану

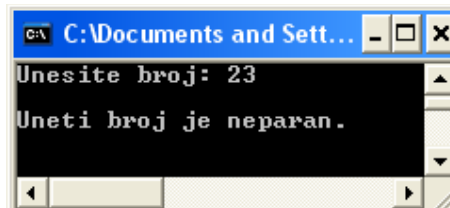
3 ГРАЊАЊЕ У ПРОГРАМУ

3.1. Саставити програм који исписује обавештење да ли је унети цео број паран или непаран.

Први начин:

```
#include <stdio.h>

main()
{
    int broj;
    printf("Unesite broj: ");
    scanf("%d", &broj);
    if (broj%2 == 0)
        printf("\nUneti broj je paran.\n");
    else
        printf("\nUneti broj je neparan.\n");
    getche();
    return 0;
}
```



Испис на екрану

Други начин (условни оператор):

```
#include <stdio.h>

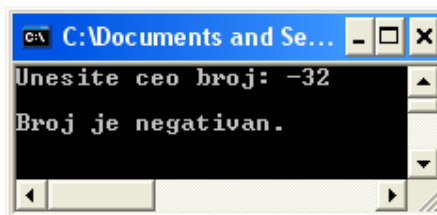
main()
{
    int broj;
    printf("Unesite broj: "); scanf("%d", &broj);
    (broj%2 == 0) ? printf("\nUneti broj je paran.\n")
                  : printf("\nUneti broj je neparan.\n");
    getche();
    return 0;
}
```

3.2. Саставити програм који исписује обавештење да ли је унети број позитиван, негативан или је једнак нули.

Први начин:

```
#include <stdio.h>

main()
{
    int a;
    printf("Unesite ceo broj: ");
    scanf("%d", &a);
    if(a < 0)
        printf("\nBroj je negativan.\n");
    else if(a > 0)
        printf("\nBroj je pozitivan.\n");
    else
        printf("\nBroj je nula.\n");
    getche();
    return 0;
}
```



Испис на екрану

Други начин (условни оператор):

```
#include <stdio.h>

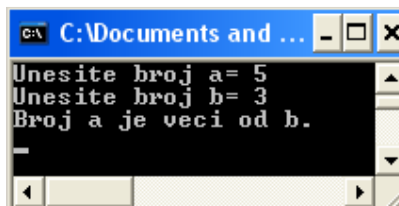
main()
{
    int a;
    printf("Unesite ceo broj: "); scanf("%d", &a);
    if(a == 0)
        printf("\nBroj je nula.\n");
    else
        (a > 0) ? printf("\nBroj je pozitivan.\n")
                : printf("\nBroj je negativan.\n");
    getche();
    return 0;
}
```

3.3. Саставити програм који за два унета цела броја исписује какав постоји релациони однос између њих (једнаки су, први већи од другог или први је мањи од другог).

Први начин:

```
#include <stdio.h>

main()
{
    int a, b;
    printf("Unesite broj a= ");
    scanf("%d", &a);
    printf("Unesite broj b= ");
    scanf("%d", &b);
    if(a == b)
        printf("Brojevi su jednaki.\n");
}
```



Испис на екрану


```
else if (a > b)
    printf("Broj a je veci od b.\n");
else
    printf("Broj a je manji od b.\n");
getche();
return 0;
}
```

Други начин (условни оператор):

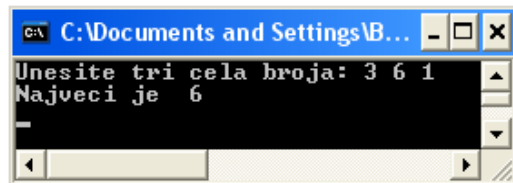
```
#include <stdio.h>

main()
{
    int a, b;
    printf("Unesite broj a= ");
    scanf("%d", &a);
    printf("Unesite broj b= ");
    scanf("%d", &b);
    if(a == b)
        printf("Brojevi su jednaki.\n");
    else
        (a > b) ? printf("Broj a je veci od b.\n")
                : printf("Broj a je manji od b.\n");
    getche();
    return 0;
}
```

3.4. Саставити програм који за три унета цела броја исписује највећи.

```
#include <stdio.h>

main()
{
    int a, b, c, max;
    printf("Unesite tri cela broja: ");
    scanf("%d%d%d", &a, &b, &c);
    max=a;
    if(b>max)
        max=b;
    if(c>max)
        max=c;
    printf("Najveci je %d\n", max);
    getche();
    return 0;
}
```

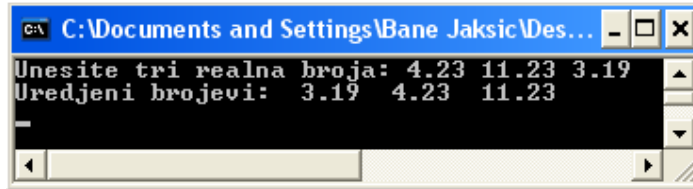


Испис на екрану

3.5. Саставити програм који три унета реална броја уређује у неоппадајућем редоследу.

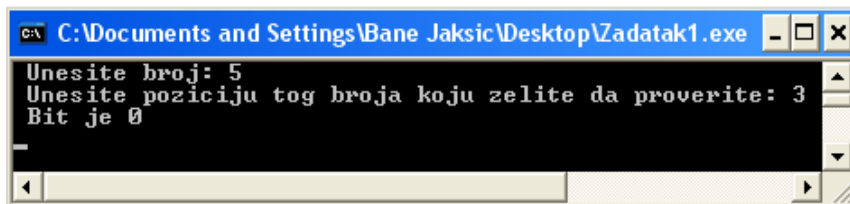
```
#include <stdio.h>

main()
{
    double x, y, z, p;
    printf("Unesite tri realna broja: ");
    scanf("%lf%lf%lf", &x, &y, &z);
    if(x>y)
    {
        p=x; x=y; y=p;
    }
    if(x>z)
    {
        p=x; x=z; z=p;
    }
    if(y>z)
    {
        p=y; y=z; z=p;
    }
    printf("Uredjeni brojevi:  %.2f  %.2f  %.2f\n", x, y, z);
    getch();
    return 0;
}
```

*Испис на екрану***3.6.** Саставити програм који проверава и исписује да ли се на **k**-том месту унетог броја **n** налази бит који има вредност 1 или 0.

```
#include <stdio.h>

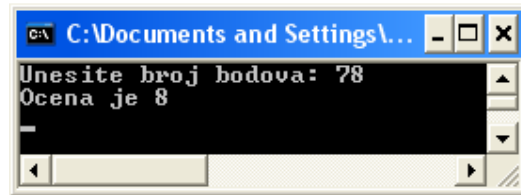
main()
{
    int n,k;
    printf(" Unesite broj: ");
    scanf("%d",&n);
    printf(" Unesite poziciju tog broja koju zelite da proverite: ");
    scanf("%d",&k);
    if((n & (1 << k))!=0)
        printf(" Bit je 1\n");
    else
        printf(" Bit je 0\n");
    getch();
    return 0;
}
```

*Испис на екрану*

3.7. Саставити програм који ће на основу унетих броја поена (од нула до 100) исписати одговарајућу оцену (0-50 пет, 51-60 шест, 61-70 седам, 71-80 осам, 81-90 девет, 91-100 десет).

```
#include <stdio.h>

main()
{
    int a;
    printf("Unesite broj bodova: ");
    scanf ("%d", &a);
    if (a>90)
        printf("Ocena je 10\n");
    else if (a>80)
        printf("Ocena je 9\n");
    else if (a>70)
        printf("Ocena je 8\n");
    else if (a>60)
        printf ("Ocena je 7\n");
    else if (a>50)
        printf("Ocena je 6\n");
    else
        printf("Ocena je 5\n");
    getche();
    return 0;
}
```



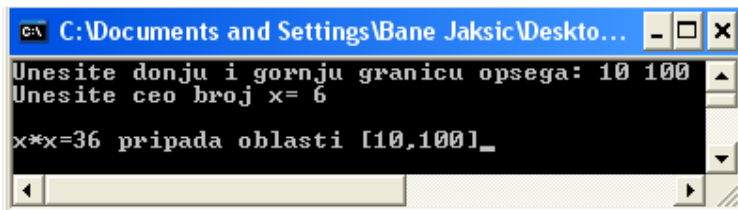
Испис на екрану

3.8. Саставити програм који ће за унети опсег позитивних целих бројева од **a** до **b** исписати да ли се у задатом опсегу налази квадрат броја **x** (број **x** се уноси са тастатуре).

Први начин:

```
#include <stdio.h>

main()
{
    int a, b, x;
    printf("Unesite donju i gornju granicu opsega: ");
    scanf("%d %d", &a, &b);
    printf("Unesite ceo broj x= ");
    scanf("%d", &x);
    if((a <= x*x) && (x*x <= b))
        printf("\nx*x=%d pripada oblasti [%d,%d]", x*x, a, b);
    else
        printf("\nx*x=%d ne pripada oblasti [%d,%d]", x*x, a, b);
    getche();
    return 0;
}
```



Испис на екрану

Други начин (условни оператор):

```
#include <stdio.h>

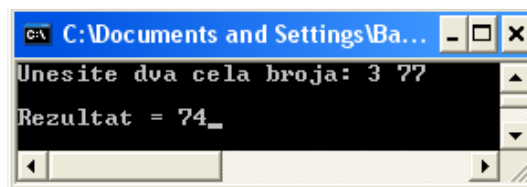
main()
{
    int a, b, x;
    printf("Unesite donju i gornju granicu opsega: ");
    scanf("%d %d", &a, &b);
    printf("Unesite ceo broj x= ");
    scanf("%d", &x);
    ((a <= x*x) && (x*x <= b))
        ? printf("\nx*x=%d pripada oblasti [%d,%d]", x*x, a, b)
        : printf("\nx*x=%d ne pripada oblasti [%d,%d]", x*x, a, b);
    getche();
    return 0;
}
```

3.9. Саставити програм који ће учитати два броја и од већег одузети мањи и приказати резултат.

Први начин:

```
#include <stdio.h>

main()
{
    int x, y;
    printf("Unesite dva cela broja: ");
    scanf("%d %d", &x, &y);
    if(x<y)
        printf("\nRezultat = %d", y-x);
    else
        printf("\nRezultat = %d", x-y);
    getche();
    return 0;
}
```



Испис на екрану

Други начин (условни оператор):

```
#include <stdio.h>

main()
{
    int x, y;
    printf("Unesite dva cela broja: ");
    scanf("%d %d", &x, &y);
    printf("\nRezultat = %d", (x<y)? y-x : x-y);
    getche();
    return 0;
}
```

3.10. Саставити програм за одређивање сигнум функције и исписивање резултата за унети реалан

$$\text{број } x. \quad y = \text{sgn}(x) = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases}$$

Први начин:

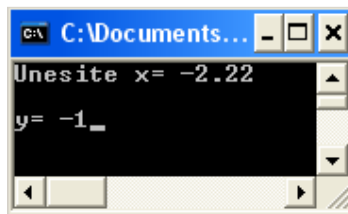
```
#include <stdio.h>

main()
{
    float x;
    int y;
    printf("Unesite x= ");
    scanf("%f", &x);
    if(x==0)
        y=0;
    else if(x>0)
        y=1;
    else
        y=-1;
    printf("\ny= %d", y);
    getch();
    return 0;
}
```

Други начин (условни оператор):

```
#include <stdio.h>

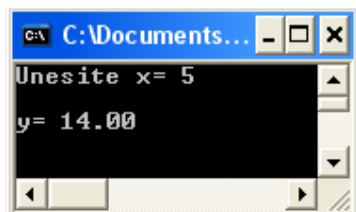
main()
{
    float x;
    int y;
    printf("Unesite x= ");
    scanf ("%f", &x);
    y = (x>0)? 1 : (x<0)? -1 : 0;
    printf("\ny= %d", y);
    getch();
    return 0;
}
```



Испис на екрану

3.11. Саставити програм за израчунавање функције y за унето x . Функција y је дефинисана на следећи начин:

$$y = \begin{cases} 2x, & -2 < x \leq 2 \\ 3x-1, & 5 \leq x < 7 \\ 1/x, & \text{остало} \end{cases}$$



Испис на екрану

```
main()
{
    float x, y;
    printf ("Unesite x= "); scanf("%f", &x);
    if((x<=2) && (x>-2))
        y=2*x;
    else if((x<7) && (x>=5))
        y=3*x-1;
    else
        y=1/x;
    printf("\ny= %.2f", y);
    getch();
    return 0;
}
```

3.12. Саставити програм за израчунавање функције z за унето x и y . Функција z је дефинисана на

$$\text{следећи начин: } z = \begin{cases} \min(x, y), & y > 0 \\ \max(x^2, y^2), & y \leq 0 \end{cases}$$

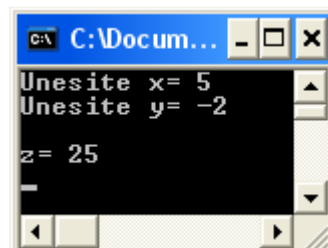
```
#include <stdio.h>

main()
{
```

```

int x, y, z;
printf("Unesite x= "); scanf("%d", &x);
printf("Unesite y= "); scanf("%d", &y);
if(y < 0)
{
    if((x*x)<(y*y)) z=y*y;
    else z=x*x;
}
else
{
    if(x<y) z=x;
    else z=y;
}
printf("\nz= %.d\n", z);
getche();
return 0;
}

```



Испис на екрану

3.13. Саставити програм који исписује обавештење да ли унете променљиве **a**, **b** и **c** које означавају дужине страница формирају троугао. Уколико формирају троугао израчунати површину троугла користећи следеће формуле:

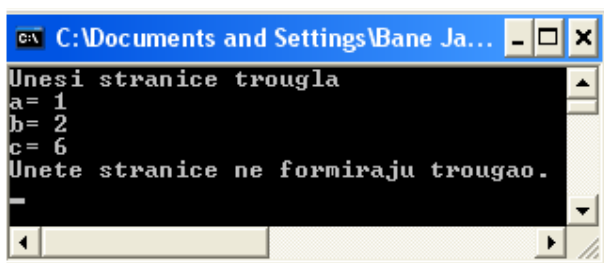
$$S = \frac{a+b+c}{2}, \quad P = \sqrt{S(S-a)(S-b)(S-c)}.$$

```

#include <stdio.h>

main()
{
    float a,b,c,p,s;
    printf("Unesi stranice trougla \na= "); scanf("%f",&a);
    printf("b= "); scanf("%f",&b);
    printf("c= "); scanf("%f",&c);
    if (a+b>c && a+c>b && b+c>a)
    {
        s=(a+b+c)/2;
        p=sqrt(s*(s-a)*(s-b)*(s-c));
        printf("Stranice formiraju trougao povrsine p= %.2f\n", p);
    }
    else printf("Stranice ne formiraju trougao.\n");
    getche();
    return 0;
}

```



Испис на екрану

3.14. Саставити програм који исписује дужину страница и величину углова троугла на основу унетих координата темена. Уколико се троугао не може формитати штампати одговарајуће обавештење.

Користити следеће формуле: $a = \sqrt{(x_B - x_C)^2 - (y_B - y_C)^2}$, $b = \sqrt{(x_C - x_A)^2 - (y_C - y_A)^2}$,
 $c = \sqrt{(x_A - x_B)^2 - (y_A - y_B)^2}$, $S = \frac{a+b+c}{2}$, $P = \sqrt{S(S-a)(S-b)(S-c)}$

```
#include <stdio.h>
#include <math.h>
#define PI 3.141592

main()
{
    double xa, ya, xb, yb, xc, yc, a, b, c, alfa, beta, gama;
    printf("Unesite koordinate tacke A(x,y): ");
    scanf("%lf%lf",&xa,&ya);
    printf("Unesite koordinate tacke B(x,y): ");
    scanf("%lf%lf",&xb,&yb);
    printf("Unesite koordinate tacke C(x,y): ");
    scanf("%lf%lf",&xc,&yc);
    a = sqrt(pow(xb-xc,2) + pow(yb-yc,2));
    b = sqrt(pow(xa-xc,2) + pow(ya-yc,2));
    c = sqrt(pow(xa-xb,2) + pow(ya-yb,2));
    printf("\nStranice:\n");
    printf("a= %f\n", a);
    printf("b= %f\n", b);
    printf("c= %f\n", c);
    if(a+b>c && a+c>b && b+c>a)
    {
        alfa = acos( (b*b+c*c-a*a) / (2*b*c) );
        beta = acos( (c*c+a*a-b*b) / (2*c*a) );
        gama = acos( (a*a+b*b-c*c) / (2*a*b) );
        alfa *= 180/PI;
        beta *= 180/PI;
        gama *= 180/PI;
        printf("\nUglovi:\n");
        printf("Alfa = %.2f stepeni\n", alfa);
        printf("Beta = %.2f stepeni\n", beta);
        printf("Gama = %.2f stepeni\n", gama);
    }
    else printf("Ne moze se kreirati trougao.\n");
    getch();
    return 0;
}
```

```

C:\Documents and Settings\Bane Ja...
Unesite koordinate tacke A(x,y): 1 2
Unesite koordinate tacke B(x,y): 3 4
Unesite koordinate tacke C(x,y): 5 8

Stranice:
a= 4.472136
b= 7.211103
c= 2.828427

Uglovi:
Alfa = 11.31 stepeni
Beta = 161.57 stepeni
Gama = 7.13 stepeni

```

Испис на екрану

3.15. Саставити програм којим се испитује да ли се секу праве $y = a_1 \cdot x + b_1$ и $y = a_2 \cdot x + b_2$. Ако се секу одредити координате пресека. Коефицијенти a_1 , b_1 , a_2 и b_2 се уносе са тастатуре

```

#include <stdio.h>

main()
{
    double a1, a2, b1, b2, x, y;
    printf("Unesite koeficijente:\na1= "); scanf("%lf", &a1);
    printf("b1= "); scanf("%lf", &b1);
    printf("a2= "); scanf("%lf", &a2);
    printf("b2= "); scanf("%lf", &b2);
    printf("\nPrave:\nny=%.2fx+%.2f\nny=%.2fx+%.2f\n", a1, b1, a2, b2);
    if(a1==a2)
    {
        if(b1==b2) printf("\nPrave su podudarne.");
        else printf("\nPrave su paralelne.");
    }
    else
    {
        x=(b2-b1)/(a1-a2);
        y=a1*x+b1;
        printf("\nTacka preseka je x= %.2f, y= %.2f\n", x, y);
    }
    getch();
    return 0;
}

```

```

C:\Documents and Settings\Bane J...
Unesite koeficijente:
a1= 1
b1= 2
a2= 3
b2= 4

Prave:
y=1.00x+2.00
y=3.00x+4.00

Tacka preseka je x= -1.00, y= 1.00

```

Испис на екрану (Задатак 15)

```

C:\Documents and Settings\Bane J...
Unesite znak [<, >]: <
Unesite koeficijente:
a= 1.23
b= 5.67

Najednacinu ima oblik: 1.23x+5.67<0
Resenje x<-4.61_

```

Испис на екрану (Задатак 16)

3.16. Саставити програм којим се решава нелинеарна једначина $a \cdot x + b \leftrightarrow 0$ где је \leftrightarrow знак < или >, и $a \neq 0$. Коefицијенти **a** и **b** и знак се уносе са тастатуре.

```
#include <stdio.h>

main()
{
    char z;
    double a, b, x;
    printf("Unesite znak [<,>]: "); scanf("%c", &z);
    printf("Unesite koeficijente:\na= "); scanf("%lf", &a);
    printf("b= "); scanf("%lf", &b);
    printf("\nNejednacina ima oblik: %.2fx+%.2f%c0\n", a, b, z);
    x=-b/a;
    if(a>0) printf("\nResenje x%c%.2f", z, x);
    else printf("\nResenje .2f%c%x", z, x);
    getche(); return 0;
}
```

3.17. Саставити програм за решавање система од две линеарне једначине: $a_1x + b_1y = c_1$ и $a_2x + b_2y = c_2$. Коefицијенти **a1**, **a2**, **b1**, **b2**, **c1** и **c2** се уносе са тастатуре. За решавање система

користити методу детерминанти: $D = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} = a_1b_2 - a_2b_1$, $D_x = \begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix} = c_1b_2 - c_2b_1$ и

$$D_y = \begin{vmatrix} a_1 & c_1 \\ a_2 & c_2 \end{vmatrix} = a_1c_2 - a_2c_1.$$

Систем има три решења:

- 1) $D \neq 0$: $x = \frac{D_x}{D}$, $y = \frac{D_y}{D}$
- 2) $D = D_x = D_y = 0$: неодређено (бесконачно решења)
- 3) остали случајеви: нема решења.

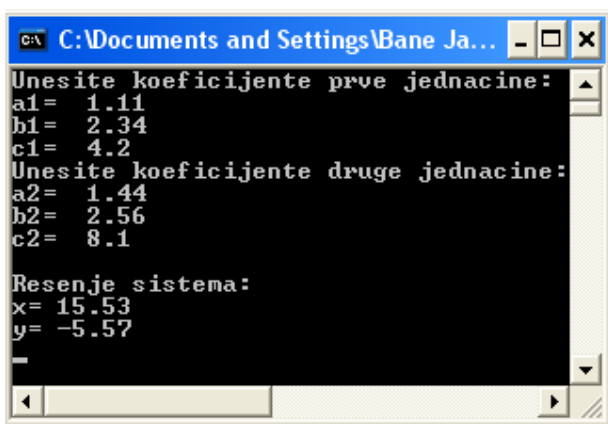
```
#include <stdio.h>

main()
{
    double a1, b1, c1, a2, b2, c2, D, Dx, Dy, x, y;
    printf ("Unesite koeficijente prve jednacine:\na1= ");
    scanf("%lf", &a1);
    printf("b1= "); scanf("%lf", &b1);
    printf("c1= "); scanf("%lf", &c1);
    printf("Unesite koeficijente druge jednacine:\na2= ");
    scanf("%lf", &a2);
    printf("b2= "); scanf("%lf", &b2);
    printf("c2= "); scanf("%lf", &c2);
    D = a1 * b2 - a2 * b1;
    Dx = c1 * b2 - c2 * b1;
    Dy = a1 * c2 - a2 * c1;
    if(D != 0)
    {
        x=Dx/D;
        y=Dy/D;
        printf ("\nResenje sistema:\nx= %.2f\n", x);
    }
}
```

```

    printf ("y= %.2f\n", y);
}
else
    if(Dx==0 && Dy==0)
        printf ("Sistem ima beskonacno resenja.\n");
    else printf ("Sistem nema resenja.\n");
getche();
return(0);
}

```



Испис на екрану

3.18. Саставити програм за решавање квадратне једначине $ax^2 + bx + c = 0$. У зависности од коефицијента a и дискриминанте $D = b^2 - 4ac$ имамо следећа решења:

- 1) $a \neq 0$ и $D > 0$: два различита и реална ($x_{1,2} = \frac{-b \pm \sqrt{d}}{2a}$),
- 2) $a \neq 0$ и $D = 0$: два једнака реална ($x_{1,2} = \frac{-b}{2a}$),
- 3) $a \neq 0$ и $D < 0$: два коњуговано комплексна ($x_{1,2} = \frac{-b \pm i\sqrt{d}}{2a}$),
- 4) $a = 0$ и $b \neq 0$: линеарна једначина, решење ($x = \frac{-c}{b}$),
- 5) $a = 0$ и $b = 0$: нема решења.

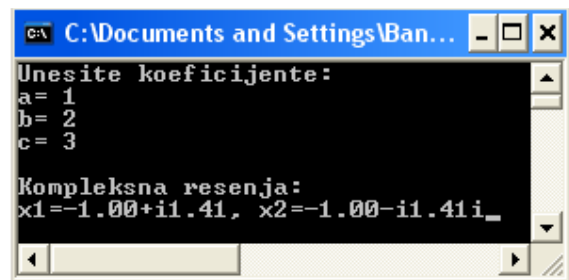
```

#include <stdio.h>
#include <math.h>

main()
{
    double a, b, c, D, x1, x2;
    printf("Unesite koeficijente:\na= "); scanf("%lf",&a);
    printf("b= "); scanf("%lf",&b);
    printf("c= "); scanf("%lf",&c);
    if(a!=0)
    {
        D=b*b-4*a*c;
        if(D>0)
        {
            x1=(-b+sqrt(D))/(2*a);
            x2=(-b-sqrt(D))/(2*a);

```

```
        printf("n\\Resenja:\\nx1=%.2f, x2=%.2f",x1,x2);
    }
    else if(D==0)
    {
        x1=(-b/(2*a));
        printf("n\\Resenje:\\nx1=x2=%.2f",x1);
    }
    else
    {
        x1=-b/(2*a);
        x2=sqrt(-D)/(2*a);
        printf("n\\Kompleksna resenja:\\n");
        printf("x1=%.2f+i%.2f, x2=%.2f-i%.2fi",x1,x2,x1,x2);
    }
}
else
{
    if(b!=0)
    {
        x1=-c/b;
        printf("n\\Resenje:\\nx=%.2f",x1);
    }
    else printf("Sistem nema resenja.");
}
getche();
return 0;
}
```



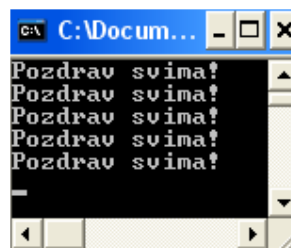
Испис на екрану

4 FOR ПЕТЉА

4.1. Саставити програм који ће пет пута исписати реченицу Pozdrav svima! употребом **FOR** петље.

```
#include <stdio.h>

main()
{
    int i;
    for(i=1; i<6; i++)
        printf("Pozdrav svima!\n");
    getch();
    return 0;
}
```



Испис на екрану

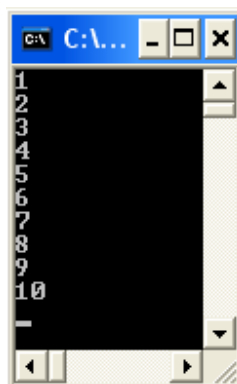
4.2. Саставити програм који употребом **FOR** петље исписује:

- а) све бројеве прве десетице,
- б) само непарне бројеве прве десетице
- в) све бројеве прве десетице у обрнутом редоследу.

а)

```
#include <stdio.h>

main()
{
    int i;
    for(i=1; i<=10; i=i+1)
        printf("%d\n", i);
    getch();
    return 0;
}
```

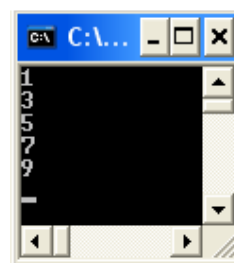


Испис на екрану а)

б)

```
#include <stdio.h>

main()
{
    int i;
    for (i=1; i<=10; i=i+2)
        printf("%d\n", i);
    getch();
    return 0;
}
```

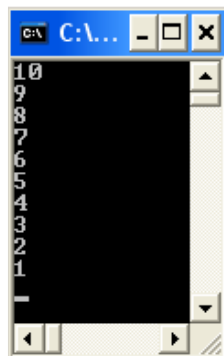


Испис на екрану б)

B)

```
#include <stdio.h>

main()
{
    int i;
    for(i=10; i>=1; i=i-1)
        printf("%d\n", i);
    getch();
    return 0;
}
```

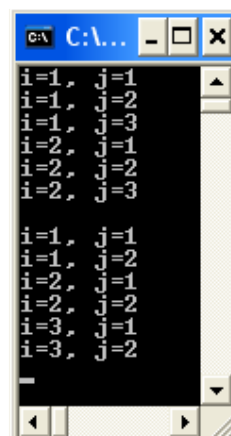


Испис на екрану в)

4.3. Које вредности **i** и **j** имају након сваке промене унутар **FOR** петље у следећем програмском коду:

```
#include <stdio.h>

main()
{
    int i, j;
    for(i=1; i<3; i++)
        for(j=1; j<4; j++)
            printf("i=%d, j=%d\n", i, j);
    printf("\n");
    for(i=1; i<4; i++)
        for(j=1; j<3; j++)
            printf("i=%d, j=%d\n", i, j);
    getch();
    return 0;
}
```

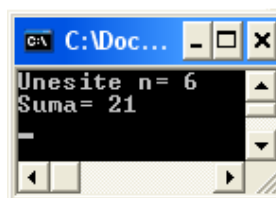


Испис на екрану

4.4. Саставити програм који за унето **n** испишује збир првих **n** целих позитивних бројева.

```
#include <stdio.h>

main()
{
    int n, s, i;
    printf("Unesite n= ");
    scanf("%d", &n);
    s=0;
    for(i=1; i<=n; i++)
        s+=i;
    printf("Suma= %d\n", s);
    getch();
    return 0;
}
```

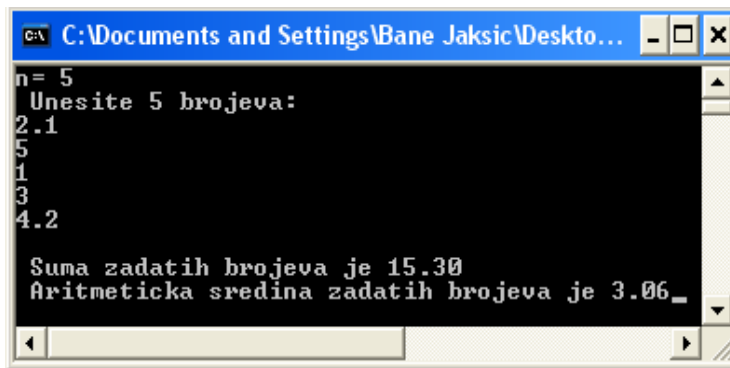


Испис на екрану

4.5. Саставити програм који за унето **n** учитава **n** реалних бројева и приказује њихов збир и аритметичку средину.

```
#include <stdio.h>

main()
{
    int n, i;
    float a, s, ars;
    s=0;
    printf("n= ");
    scanf("%d",&n);
    printf(" Unesite %d brojeva:\n", n);
    for(i=1; i<=n; i++)
    {
        scanf("%f",&a);
        s+=a;
    }
    ars=s/n;
    printf("\n Suma zadatih brojeva je %.2f", s);
    printf("\n Aritmeticka sredina zadatih brojeva je %.2f", ars);
    getch();
    return 0;
}
```

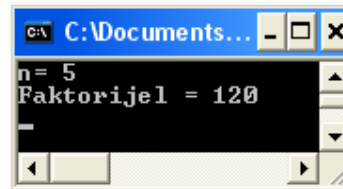


Испис на екрану

4.6. Саставити програм који за унети цео број **n** приказује његов факторијел.

```
#include <stdio.h>

main()
{
    int i,n;
    long faktorijel=1;
    printf("n= ");
    scanf("%d",&n);
    for(i=1;i<=n;i++)
        faktorijel=faktorijel*i;
    printf("Faktorijel = %ld\n", faktorijel);
    getch();
    return 0;
}
```

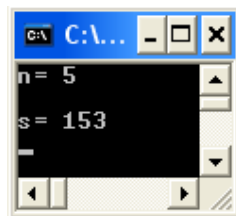


Испис на екрану

4.7. Саставити програм којим се за унети природан број n израчунава суму: $S=1!+2!+3!+\dots+n!$.

```
#include <stdio.h>

main()
{
    int i,n;
    long f=1, s=0;
    printf("n= ");
    scanf("%d",&n);
    for(i=1; i<=n; i++)
    {
        f*=i;
        s+=f;
    }
    printf("\ns= %ld\n", s);
    getche();
    return 0;
}
```



Испис на екрану

4.8. Саставити програм којим се, за дате природне бројеве m и n , израчунава израз:

a) $S = n(n+m)(n+2m)\dots(n+m \cdot m)$

б) $S = \frac{1}{n+m} - \frac{1}{n+2m} + \frac{1}{n+3m} - \dots (-1)^{m+1} \frac{1}{n+m \cdot m}$

a)

```
#include <stdio.h>

main()
{
    int i, m, n;
    long s;
    printf("n= ");
    scanf("%d",&n);
    printf("m= ");
    scanf("%d",&m);
    s=1;
    for(i=0; i<=m; i++)
        s=s*(n+i*m);
    printf("\ns= %ld\n", s);
    getche();
    return 0;
}
```

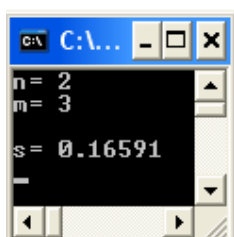


Испис на екрану а)

б)

```
#include <stdio.h>

main()
{
    int i, m, n, znak;
    float s;
    printf("n= ");
    scanf("%d",&n);
    printf("m= ");
    scanf("%d",&m);
    s=0;
    znak=1;
    for(i=1; i<=m; i++)
    {
        s=s+(float)znak/(n+i*m);
        znak=-znak;
    }
    printf("\ns= %.5f\n", s);
    getche();
    return 0;
}
```



Испис на екрану б)

4.9. Саставити програм којим се, за дати природни n израчунава израз:

$$a) S = \frac{1!}{\frac{1}{2}} + \frac{2!}{\frac{1}{2} + \frac{1}{3}} + \frac{3!}{\frac{1}{2} + \frac{1}{3} + \frac{1}{4}} + \dots + \frac{n!}{\frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n+1}}$$

$$b) S = 1 - \frac{1+2}{2!} + \frac{1+2+3}{3!} - \dots (-1)^{n-1} \frac{1+2+3+\dots+n}{n!}$$

$$B) S = \frac{\cos(1)}{\sin(1)} * \frac{\cos(1) + \cos(2)}{\sin(1) + \sin(2)} * \dots * \frac{\cos(1) + \cos(2) + \dots + \cos(n)}{\sin(1) + \sin(2) + \dots + \sin(n)}$$

```
a) #include <stdio.h>

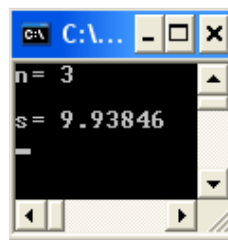
main()
{
    int i, n, fakt;
    float s, q;
    printf("n= ");
    scanf("%d", &n);
    s=0;
    q=0;
    fakt=1;
    for(i=1; i<=n; i++)
    {
        fakt=fakt*i;
        q=q+1./(1+i);
        s=s+fakt/q;
    }
    printf("\ns= %.5f\n", s);
    getche();
    return 0;
}
```

```
b) #include <stdio.h>

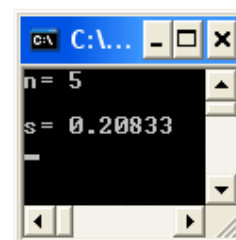
main()
{
    int i, n, fakt, znak;
    float s, q;
    printf("n= ");
    scanf("%d", &n);
    s=0;
    q=0;
    fakt=1;
    znak=1;
    for(i=1; i<=n; i++)
    {
        fakt=fakt*i;
        q=q+i;
        s=s+znak*q/(float)fakt;
        znak=-znak;
    }
    printf("\ns= %.5f\n", s);
    getche();
    return 0;
}
```

```
B) #include <stdio.h>
#include <math.h>

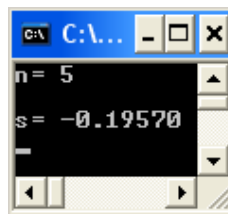
main()
{
    int i, n;
    float s, p, q;
    printf("n= ");
    scanf("%d", &n);
    s=1;
    p=0;
    q=0;
    for(i=1; i<=n; i++)
    {
        p=p+cos(i);
        q=q+sin(i);
        s=s*p/(float)q;
    }
    printf("\ns= %.5f\n", s);
    getche();
    return 0;
}
```



Испис на екрану а)



Испис на екрану б)



Испис на екрану в)

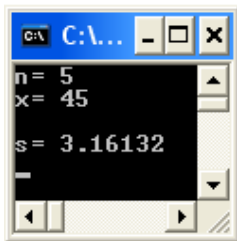
4.10. Саставити програм којим се, за дати природни број n и реалан број x , израчунава израз:

a) $S = \sin(x) + \sin^2(x) + \dots + \sin^n(x)$

б) $S = \cos(x) + \cos(x^2) + \dots + \cos(x^n)$

```
a) #include <stdio.h>
#include <math.h>

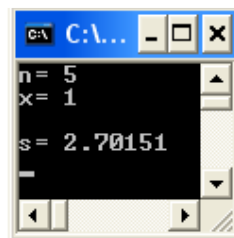
main()
{
    int i, n;
    float s, p, x;
    printf("n= ");
    scanf("%d",&n);
    printf("x= ");
    scanf("%f",&x);
    s=0;
    p=1;
    for(i=1; i<=n; i++)
    {
        p=p*sin(x);
        s=s+p;
    }
    printf("\ns= %.5f\n", s);
    getch();
    return 0;
}
```



Испис на екрану а)

```
б) #include <stdio.h>
#include <math.h>

main()
{
    int i, n;
    float s, p, x;
    printf("n= ");
    scanf("%d",&n);
    printf("x= ");
    scanf("%f",&x);
    s=0;
    p=1;
    for(i=1; i<=n; i++)
    {
        p=p*x;
        s=s+cos(p);
    }
    printf("\ns= %.5f\n", s);
    getch();
    return 0;
}
```

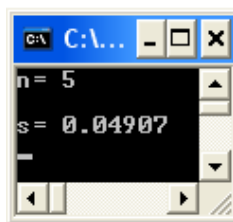


Испис на екрану б)

4.11. Саставити програм којим се, за дати природни број n , израчунава израз:

$$S = \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2+\sqrt{2}}} \cdot \frac{1}{\sqrt{2+\sqrt{2+\sqrt{2}}}} \cdot \dots \cdot \frac{1}{\sqrt{2+\sqrt{2+\dots+\sqrt{2}}}}$$

Код последњег фактора квадратни корен је примењен n пута.



Испис на екрану

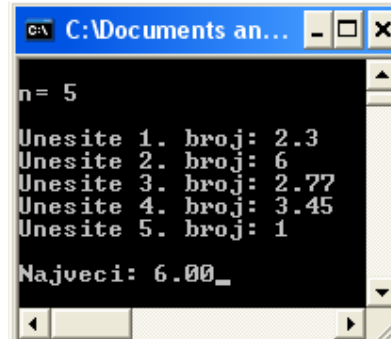
```
#include <stdio.h>
#include <math.h>

main()
{
    int i, n;
    float s, p;
    printf("n= "); scanf("%d",&n);
    s=1; p=0;
    for(i=1; i<=n; i++)
    {
        p=sqrt(p+2);
        s=s/(float)p;
    }
    printf("\ns= %.5f\n", s);
    getch();
    return 0;
}
```

4.12. Саставити програм који ће учитати **n** реалних бројева и исписати највећи.

```
#include <stdio.h>

main()
{
    int i, n;
    float br, max;
    printf("\nn= ");
    scanf("%d",&n);
    printf("\nUnesite 1. broj: ");
    scanf("%f",&br);
    max=br;
    for(i=2; i<=n; i++)
    {
        printf("Unesite %d. broj: ",i);
        scanf("%f",&br);
        if(br>max) max=br;
    }
    printf("\nNajveci: %.2f", max);
    getch();
    return 0;
}
```

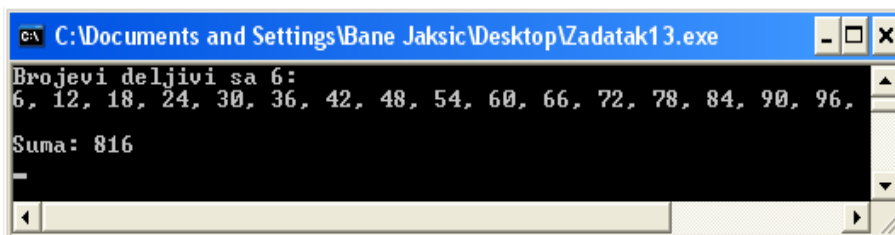


Испис на екрану

4.13. Саставити програм који ће исписати све бројева прве стотине који су дељиви са 6 и њихову суму.

```
#include <stdio.h>

main()
{
    int i, s=0;
    printf("Brojevi deljivi sa 6:\n");
    for(i=1; i<=100; i++)
    {
        if(i%6 == 0)
        {
            s=s+i;
            printf("\n%d", i);
        }
    }
    printf("\n\nSuma: %d\n", s);
    getch();
    return 0;
}
```

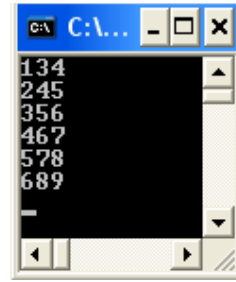


Испис на екрану

4.14. Саставити програм који исписује све троцифрене бројеве код којих је друга цифра за 2 већа од прве, а трећа за 1 већа од друге.

```
#include <stdio.h>

main()
{
    int a, b, c;
    for(a=1; a<=9; a++)
        for(b=3; b<=9; b++)
            for(c=4; c<=9; c++)
                if(b == a+2 && c == b+1)
                    printf("%d%d%d\n", a, b, c);
    getch();
    return 0;
}
```

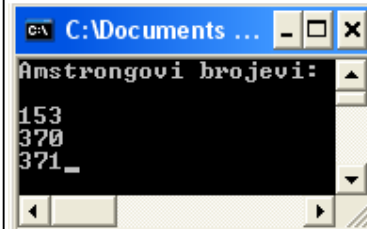


Испис на екрану

4.15. Саставити програм којим се исписују сви троцифрени Амстронгови бројеви. Троцифрени број је Амстронгов ако је једнак збиру кубова својих цифара.

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

main()
{
    int a, b, c, broj;
    printf("Amstrongovi brojevi:\n");
    for(a=1; a<=9; a++)
        for(b=0; b<9; b++)
            for(c=0; c<9; c++)
            {
                broj=100*a+10*b+c;
                if(broj == pow(a,3)+pow(b,3)+pow(c,3))
                    printf("\n%d", broj);
            }
    getch();
    return 0;
}
```



Испис на екрану

4.16. Саставити програм којим се исписују сви троцифрени бројеви ABC који имају својство $(ABC) = (AB)^2 - C^2$, где су непознате цифре $1 \leq A \leq 9$, $0 \leq B \leq 9$, $0 \leq C \leq 9$. На пример: $147 = 14^2 - 7^2$.

```
#include <stdio.h>
main()
{
    int a, b, c, broj;
    for(a=1; a<=9; a++)
        for(b=0; b<9; b++)
            for(c=0; c<9; c++)
            {
                broj=100*a+10*b+c;
                if(broj == ((10*a+b)*(10*a+b)-c*c))
                    printf("\n%d", broj);
            }
    getch(); return 0;
}
```

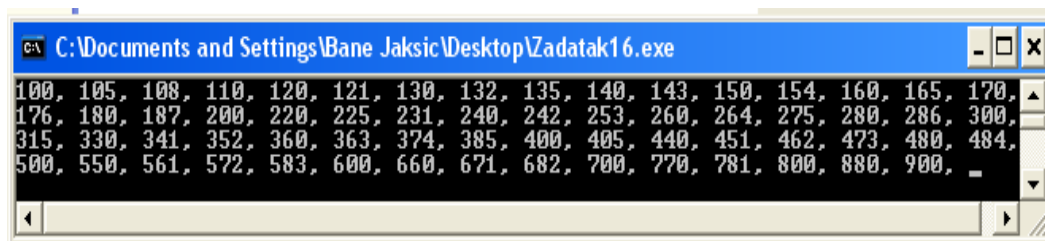


Испис на екрану

4.17. Саставити програм којим се исписују сви троцифрени бројеви који имају особину да су дељиви бројем који се добија избацивањем средње цифре.

```
#include <stdio.h>

main()
{
    int a, b, c, broj, dvocif;
    for(a=1; a<=9; a++)
        for(b=0; b<9; b++)
            for(c=0; c<9; c++)
            {
                broj=100*a+10*b+c;
                dvocif=10*a+c;
                if(broj%dvocif == 0)
                    printf("%d, ", broj);
            }
    getch();
    return 0;
}
```

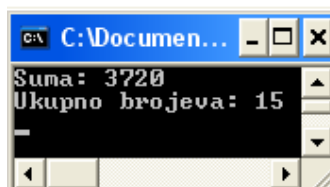


Испис на екрану

4.18. Саставити програм који рачуна суму троцифрених природних бројева чији је збир цифара једнак 5. Исписати и обавештење колико бројева има такву особину.

```
#include <stdio.h>

main()
{
    int a, b, c, s=0, br, n=0;
    for(a=1; a<=9; a++)
        for(b=0; b<9; b++)
            for(c=0; c<9; c++)
            {
                br=100*a+10*b+c;
                if (a+b+c == 5)
                {
                    n++;
                    s+=br;
                }
            }
    printf("Suma: %d\n", s);
    printf("Ukupno brojeva: %d\n", n);
    getch();
    return 0;
}
```

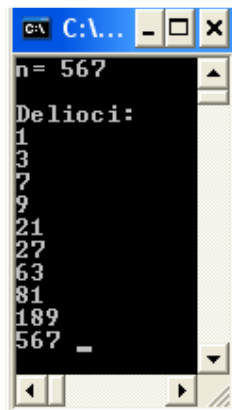


Испис на екрану

4.19. Саставити програм који ће исписати све делиоце унетог броја **n**.

```
#include <stdio.h>

main()
{
    int i, n;
    printf("n= ");
    scanf("%d",&n);
    printf("\nDelioci:");
    for(i=1; i<=n; i++)
    {
        if(n%i == 0)
            printf("\n%d ", i);
    }
    getch();
    return 0;
}
```

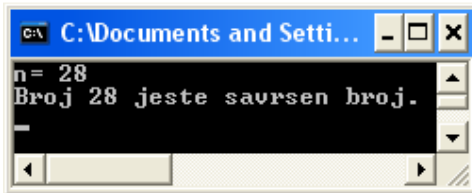


Испис на екрану

4.20. Саставити програм који ће исписати обавештење да ли је унети број **n** савршен. Број је савршен ако је једнак суми својих делиоца искључујући њега самог. На пример, $28=1+2+4+7+14$.

```
#include <stdio.h>

main()
{
    int n, i, suma=0;
    printf("n= ");
    scanf("%d",&n);
    for(i=1; i<n; i++)
    {
        if(n%i==0)
            suma+=i;
    }
    if(suma==n)
        printf("Broj %d jeste savrsen broj.\n",n);
    else
        printf("Broj %d nije savrsen broj.\n",n);
    getch();
    return 0;
}
```



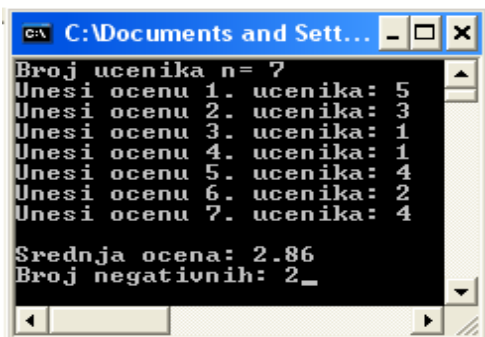
Испис на екрану

4.21. Саставити програм који ће учитати оцене за **n** ученика и исписати просечну оцену свих ученика и број ученика који имају оцену 1.

```
#include <stdio.h>

main()
{
    int i, br=0, n, ocena;
    float s,srednja;
    printf("Broj ucenika n= ");
    scanf("%d",&n);
    for(i=1; i<=n; i++)
```

```
{
    printf("Unesi ocenu %d. učenika: ", i);
    scanf("%d", &ocena);
    s+=ocena;
    if(ocena==1) br++;
}
srednja=s/n;
printf("\nSrednja ocena: %.2f", srednja);
printf("\nBroj negativnih: %d", br);
getche();
return 0;
}
```



Испис на екрану

4.22. Електронски сат показује време у сатима, минутима и секундама. Саставити програм који ће исписати све моменте времена у току једног дана такве да је сума цифара на сату једнака датом броју **n**. Испис треба да је у облику (n=33):

0 sat 11 min 31 sec

1 sat 32 min 0 sec .

```
#include <stdio.h>

main()
{
    int sat, min, sec, n;
    printf("n= ");
    scanf("%d", &n);
    for(sat=0; sat<=23; sat++)
        for(min=0; min<=59; min++)
            for(sec=0; sec<=59; sec++)
            {
                if(n==(sat/10+sat%10+min/10+min%10+sec/10+sec%10))
                    printf("\n%d sat %d min %d sec", sat, min, sec);
            }
    getche();
    return 0;
}
```

```

C:\Documents ... - [ ] X
n= 2
0 sat 0 min 2 sec
0 sat 0 min 11 sec
0 sat 0 min 20 sec
0 sat 1 min 1 sec
0 sat 1 min 10 sec
0 sat 2 min 0 sec
0 sat 10 min 1 sec
0 sat 10 min 10 sec
0 sat 11 min 0 sec
0 sat 20 min 0 sec
1 sat 0 min 1 sec
1 sat 0 min 10 sec
1 sat 1 min 0 sec
1 sat 10 min 0 sec
2 sat 0 min 0 sec
10 sat 0 min 1 sec
10 sat 0 min 10 sec
10 sat 1 min 0 sec
10 sat 10 min 0 sec
11 sat 0 min 0 sec
20 sat 0 min 0 sec_

```

Истис на екрану (Задатак 22)

```

C:\Documents a... - [ ] X
xmin= 5
xmax= 7
dx= 0.2

      x          y
=====
5.000    0.458
5.200    0.438
5.400    0.419
5.600    0.402
5.800    0.386
6.000    0.371
6.200    0.358
6.400    0.345
6.600    0.334
6.800    0.323

```

Истис на екрану (Задатак 23)

4.23. Саставити програм за табелирање функције $y = \frac{2x+1}{x^2-1}$ у опсегу од **xmin** до **xmax** са кораком **dx**.

```

#include <stdio.h>

main()
{
    double xmin, xmax, dx, x, y;
    printf("xmin= ");
    scanf("%lf", &xmin);
    printf("xmax= ");
    scanf("%lf", &xmax);
    printf("dx= ");
    scanf("%lf", &dx);
    printf("\n      x          y\n =====\n");
    for(x=xmin; x<=xmax; x+=dx)
    {
        y=(2*x+1)/(x*x-1);
        printf ("%10.3f%10.3f\n", x, y);
    }
    getch();
    return 0;
}

```

4.24. Саставити програм за табелирање функције $y = (1+x) \cdot (1+x^2) \cdot \dots \cdot (1+x^n)$ у опсегу од **xmin** до **xmax** са кораком **dx**. Степен функције **n** се уноси са тастатуре.

```
#include <stdio.h>

main()
{
    double xmin, xmax, dx, x, y=1, p=1;
    int i, n;
    printf("n= ");
    scanf("%d", &n);
    printf("xmin= ");
    scanf("%lf", &xmin);
    printf("xmax= ");
    scanf("%lf", &xmax);
    printf("dx= ");
    scanf("%lf", &dx);
    printf("\n      x          y\n");
    printf("=====\n");
    for(x=xmin; x<=xmax; x+=dx)
    {
        for(i=1; i<=n; i++)
        {
            p*=x;
            y*=(1+p);
        }
        printf("%10.6f  %11.6f\n", x, y);
        p=1;
        y=1;
    }
    getch();
    return 0;
}
```

```
n= 5
xmin= 0
xmax= 1.5
dx= 0.1

      x          y
=====
0.000000    1.000000
0.100000    1.112233
0.200000    1.260400
0.300000    1.470612
0.400000    1.790318
0.500000    2.311249
0.600000    3.221360
0.700000    4.927615
0.800000    8.353288
0.900000   15.661908
1.000000   32.000000
1.100000   69.588509
1.200000  157.007551
1.300000  359.469485
1.400000  821.350557
```

Испис на екрану

4.25. Саставити програм којим се за све углове од 0 до 90 степени са кораком промене (дефинисаним у степенима) израчунава и испишује вредност синусне функције. Функција \sin као улазни параметар захтева угао у радијанима.

```
#include <stdio.h>
#include <math.h>
#define PI 3.141592


main()
{
    double dx=0, UgaoRad, UgaoSte;
    printf("Unesite korak u stepenima: ");
    scanf("%lf", &dx);
    printf("\n      x          sin(x)\n");
    printf("\n=====\n");
    for(UgaoSte=0; UgaoSte<=90; UgaoSte+=dx)
    {
        UgaoRad = (PI*UgaoSte)/180.0;
        printf("%f  %f\n", UgaoSte, sin(UgaoRad));
    }
    getch();
    return 0;
}
```

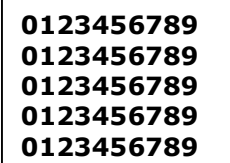
```
Unesite korak u stepenima: 5

      x          sin(x)
=====
0.000000    0.000000
5.000000    0.087156
10.000000   0.173648
15.000000   0.258819
20.000000   0.342020
25.000000   0.422618
30.000000   0.500000
35.000000   0.573576
40.000000   0.642787
45.000000   0.707107
50.000000   0.766044
55.000000   0.819152
60.000000   0.866025
65.000000   0.906308
70.000000   0.939693
75.000000   0.965926
80.000000   0.984808
85.000000   0.996195
90.000000   1.000000
```

Испис на екрану

4.26. Саставити програм који за унети позитиван цео број **n** исцртава облик приказан на слици, на слици је **n=5**.

a) 

б) 

a)

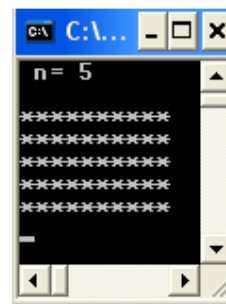
```
#include <stdio.h>

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<2*n; j++)
            printf("*");
        printf("\n");
    }
    getch();
    return 0;
}
```

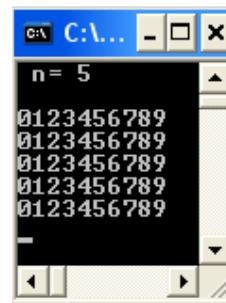
б)

```
#include <stdio.h>

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<2*n; j++)
            printf("%d", j);
        printf("\n");
    }
    getch();
    return 0;
}
```



Испис на екрану а)

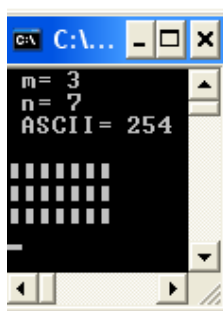


Испис на екрану б)

4.27. Саставити програм који за унете позитивне целе бројеве **m** и **n** исцртава правоугаоник формиран од знака чији се ASCII код уноси са тастатуре.

```
#include <stdio.h>

main()
{
    int i, j, m, n, a;
    printf(" m= ");
    scanf("%d", &m);
    printf(" n= ");
    scanf("%d", &n);
    printf(" ASCII= ");
    scanf("%d", &a);
    printf("\n");
    for(i=0; i<m; i++)
    {
        for(j=0; j<n; j++)
            printf("%c", a);
        printf("\n");
    }
    getch();
    return 0;
}
```



Испис на екрану

4.28. Саставити програм који за унети позитиван цео број **n** исцртава облик приказан на слици, на слици је **n=5**.

а)	б)	в)	г)
<pre> * * * * * * * * * * * * * * * * * * * * * * * * * </pre>	<pre> * * * * * * * * * * * * * * * * * * </pre>	<pre> * * * * * * * * * * * * * * * </pre>	<pre> * * * * * * * * * * * * </pre>
д)	ђ)	е)	ж)
<pre> * * * * * * * * * * * * * * * </pre>	<pre> * * * * * * * * * * * * * * </pre>	<pre> * * * * * * * * * * * * * * * </pre>	<pre> * * * * * * * * * * * * * * * </pre>

а) `#include <stdio.h>`

```

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        for(j=1; j<=n; j++)
            printf("* ");
        printf("\n");
    }
    getch();
    return 0;
}

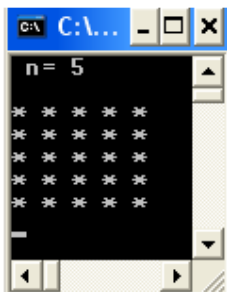
```

в) `#include <stdio.h>`

```

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        for(j=1; j<=i; j++)
            printf("* ");
        printf("\n");
    }
    getch();
    return 0;
}

```



Испис на екрану а)



Испис на екрану б)



Испис на екрану в)



Испис на екрану г)

```

6) #include <stdio.h>

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        for(j=1; j<=n; j++)
        {
            if(i==1 || i==n)
                printf("* ");
            else if (j==1 || j==n)
                printf("* ");
            else printf("  ");
        }
        printf("\n");
    }
    getche();
    return 0;
}

```

```

г) #include <stdio.h>

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        for(j=1; j<=i; j++)
        {
            if(i==n)
                printf("* ");
            else if (j==1 || j==i)
                printf("* ");
            else printf("  ");
        }
        printf("\n");
    }
    getche();
    return 0;
}

```

```

д) #include <stdio.h>

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        for(j=n; j>=i; j--)
            printf("* ");
        printf("\n");
    }
    getche();
    return 0;
}

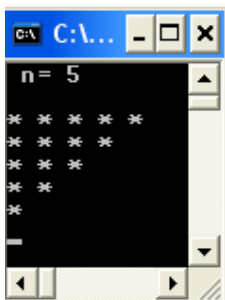
```

```

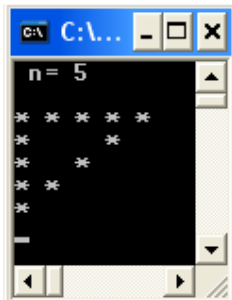
ђ) #include <stdio.h>

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        for(j=n; j>=i; j--)
        {
            if(i==1) printf("* ");
            else if(j==n || j==i)
                printf("* ");
            else printf("  ");
        }
        printf("\n");
    }
    getche();
    return 0;
}

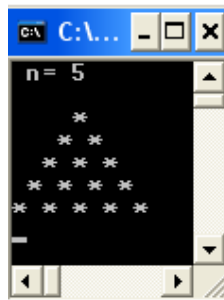
```



Испис на екрану д)



Испис на екрану ѓ)



Испис на екрану е)



Испис на екрану ж)

е)

```
#include <stdio.h>

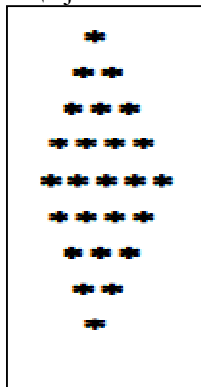
main()
{
    int i, j, n, k;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        for(k=n; k>i; k--)
            printf(" ");
        for(j=1; j<=i; j++)
            printf("* ");
        printf("\n");
    }
    getche();
    return 0;
}
```

ж)

```
#include <stdio.h>

main()
{
    int i, j, n, k;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=n; i>=1; i--)
    {
        for(k=n; k>i; k--)
            printf(" ");
        for(j=1; j<=i; j++)
            printf("* ");
        printf("\n");
    }
    getche();
    return 0;
}
```

4.29. Саставити програм који за унети позитивни цео број **n** исцртава облик приказан на слици, на слици је **n=5**.



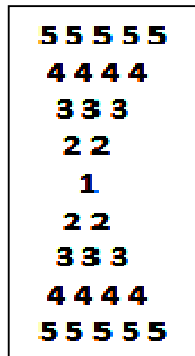
```
#include <stdio.h>

main()
{
    int n, i, j, k;
    printf("n= "); scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        for(k=n; k>i; k--)
            printf(" ");
        for(j=0; j<i; j++)
            printf("* ");
        printf("\n");
    }
    for(i=n-1; i>0; i--)
    {
        for(k=n; k>i; k--)
            printf(" ");
        for(j=0; j<i; j++)
            printf("* ");
        printf("\n");
    }
    getche(); return 0;
}
```



Испис на екрану

4.30. Саставити програм који за унети позитиван цео број **n** (од 1 до 9) исцртава облик приказан на слици, на слици је **n=5**.



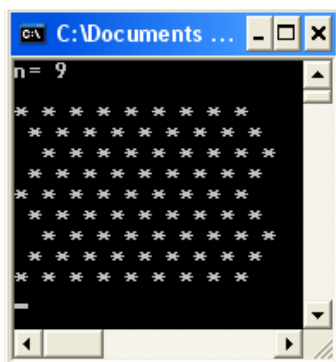
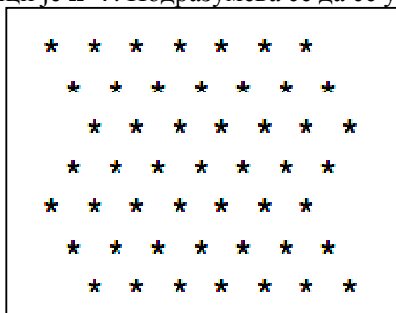
```
#include <stdio.h>

main()
{
    int n, i, j, k, m;
    printf("n= "); scanf("%d",&n);
    printf("\n");
    for(i=n; i>0; i--)
    {
        m=i;
        for(k=n; k>i; k--)
            printf(" ");
        for(j=0; j<i; j++)
            printf("%d ", m);
        printf("\n");
    }
    for(i=2; i<=n; i++)
    {
        m=i;
        for(k=n; k>i; k--)
            printf(" ");
        for(j=0; j<i; j++)
            printf("%d ", m);
        printf("\n");
    }
    getch();
    return 0;
}
```



Испис на екрану

4.31. Саставити програм који за унети цео позитиван број **n** исцртава облик приказан на слици, на слици је **n=7**. Подразумева се да се уноси непаран број.

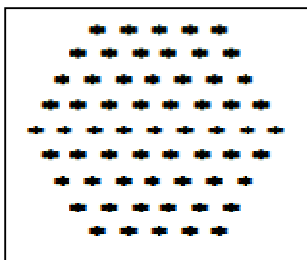


Испис на екрану

```
#include <stdio.h>

main()
{
    int i, j, n;
    printf("n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        if(i%2==0) printf(" ");
        if(i%4==3) printf(" ");
        for(j=1; j<=n; j++)
            printf("* ");
        printf("\n");
    }
    getch();
    return 0;
}
```

4.32. Саставити програм који за унети позитиван цео број n исцртава облик приказан на слици, на слици је $n=5$.

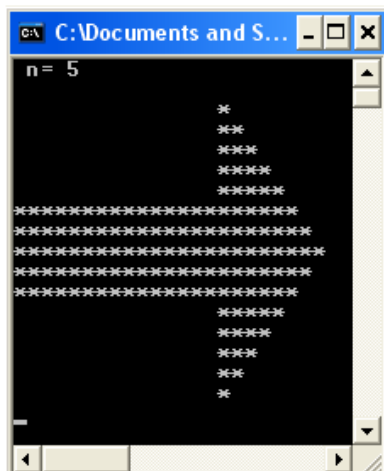
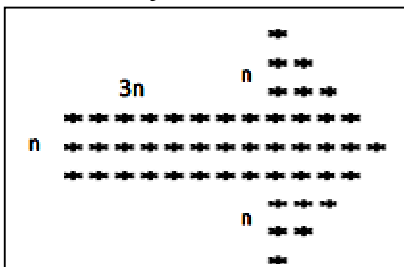


Испит на екрану

```
#include <stdio.h>

main()
{
    int n, i, j, nzv, nrz, kor=1;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    nzv=n,
    nrz=n-1;
    for(i=1; i<=2*n-1; i++)
    {
        for(j=1; j<=nrz; j++)
            printf(" ");
        for(j=1; j<=nzv; j++)
            printf("* ");
        printf("\n");
        if(i==n) kor = -1;
        nzv += kor;
        nrz -= kor;
    }
    getche();
    return 0;
}
```

4.33. Саставити програм који за унето непарно и позитивно n исцртава стрелицу приказану на слици, на слици је $n=3$.

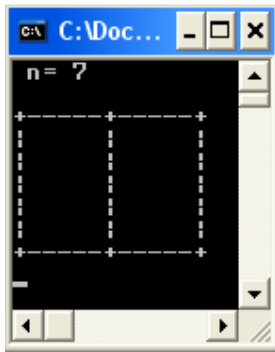
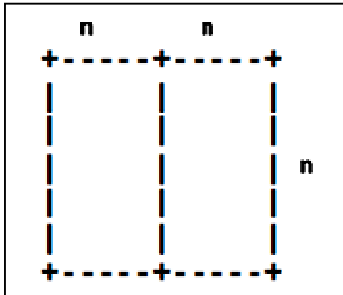


Испит на екрану

```
#include <stdio.h>

main()
{
    int n, i, j, nzv=0, kor=1;
    char rep = ' ';
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=1; i<=3*n; i++)
    {
        nzv+=kor;
        for(j=1; j<=3*n; j++)
            printf("%c", rep);
        for(j=1; j<=nzv; j++)
            printf("*");
        printf("\n");
        if(i==n) rep='*';
        if(i==3*n/2+1) kor=-1;
        if(i==2*n) rep = ' ';
    }
    getche();
    return 0;
}
```

4.34. Саставити програм који за унети цео позитиван и непаран број n исцртава облик приказан на слици, на слици је $n=7$. Укупна ширина је $2n-1$.

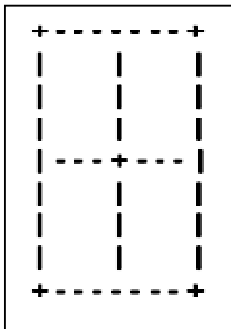


Испис на екрану

```
#include <stdio.h>

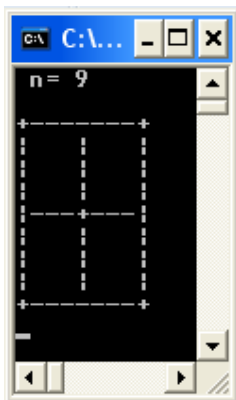
main()
{
    int i, j, n;
    printf(" n= "); scanf("%d", &n);
    printf("\n");
    for(i=0; i<2*n-1; i++)
    {
        if(i==0 || i==n-1 || i==2*n-2)
            printf("+");
        else printf("-");
    }
    printf("\n");
    for(j=0; j<n-2; j++)
    {
        for(i=0; i<2*n-1; i++)
        {
            if(i==0 || i==n-1 || i==2*n-2)
                printf("|");
            else printf(" ");
        }
        printf("\n");
    }
    for(i=0; i<2*n-1; i++)
    {
        if(i==0 || i==n-1 || i==2*n-2)
            printf("+");
        else printf("-");
    }
    printf("\n");
    getch();
    return 0;
}
```

4.35. Саставити програм који за унети цео позитиван и непаран број n исцртава облик приказан на слици, на слици је $n=9$.



```
#include <stdio.h>

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    printf("+");
    for(i=0; i<n-2; i++)
        printf("-");
    printf("+\n");
    for(i=1; i<n-1; i++)
    {
        printf("|");
        for(j=1; j<n-1; j++)
            if(i==n/2 && j==n/2) printf("+");
    }
}
```



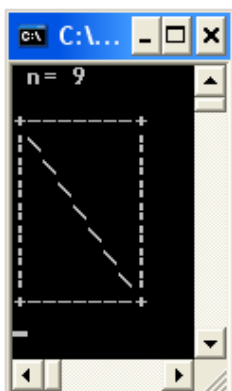
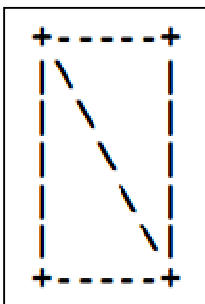
Испис на екрану

```

        else if(j==n/2 ) printf("|");
        else if(i==n/2 ) printf("-");
        else printf(" ");
        printf("|\\n");
    }
    printf("+");
    for(i=0; i<n-2; i++)
        printf("-");
    printf("+\\n");
    getch();
    return 0;
}

```

4.36. Саставити програм који за унети цео позитиван и непаран број **n** исцртава облик приказан на слици, на слици је **n=7**.



Испис на екрану

```

#include <stdio.h>

main()
{
    int i, j, n;
    printf(" n= ");
    scanf("%d", &n);
    printf("\\n");

    printf("+");
    for(i=0; i<n-2; i++)
        printf("-");
    printf ("+\\n");

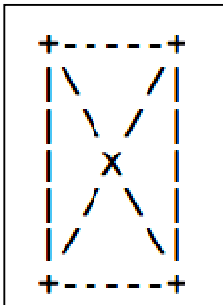
    for(j=0; j<n-2; j++)
    {
        printf("|");
        for(i=0; i<n-2; i++)
        {
            if(i==j) printf("\\\\");
            else printf(" ");
        }
        printf("|\\n");
    }

    printf ("");
    for(i=0; i<n-2; i++)
        printf ("-");
    printf ("");
    printf ("");

    getch();
    return 0;
}

```


4.37. Саставити програм који за унети цео позитиван и непаран број **n** исцртава облик приказан на слици, на слици је **n=7**.



Испис на екрану

```
#include <stdio.h>

main()
{
    int n, i, j;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");

    printf("+");
    for(i=0; i<n-2; i++)
        printf("-");
    printf("+\n");

    for(i=1; i<n-1; i++)
    {
        printf("|");
        for(j=1; j<n-1; j++)
        {
            if(i==n/2 && j==n/2) printf("X");
            else if(i==j) printf("\\");
            else if(i==n-1-j) printf("/");
            else printf(" ");
        }
        printf("|\n");
    }

    printf("+");
    for(i=0; i<n-2; i++)
        printf("-");

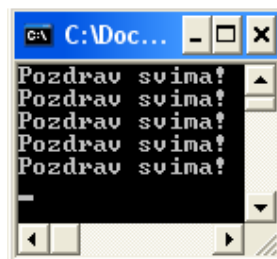
    printf("+\n");
    getch();
    return 0;
}
```

5 WHILE ПЕТЉА

5.1. Саставити програм који ће пет пута исписати реченицу `Pozdrav svima!` употребом **WHILE** петље.

```
#include <stdio.h>

main()
{
    int i;
    i=1;
    while(i<=5)
    {
        printf("Pozdrav svima!\n");
        i++;
    }
    getch();
    return 0;
}
```



Испис на екрану

5.2. Саставити програм који употребом **WHILE** петље исписује:

- а) све бројеве прве десетице,
- б) само парне бројеве прве десетице
- в) све бројеве прве десетице у обрнутом редоследу.

а)

```
#include <stdio.h>

main()
{
    int i;
    i=1;
    while(i<=10)
    {
        printf("%d\n", i);
        i++;
    }
    getch();
    return 0;
}
```

б)

```
#include <stdio.h>

main()
{
    int i;
    i=2;
    while(i<=10)
    {
        printf("%d\n", i);
        i+=2;
    }
    getch();
    return 0;
}
```



Испис на екрану а)

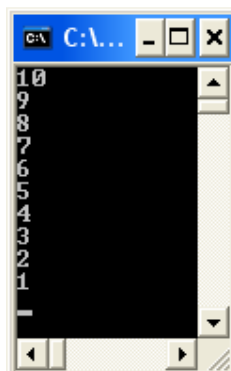


Испис на екрану б)

В)

```
#include <stdio.h>

main()
{
    int i;
    i=10;
    while(i<=10 & i>0)
    {
        printf("%d\n", i);
        i--;
    }
    getch();
    return 0;
}
```

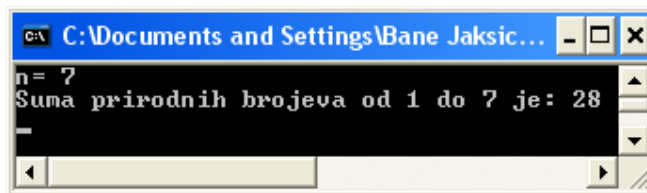


Испис на екрану в)

5.3. Саставити програм за исписивање суме природних бројева од 1 до **n**. Број **n** се уноси са тастатуре.

```
#include <stdio.h>

main()
{
    int n, i=1, suma=0;
    printf("n= ");
    scanf("%d",&n);
    while(i <= n)
    {
        suma+= i;
        i++;
    }
    printf("Suma prirodnih brojeva od 1 do %d je: %d\n", n, suma);
    getch();
    return 0;
}
```



Испис на екрану

5.4. Саставити програм за исписивање суме сваког трећег природног броја од 1 до **n**. Број **n** се уноси са тастатуре.

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

```
main()
```

```
{
```

```
    int i=1, suma=0, n;
```

```
    printf(" n= ");
```

```
    scanf("%d",&n);
```

```
    while(i<=n)
```

```
    {
```

```
        suma=suma+i;
```

```
        i=i+3;
```

```
    }
```

```
    printf(" Suma svakog treceg broja, od 1 do %d, je %d\n", n,suma);
```

```
    getch();
```

```
    return 0;
```

```
}
```



Испис на екрану

5.5. Саставити програм којим се за дати природни број **n** израчунава сума:

$$S = \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \dots + \frac{1}{(2n+1)^2}$$

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

```
#include <math.h>
```

```
main()
```

```
{
```

```
    int i=1, n;
```

```
    float s=0;
```

```
    printf(" n= "); scanf("%d", &n);
```

```
    while(i<=n)
```

```
    {
```

```
        s+=1./pow(2*i+1,2);
```

```
        i++;
```

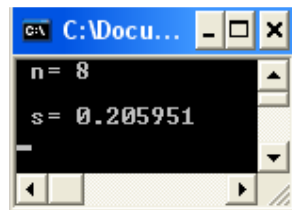
```
    }
```

```
    printf("\n s= %f\n", s);
```

```
    getch();
```

```
    return 0;
```

```
}
```



Испис на екрану

5.6. Саставити програм за израчунавање суме **s** квадрата парних и кубова непарних природних бројева од **n** до **m** (**n<m**).

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

```
#include <math.h>
```

```
main()
```

```
{
```

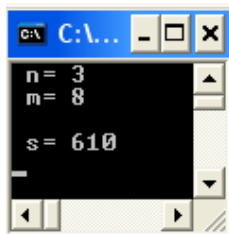
```
    int i, n, m;
```

```
    long s=0;
```

```

printf(" n= "); scanf("%d", &n);
printf(" m= "); scanf("%d", &m);
i=n;
while(i<=m)
{
    if(i%2==0) s=s+pow(i,2);
    else s=s+pow(i,3);
    i++;
}
printf("\n s= %ld\n", s);
getche();
return 0;
}

```



Испис на екрану

5.7. Саставити програм којим се:

- а) исписује **n** елемената Фибоначијевог низа;
 б) израчунава и исписује сума првих **n** елемената Фибоначијевог низа.
 Фибоначијев низ: $f_1=1, f_2=1, f_i=f_{i-1}+f_{i-2}, i=3, 4, 5, \dots$

а) #include <stdio.h>

```

main()
{
    int i=3, n, fpp=1, fp=1, fn;
    printf(" n= ");
    scanf("%d", &n);
    printf("\n f1= 1\n f2= 1\n");
    while(i<=n)
    {
        fn=fp+fpp;
        fpp=fp;
        fp=fn;
        printf(" f%d= %d\n", i, fn);
        i++;
    }
    getche();
    return 0;
}

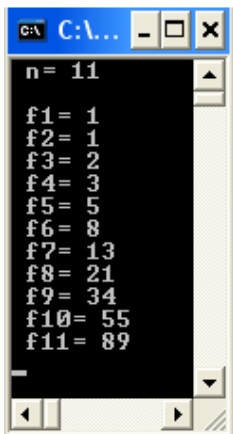
```

б) #include <stdio.h>

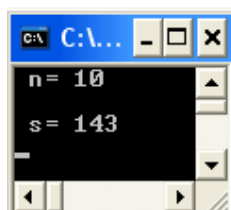
```

main()
{
    int i=3, n, fpp=1, fp=1, fn, s=2;
    printf(" n= ");
    scanf("%d", &n);
    while(i<=n)
    {
        fn=fp+fpp;
        s=s+fn;
        fpp=fp;
        fp=fn;
        i++;
    }
    printf("\n s= %d\n", s);
    getche();
    return 0;
}

```



Испис на екрану а)

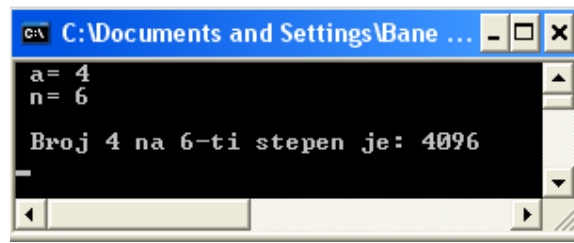


Испис на екрану б)

5.8. Саставити програм за исписивање **n**-тог степена броја **a**. Оба броја се уносе са тастатуре.

```
#include <stdio.h>

main()
{
    int a, n, i=1, stepen=1;
    printf(" a= ");
    scanf("%d",&a);
    printf(" n= ");
    scanf("%d",&n);
    while(i<=n)
    {
        stepen*=a;
        i++;
    }
    printf("\n Broj %d na %d-ti stepen je: %d\n",a,n,stepen);
    getch();
    return 0;
}
```

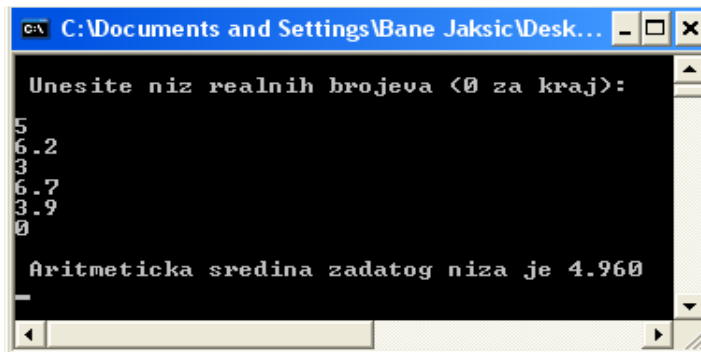


Испис на екрану

5.9. Саставити програм за израчунавање средње вредности унетих реалних бројева. Користити број 0 као **STOP** код за крај учитавања.

```
#include <stdio.h>
#define STOP 0

main()
{
    int n=0;
    float x, suma=0;
    printf("\n Unesite niz realnih brojeva (0 za kraj):\n\n");
    scanf("%f",&x);
    while(x!=STOP)
    {
        suma += x;
        n++;
        scanf("%f",&x);
    }
    if (n==0)
        printf("\n Nije zadat niz realnih brojeva.\n\n");
    else
        printf("\n Aritmeticka sredina zadatog niza je %.3f\n",suma/n);
    getch();
    return 0;
}
```

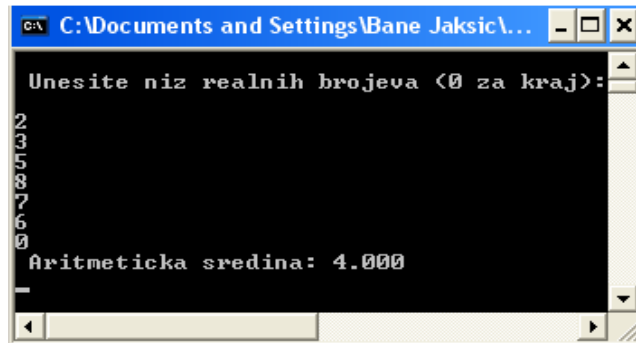


Испис на екрану

5.10. Саставити програм који ће учитавати један за другим низ бројева. Крај уноса означен је нулом. Наћи и исписати аритметичку средину учитаних бројева узимајући у обзир само оне бројеве који су већи или једнаки 2 и мањи или једнаки 6.

```
#include <stdio.h>

main()
{
    int i=0;
    float x, suma=0, sredina;
    printf("\n Unesite niz realnih brojeva (0 za kraj):\n\n");
    while(1)
    {
        scanf("%f",&x);
        if(x==0)
        {
            sredina=suma/i;
            printf(" Aritmeticka sredina: %.3f\n", sredina);
        }
        if(x>=2 && x<=6)
        {
            suma+=x;
            i++;
        }
    }
    getch();
    return 0;
}
```



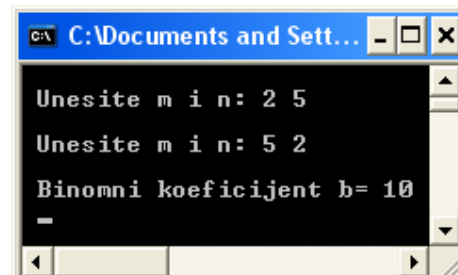
Испис на екрану

5.11. Саставити програм који ће учитати природне бројеве **m** и **n** (**m>n**). Уколико услови приликом уноса нису испуњени поновити учитавање. Наћи и исписати биномни коефицијент:

$$b = \binom{m}{n} \frac{m!}{n!(m-n)!}$$

```
#include <stdio.h>

main()
{
    int m=0, n, i, brojilac=1, imenilac=1;
    while(m<1 || n<1 || n>m)
    {
        printf("\n Unesite m i n: ");
        scanf("%d%d",&m,&n);
    }
    for(i=m;i>m-n;i--)
        brojilac=brojilac*i;
    for(i=1;i<=n;i++)
        imenilac=imenilac*i;
    brojilac=brojilac/imenilac;
    printf("\n Binomni koeficijent b= %d \n ", brojilac);
    getch();
    return 0;
}
```

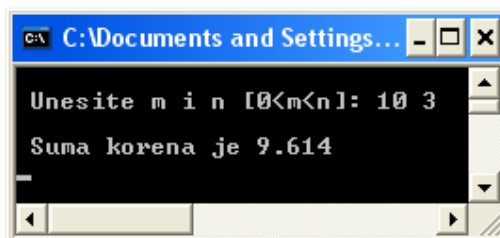


Испис на екрану

5.12. Саставити програм који ће учитати два броја **m** и **n**. Оба броја треба да буду природна. Ако тај услов није испуњен, учитавање треба поновити. Ако је **n < m**, заменити **m** са **n**. Наћи и исписати суму квадратних корена свих непарних бројева од **m** до **n**.

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

main()
{
    int m=0, n, pomoc, i;
    float suma=0, koren;
    printf("\n Unesite m i n [0<m<n]: ");
    while(m<1 || n<1)
        scanf("%d%d", &m, &n);
    if(m>n)
    {
        pomoc=n;
        n=m;
        m=pomoc;
    }
    if(m%2==0) m++;
    for(i=m; i<=n; i=i+2)
    {
        koren=sqrt(i);
        suma+=koren;
    }
    printf("\n Suma korena je %.3f\n", suma);
    getch();
    return 0;
}
```

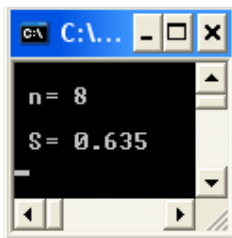


Испис на екрану

5.13. Саставити програм којим се за унето **n** рачуна сума: $S = 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \dots + (-1)^{n-1} \frac{1}{n}$

```
#include <stdio.h>

main()
{
    int znak=1, i=1, n;
    float suma=0;
    printf("\n n= ");
    scanf("%d", &n);
    while(i<=n)
    {
        suma+=(float)znak/i;
        i++;
        znak=-znak;
    }
    printf("\n S= %.3f\n", suma);
    getch();
    return 0;
}
```



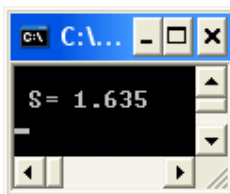
Испис на екрану

5.14. Саставити програм којим се приближно рачуна сума: $S = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \dots = \sum_{n=1}^{\infty} \frac{1}{n^2}$

Сумирање вршити све док је општи члан већи од 10^{-4} .

```
#include <stdio.h>

main()
{
    int n=1;
    float suma=0, clan;
    clan=1/(n*n);
    while(clan>1e-4)
    {
        suma=suma+clan;
        n++;
        clan=1.0/(n*n);
    }
    printf("\n S= %.3f\n", suma);
    getche();
    return 0;
}
```



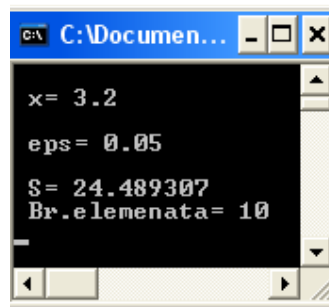
Испис на екрану

5.15. Саставити програм којим за унто x приближно рачуна сума: $S = 1 - \frac{x}{1!} + \frac{x^2}{2!} - \dots + (-1)^n \frac{x^n}{n!} + \dots$

Сумирање вршити до члана (укључујући и њега) који је по апсолутној вредности мањи од задатог броја **eps**. Исписати и колико је елемената сумирано.

```
#include <stdio.h>

main()
{
    double suma=1.0, x, eps, clan;
    int n=1, i;
    printf("\n x= ");
    scanf("%lf",&x);
    printf("\n eps= ");
    scanf("%lf",&eps);
    clan=x;
    while(abs(clan)>=eps)
    {
        suma+=clan;
        clan=x;
        n++;
        for(i=2;i<=n;i++)
            clan=-clan*(-x/i);
    }
    printf("\n S= %lf", suma);
    printf("\n Br.elemenata= %d\n", n);
    getche();
    return 0;
}
```



Испис на екрану

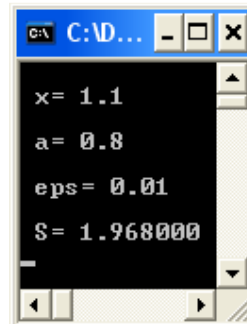
5.16. Саставити програм којим се за унето **x** приближно рачуна сума:

$$S = 1 + ax + \frac{a(a-1)}{2!}x^2 + \dots + \frac{a(a-1)\dots(a-n+1)}{n!}x^n + \dots$$

где је **a** реалан број (уноси се са тастатуре). Сумирати до члана који је по апсолутној вредности мањи од задатог броја **eps**.

```
#include <stdio.h>

main()
{
    double suma=1.0, x, a, eps, clan;
    int i=0;
    printf("\n x= ");
    scanf("%lf",&x);
    printf("\n a= ");
    scanf("%lf",&a);
    printf("\n eps= ");
    scanf("%lf",&eps);
    clan=x;
    while(fabs(clan)>=eps)
    {
        i++;
        clan=clan*(a-i+1)*x/i;
        suma=suma+clan;
    }
    printf("\n S= %lf\n", suma);
    getch();
    return 0;
}
```



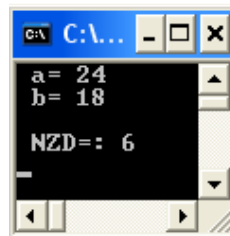
Испис на екрану

5.17. Саставити програм којим се исписује највећи заједнички делилац (NZD) бројева **a** и **b** помоћу Еуклидовог алгоритма:

- ако је **a=b**, тада је NZD=a и то је крај алгоритма;
- ако је **a≠b**, тада од већег броја одузимамо мањи и враћамо се на први корак.

```
#include <stdio.h>

main()
{
    int a, b;
    printf(" a= ");
    scanf("%d",&a);
    printf(" b= ");
    scanf("%d", &b);
    while(a!=b)
    {
        if(a>b) a-=b;
        else b-=a;
    }
    printf("\n NZD=: %d\n", a);
    getch();
    return 0;
}
```

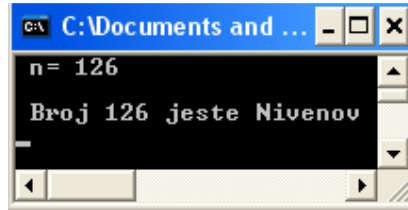


Испис на екрану

5.18. Саставити програм који испитује да ли је унети број Нивенов. Нивенов број је број који је дељив са сумом својих цифара.

```
#include <stdio.h>

main()
{
    int n, k, suma=0;
    printf(" n= ");
    scanf("%d", &n);
    k=n;
    while(k > 0)
    {
        suma+=k%10;
        k /= 10;
    }
    if(k%suma==0)
        printf("\n Broj %d jeste Nivenov\n", n);
    else
        printf("\n Broj %d nije Nivenov\n", n);
    getche();
    return 0;
}
```

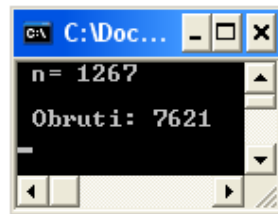


Испис на екрану

5.19. Саставити програм који за унети природни број исписује број чије су цифре у обрнутом редоследу унетог броја.

```
#include <stdio.h>

main()
{
    int n, obrnuti=0;
    printf(" n= ");
    scanf("%d", &n);
    while(n > 0)
    {
        obrnuti = obrnuti*10 + n%10;
        n /= 10;
    }
    printf("\n Obruti: %d\n", obrnuti);
    getche();
    return 0;
}
```



Испис на екрану

5.20. Саставити програм којим се дати природни број раставља на просте факторе. На пример, за 28 треба исписати: 2 2 7.

```
#include <stdio.h>

main()
{
    int m, n, k;
    printf(" n= ");
```

```

scanf("%d",&n);
printf("\n\n Prosti faktori:");
m=n/2;
for (k=2;k<=m;k++)
{
    while(n%k==0)
    {
        printf("\n%3d", k);
        n/=k;
    }
}
getche();
return 0;
}

```



Испис на екрану

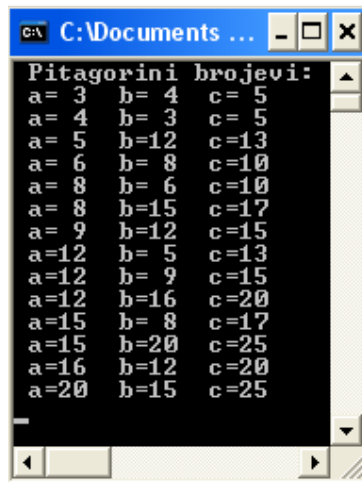
5.21. Саставити програм којим се испишују сви Питагорини бројеви **a**, **b** и **c** за $1 \leq a \leq 20$, $1 \leq b \leq 20$. Бројеви су Питагорини ако важи: $a^2 + b^2 = c^2$.

```

#include <stdio.h>

main()
{
    int a, b, c, zbir;
    printf(" Pitagorini brojevi:\n");
    for(a=1;a<=20;a++)
    {
        for(b=1;b<=20;b++)
        {
            zbir=a*a+b*b;
            c=1;
            while(c*c<=zbir)
            {
                if(c*c==zbir)
                {
                    printf(" a=%2d  b=%2d  c=%2d\n", a,b,c);
                    c++;
                }
            }
        }
    }
    getche();
    return 0;
}

```



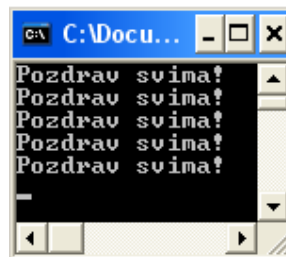
Испис на екрану

6 DO...WHILE ПЕТЉА

6.1. Саставити програм који ће пет пута исписати реченицу Pozdrav svima! употребом **DO WHILE** петље.

```
#include <stdio.h>

main()
{
    int i;
    i=1;
    do
    {
        printf("Pozdrav svima!\n");
        i++;
    }
    while(i<=5);
    getch();
    return 0;
}
```



Испис на екрану

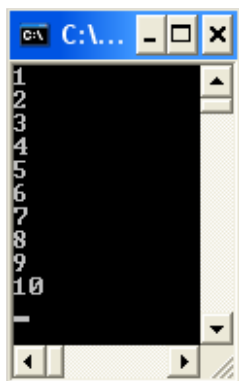
6.2. Саставити програм који употребом **DO WHILE** петље исписује:

- а) све бројеве прве десетице,
- б) само парне бројеве прве десетице
- в) све бројеве прве десетице у обрнутом редоследу.

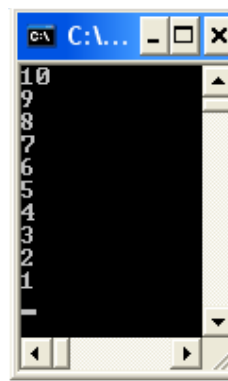
а)

```
#include <stdio.h>

main()
{
    int i;
    i=1;
    do
    {
        printf("%d\n", i);
        i++;
    }
    while(i<=10);
    getch();
    return 0;
}
```



Испис на екрану а)



Испис на екрану в)

6)

```
#include <stdio.h>

main()
{
    int i;
    i=2;
    do
    {
        printf("%d\n", i);
        i+=2;
    }
    while(i<=10);
    getch();
    return 0;
}
```



Испис на екрану б)

в)

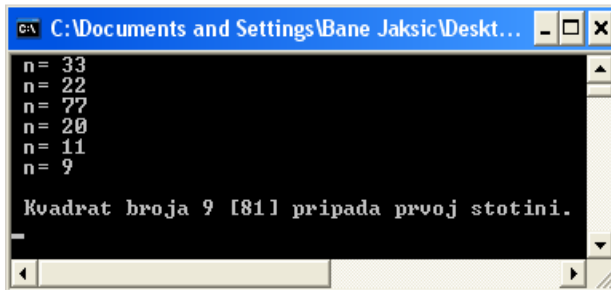
```
#include <stdio.h>

main()
{
    int i;
    i=10;
    do
    {
        printf("%d\n", i);
        i--;
    }
    while(i<=10 & i>0);
    getch();
    return 0;
}
```

6.3. Саставити програм који омогућује унос целих бројева све док препозна број чији квадрат припада првој стотини.

```
#include <stdio.h>

main()
{
    int n;
    do
    {
        printf(" n= ");
        scanf("%d", &n);
    }
    while(n*n<=0 || n*n>100);
    printf("\n Kvadrat broja %d [%d] pripada prvoj stotini.\n", n, n*n);
    getch();
    return 0;
}
```

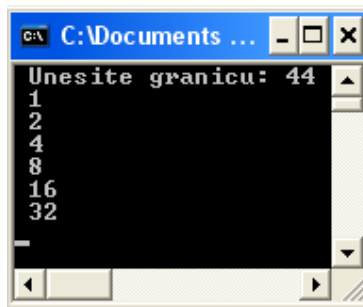


Испис на екрану

6.4. Саставити програм којим се исписују сви степени броја 2 који нису већи од унете вредности променљиве границе, а која је већа од броја 2.

```
#include <stdio.h>

main()
{
    int stepen, granica;
    printf(" Unesite granicu: ");
    scanf("%d", &granica);
    stepen=1;
    do
    {
        printf(" %d\n", stepen);
        stepen*=2;
    }
    while(stepen <= granica);
    getch();
    return 0;
}
```



Испис на екрану

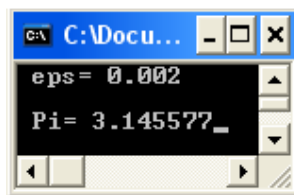
6.5. Саставити програм којим се израчунава број $\pi=3.1415926\dots$ коришћењем формуле:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots$$

Рачунање прекинути када се добије члан суме чија је апсолутна вредност мања од позитивног задатог броја **eps**.

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

main()
{
    float znak=-1.0;
    int i=1;
    float clan=1.0, suma=1.0, eps;
    printf(" eps= ");
    scanf("%f", &eps);
    do
    {
        clan=(float)znak/(2*i+1);
        suma+=clan;
        znak=-znak;
        i++;
    }
    while(fabs(clan) > eps);
    printf("\n Pi= %f", 4*suma);
    getch();
    return 0;
}
```

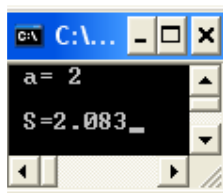


Испис на екрану

6.6. Саставити програм којим се међу бројевима $1, 1+\frac{1}{2}, 1+\frac{1}{2}+\frac{1}{3}, \dots$ проналази први већи од задатог броја **a**.

```
#include <stdio.h>

main()
{
    float a, i=0, s=0;
    printf(" a= ");
    scanf("%f",&a);
    do
    {
        i++;
        s=s+1/i;
    }
    while(s<a);
    printf("\n S=%.3f", s);
    getch();
    return 0;
}
```

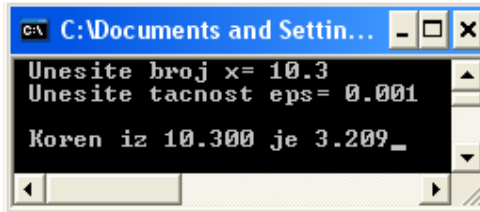


Испис на екрану

6.7. Саставити програм који ће израчунати квадратни корен Њутновом методом са задатом тачношћу. Њутнова формула гласи: $y_{i+1} = \frac{1}{2} \left(y_i + \frac{x}{y_i} \right)$, где је x број чији се тражи корен, а почетна итерација $y_0 = x + 1$.

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

main()
{
    float x, y, z, eps;
    printf(" Unesite broj x= ");
    scanf("%f", &x);
    printf(" Unesite tacnost eps= ");
    scanf("%f", &eps);
    y=x+1;
    do
    {
        z=y;
        y=0.5*(y+x/y);
    }
    while(fabs(z-y) > eps);
    printf("\n Koren iz %.3f je %.3f", x, y);
    getch();
    return 0;
}
```

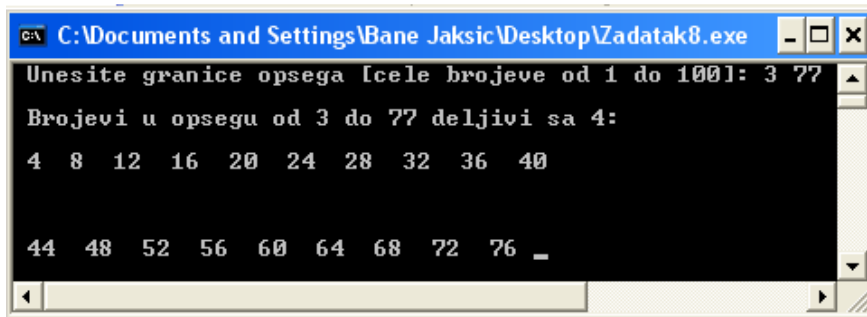


Испис на екрану

6.8. Саставити програм који на основу задатог опсега унутар прве стотине исписује бројеве дељиве са 4. Уколико се не зада коректна граница програм треба да захтева поновни унос. У сваком реду исписати по десет бројева.

```
#include <stdio.h>

main()
{
    int x, a, b, brojac=0;
    do
    {
        printf(" Unesite granice opsega [cele brojeve od 1 do 100]: ");
        scanf("%d%d",&a,&b);
    }
    while(a<1 || a>b || b<=a || b>100);
    printf("\n Brojevi u opsegu od %d do %d deljivi sa 4:\n", a, b);
    for(x=a;x<=b;x++)
    {
        if(x%4==0)
        {
            printf(" %d ", x);
            brojac++;
        }
        if(brojac%10==0)
            printf("\n");
    }
    getch();
    return 0;
}
```

Испис на екрану

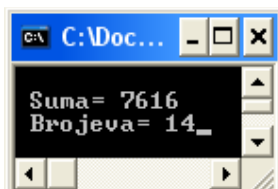
6.9. Саставити програм којим се врши сабирање и исписује сума свих троцифрених бројева дељивих са 64. Колико има таквих бројева?

```

#include <stdio.h>

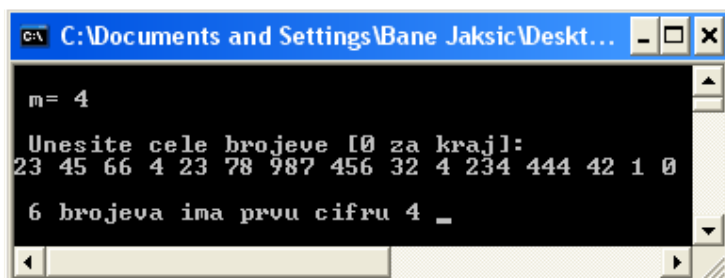
main()
{
    int i=100,s=0,br=0;
    do
    {
        if(i%64==0)
        {
            s+=i;
            br++;
        }
        i++;
    }
    while (i<=999);
    printf("\n Suma= %d", s);
    printf("\n Brojeva= %d", br);
    getch();
    return 0;
}

```



Испис на екрану

6.10. Саставити програм који ће прво да учита број **m**. Тај број мора бити већи од 0 и мањи од 10. Ако тај услов није испуњен читавање поновити. Програм треба даље да читава низ бројева, а унос се завршава са нулом. Програм треба наћи и исписати број **n** који показује колико међу учитаним бројевима има оних код којих је прва цифра **m**.



Испис на екрану

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

main()
{
    int m, n, k, br=0;
    float x;
    do
    {
        printf("\n m= ");
        scanf("%d", &m);
    }
    while (m<=0 || m>=10);
    printf ("\n Unesite cele brojeve [0 za kraj]:\n");
    scanf("%d",&k);
    do
    {
        n=log10 (k);    /*Nadji prvu cifru.*/
                        /*Moze i ovako: n=k; while(n>9) n/=10;.....*/
        n=k/pow(10,n); /*... n je prva cifra*/
        if(n==m) br++;
        scanf("%d",&k);
    }
    while(k!=0);
    printf("\n %d brojeva ima prvu cifru %d ", br, m);
    getch();
    return 0;
}
```

7 СКОКОВИ

7.1 BREAK

7.1. Шта се испишује на екрану након извршавања следећег програмског кода:

a)

```
#include <stdio.h>

main()
{
    int i;
    for(i=1;i<=10;i++)
    {
        if(i==5) break;
        printf(" i=%d \n",i);
    }
    getche();
    return 0;
}
```

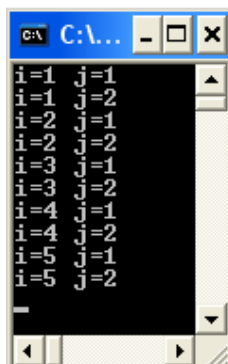


Испис на екрану а)

б)

```
#include <stdio.h>

main()
{
    int i, j;
    for(i=1;i<=5;i++)
        for(j=1;j<=10;j++)
        {
            if(j==3) break;
            printf("i=%d j=%d\n",i,j);
        }
    getche();
    return 0;
}
```

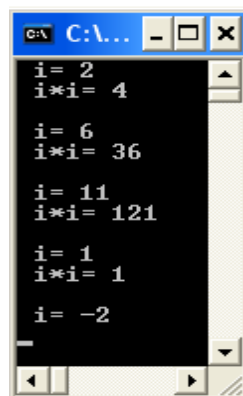


Испис на екрану б)

7.2. Саставити програм који ће учитавати позитивне целе бројеве и исписивати њихове квадрате све док не прочита негативни број.

```
#include <stdio.h>

main()
{
    int i;
    while(1)
    {
        printf(" i= "); scanf("%d",&i);
        if (i<0) break;
        printf(" i*i= %d\n\n", i*i);
    }
    getch();
    return 0;
}
```

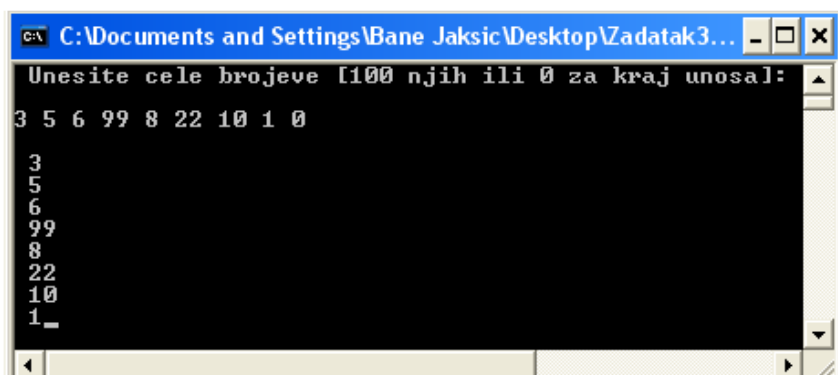


Испис на екрану

7.3. Саставити програм који са тастатуре копира на екран низ целих бројева све док се не унесе 100 или број 0 као STOP код.

```
#include <stdio.h>
#define STOP 0

main()
{
    int i,x;
    printf(" Unesite cele brojeve [100 njih ili 0 za kraj unosa]:\n\n");
    for(i=0; i<=100; i++)
    {
        scanf("%d",&x);
        if(x==STOP)
            break;
        printf("\n% d",x);
    }
    getch();
    return 0;
}
```



Испис на екрану

7.4. Саставити програм који ће исписати први број (ако постоји) који је мањи од 500, а дељив са бројевим 3, 4, 5 и 7.

```
#include <stdio.h>

main()
{
    int i;
    for(i=1; i<=500; i++)
        if(i%3==0 && i%4==0 && i%5==0 && i%7==0)
        {
            printf("%d\n", i);
            break;
        }
    getche();
    return 0;
}
```

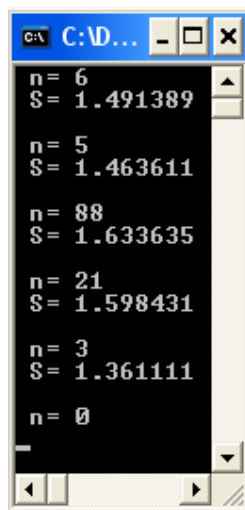


Испис на екрану

7.5. Саставити програм који за унето **n** рачуна суму $s = \sum_{i=1}^n \frac{1}{i^2}$. Програм треба да ради за произвољан број уноса **n** све док за његову вредност не прочита нулу или негативни број..

```
#include <stdio.h>

main()
{
    int n, i;
    float s;
    while(1)
    {
        printf(" n= ");
        scanf("%d", &n);
        if(n <= 0) break;
        s=0;
        for(i=1; i<=n; i++)
            s+=1./(i*i);
        printf (" S= %.6f\n\n", s);
    }
    getche();
    return 0;
}
```



Испис на екрану

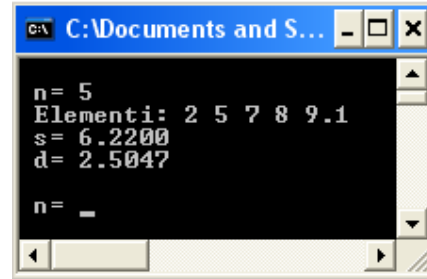
7.6. Саставити програм који ће прво учитати број **n** који означава број елемената низа, а затим учитава елементе низа. Када прочита елементе треба израчунати и исписати аритметичку средњу вредност $s = \frac{1}{n} \sum_{i=1}^n a_i$ и стандардну девијацију $d = \sqrt{\frac{1}{n} \sum_{i=1}^n a_i^2 - s^2}$. Програм треба да обрађује произвољан број комплета све док за број елемената низа не прочита нулу или негативни број.

```

#include <stdio.h>

main()
{
    while(1)
    {
        double a, s, d;
        int n,i;
        printf("\n n= ");
        scanf("%d", &n);
        if(n <= 0) break;
        printf(" Elementi: ");
        s=0, d=0;
        for(i=1; i<=n; i++)
        {
            scanf("%lf",&a);
            s+=a;
            d+=a*a;
        }
        s/=n;
        d=sqrt(d/n-s*s);
        printf(" s= %.4f\n", s);
        printf(" d= %.4f\n", d);
    }
    getch();
    return 0;
}

```



Испис на екрану

7.7. Саставити програм који ће исписати све просте бројеве од 1 до 100. Број је прост ако је дељив само са 1 и самим собом.

```

#include <stdio.h>

main()
{
    int i,j,m;
    float x;
    for(i=1; i<=100; i=i+1)
    {
        x=i;
        m=sqrt(x);
        for(j=2; j<=m; j++)
            if(i%j==0) break;
        if(i==2 || i%j!=0)
            printf(" %2d",i);
    }
    getch();
    return 0;
}

```



Испис на екрану

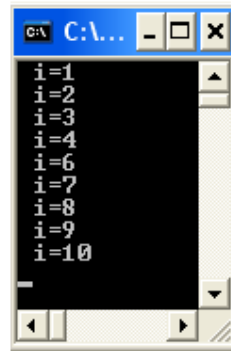
7.2 CONTINUE

7.8. Шта се исписује на екрану након извршавања следећег програмског кода:

a)

```
#include <stdio.h>

main()
{
    int i;
    for(i=1;i<=10;i++)
    {
        if(i==5) continue;
        printf(" i=%d \n",i);
    }
    getch();
    return 0;
}
```

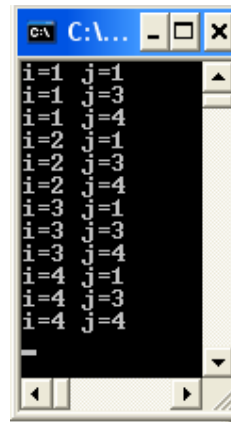


Испис на екрану а)

б)

```
#include <stdio.h>

main()
{
    int i, j;
    for(i=1;i<=4;i++)
        for(j=1;j<=4;j++)
        {
            if(j==2) continue;
            printf("i=%d j=%d\n",i,j);
        }
    getch();
    return 0;
}
```



Испис на екрану б)

7.9. Саставити програм користећи наредбу **continue** који ће исписати парне бројеве прве десетице.

```
#include <stdio.h>

main()
{
    int i;
    for(i=1;i<10; i++)
    {
        if((i%2)!=0) continue;
        printf(" %d\n", i);
    }
    getch();
    return 0;
}
```



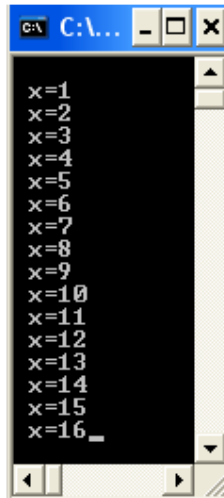
Испис на екрану

7.10. Које вредности променљиве **x** се исписују на екран након извршавања следећег програма:

а)

```
#include <stdio.h>

main()
{
    int i, x=0;
    for(i=1; i<20; i++)
    {
        if(i%5==0) continue;
        x++;
        if(i==10) break;
        printf("\n x=%d", x);
    }
    getch();
    return 0;
}
```

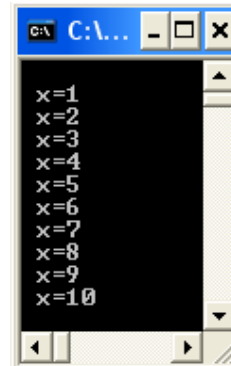


Испис на екрану а)

б)

```
#include <stdio.h>

main()
{
    int i,x=0;
    for(i=1; i<20; i++)
    {
        if(i%8==0) continue;
        x++;
        if(i==8 || i==12 || i==16) break;
        printf ("\n x=%d", x);
    }
    getch();
    return 0;
}
```

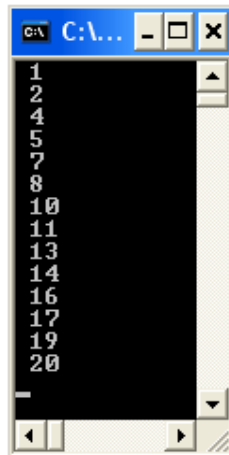


Испис на екрану б)

7.11. Саставити програм који исписује све бројеве мање од 20 који нису дељиви са бројем 3.

```
#include <stdio.h>

main()
{
    int i;
    for(i=1; i<=20; i++)
    {
        if(i%3==0) continue;
        printf(" %d\n", i);
    }
    getch();
    return 0;
}
```

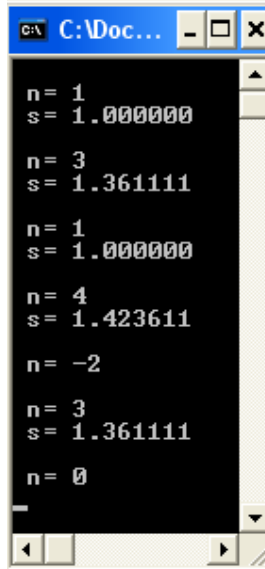


Испис на екрану

7.12. Саставити програм за рачунање суме $s = \sum_{i=1}^n \frac{1}{i^2}$ за позитивне вредности **n**. За сваки негативни број **n** тражи нову вредност, а за **n=0** прекида извршавање.

```
#include <stdio.h>

main()
{
    int n, i;
    float s;
    while(1)
    {
        printf ("\n n= ");
        scanf ("%d", &n);
        if(n == 0) break;
        if(n < 0) continue;
        s=0;
        for(i=1; i<=n; i++)
            s+=1./(i*i);
        printf (" s= %.6f\n", s);
    }
    getche();
    return 0;
}
```

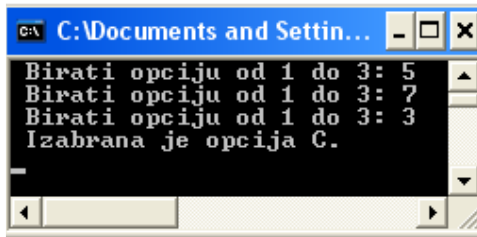


Испис на екрану

7.3 GOTO

7.13. Шта ће се исписати на екрану, уколико се унесу бројеви 5, 7 и 3 након извршавања следећег програма:

```
#include <stdio.h>
main()
{
    int opcija;
start: printf(" Birati opciju od 1 do 3: ");
    scanf("%d", &opcija);
    if(opcija<1 || opcija>3)
        goto start;
    else if(opcija==1)
        goto Num1;
    else if(opcija==2)
        goto Num2;
    else if(opcija==3)
        goto Num3;
    Num1: printf(" Izabrana je opcija A.\n");
        goto End;
    Num2: printf(" Izabrana je opcija B.\n");
        goto End;
    Num3: printf(" Izabrana je opcija C.\n");
        goto End;
    End:;
    getche();
    return 0;
}
```



```

C:\Documents and Settings\Ba...
Birati opciju od 1 do 3: 5
Birati opciju od 1 do 3: 7
Birati opciju od 1 do 3: 3
Izabrana je opcija C.

```

Испис на екрану

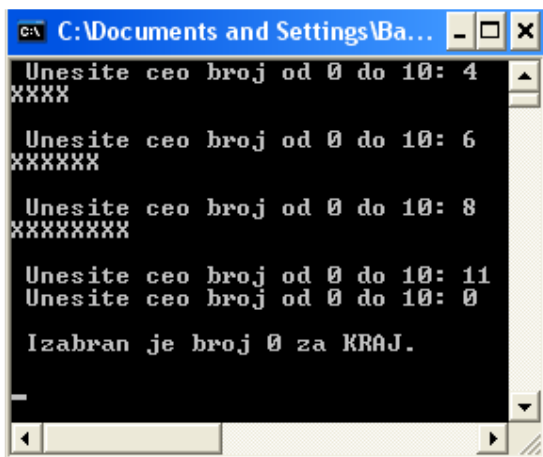
7.14. Саставити програм који ће понављати захтев за уношење броја све док се број не унесе у задатом опсегу, и који ће цртати хистограм вредности ако је у задатом опсегу.

```

#include <stdio.h>

main()
{
    int i, n;
    start: printf(" Unesite ceo broj od 0 do 10: ");
           scanf("%d", &n);
           if(n<0 || n>10)
               goto start;
           else if (n==0)
               goto Loc0;
           else
               goto Loc1;
    Loc0: printf("\n Izabran je broj 0 za KRAJ.\n\n");
           goto End;
    Loc1: for(i=0;i<n;i++)
           printf("X");
           printf("\n\n");
           goto start;
    End: ;
    getch();
    return 0;
}

```



```

C:\Documents and Settings\Ba...
Unesite ceo broj od 0 do 10: 4
XXXX

Unesite ceo broj od 0 do 10: 6
XXXXXX

Unesite ceo broj od 0 do 10: 8
XXXXXXXX

Unesite ceo broj od 0 do 10: 11
Unesite ceo broj od 0 do 10: 0
Izabran je broj 0 za KRAJ.

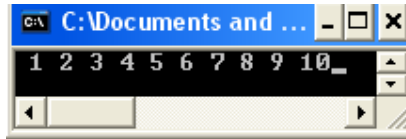
```

Испис на екрану

7.15. Саставити програм који ће исписати бројеве прве десетице помоћу **goto** наредбе.

```
#include <stdio.h>

main()
{
    int i=1;
    poc: printf(" %d", i);
    i++;
    if(i<=10)
        goto poc;
    getch();
    return 0;
}
```



Испис на екрану

7.16. Саставити програм који ће за унети природан број **n** исписати производ његових цифара.

```
#include <stdio.h>

main()
{
    int n, a, p=1;
    printf(" n= ");
    scanf("%d",&n);
    poc: a=n%10;
    p=p*a;
    n=n/10;
    if(n!=0)
        goto poc;
    printf(" p= %d", p);
    getch();
    return 0;
}
```



Испис на екрану

7.17. Саставити програм којим се дати природни број раставља на просте факторе. На пример, за 28 треба исписати: 2 2 7.

```
#include <stdio.h>

main()
{
    int n, f=2;
    printf(" n= ");
    scanf("%d",&n);
    printf(" %d=1",n);
    poc: if(n!=1)
    {
        if(n%f==0)
        {
            printf(" *%d",f);
            n=n/f;
            goto poc;
        }
    }
}
```

```
else
{
    f++;
    goto poc;
}
getch();
return 0;
}
```



Испис на екрану

7.18. Саставити програм који ће учитавати реалне бројеве све док се не прочита нула. Затим треба исписати колико је бројева прочитано, колико је међу прочитаним бројевима позитивно, а колико негативно и аритметичку средину прочитаних бројева.

```
#include <stdio.h>

main()
{
    float n, br=0, poz=0, neg=0, s=0, as;
    poc: printf("\n Ucitaj broj: ");
    scanf("%f",&n);
    if(n!=0)
    {
        br++;
        s=s+n;
        if(n>0) poz++;
        else neg++;
        goto poc;
    }
    as=s/br;
    printf("\n Ucitano: %.0f brojeva.", br);
    printf("\n Pozitivnih: %.0f brojeva. ", poz);
    printf("\n Negativnih: %.0f brojeva. ", neg);
    printf("\n Aritmeticka sredina: %.3f", as);
    getche();
    return 0;
}
```

Испис на екрану

7.4 SWITCH ... CASE

7.19. Саставити програм који за унутру редни број од 1 до 7 исписати одговарајући дан у недељи.

```
#include <stdio.h>

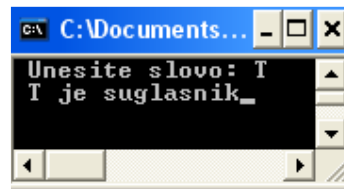
main()
{
    int x;
    printf(" n= ");
    scanf("%d",&x);
    switch(x)
    {
        case 1:
            printf("\n ponedeljak");
            break;
        case 2:
            printf("\n utorak");
            break;
        case 3:
            printf("\n sreda");
            break;
        case 4:
            printf("\n cetvrtak");
            break;
        case 5:
            printf("\n petak");
            break;
        case 6:
            printf("\n subota");
            break;
        case 7:
            printf("\n nedelja");
            break;
        default:
            printf("\n GRESKA");
            break;
    }
    getche();
    return 0;
}
```

Испис на екрану

7.20. Саставити програм који испишује на екрану да ли је унето слова самогласник или сугласник.

```
#include <stdio.h>

main()
{
    char x;
    printf(" Unesite slovo: ");
    scanf("%c",&x);
    switch (x)
    {
        case 'a':
        case 'e':
        case 'i':
        case 'o':
        case 'u':
        case 'A':
        case 'E':
        case 'I':
        case 'O':
        case 'U':
            printf(" %c je samoglasnik",x);
            break;
        default:
            printf(" %c je suglasnik",x);
    }
    getch();
    return 0;
}
```

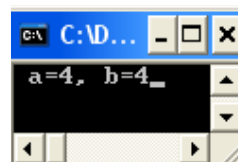


Испис на екрану

7.21. Шта се испишује на екрану након извршавање следећег програма:

```
#include <stdio.h>

main()
{
    int a=6, b=4;
    switch (a%b)
    {
        case 0:
            a++;
            b++;
        case 1:
            a++;
            b++;
            break;
        case 2:
            a--;
            b--;
        case 3:
            a--;
            b++;
            break;
    }
    printf (" a=%d, b=%d", a, b);
    getch();
    return 0;
}
```



Испис на екрану

7.22. Саставити програм који за унети знак операције (+, -, *, /) и два реална операнда испишује резултат операција.

```
#include <stdio.h>

main()
{
    float op1, op2;
    char op;
    printf(" Operator: ");
    scanf ("%c", &op);
    printf(" Operand1: ");
    scanf ("%f", &op1);
    printf(" Operand2: ");
    scanf ("%f", &op2);
    switch(op)
    {
        case '+':
            printf("Vrednost: %.2f\n", op1 + op2);
            break;
        case '-':
            printf("Vrednost: %.2f\n", op1 - op2);
            break;
        case '*':
            printf("vrednost: %.2f\n", op1 * op2);
            break;
        case '/':
            if (op2 == 0)
                printf("Greska! Deljenje nulom!\n");
            else
                printf("\n Rezultat: %.2f\n", op1 / op2);
            break;
        default:
            printf(" Greska! Pogresan operator\n");
    }
    getch();
    return 0;
}
```



Испис на екрану

7.23. Саставити програм који ће учитати дужине страница правоугаоника, а затим на основу избора опција, рачуна и испишује обим $O = 2(a + b)$, површину $P = ab$ или дужину дијагонале $d = \sqrt{a^2 + b^2}$ тог правоугаоника.

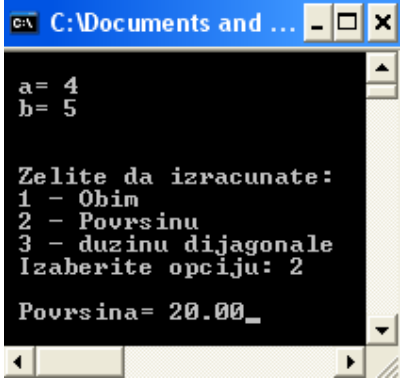
```
#include <stdio.h>
#include <math.h>

main()
{
    int i;
    float a, b, O, P, d;
    printf("\n a= ");
    scanf ("%f", &a);
    printf(" b= ");
    scanf ("%f", &b);
    if(a>0 && b>0)
    {
        printf("\n\n Zelite da izracunate:");
```

```

printf("\n 1 - Obim\n 2 - Povrsinu\n 3 - duzinu dijagonale");
printf("\n Izaberite opciju: ");
scanf("%d",&i);
switch(i)
{
    case 1:
        O=2*(a+b);
        printf("\n Obim= %.2f", O);
        break;
    case 2:
        P=a*b;
        printf("\n Povrsina= %.2f", P);
        break;
    case 3:
        d=sqrt(a*a+b*b);
        printf("\n Dijagonala= %.2f", d);
        break;
    default:
        printf("Izabrali ste pogresan broj.");
        break;
}
}
else printf ("GRESKA!!!");
getche();
return 0;
}

```



Испис на екрану

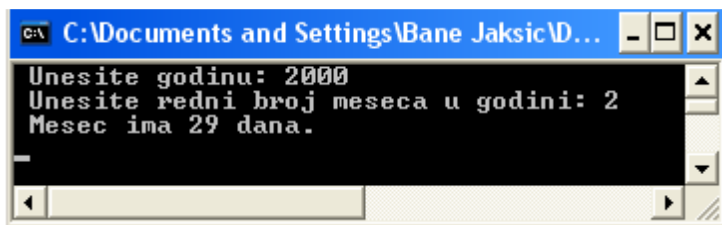
7.24. Саставити програм који за унуту годину и редни број месеца у години исписује број дана у том месецу. Узети у обзир преступне године.

```

#include <stdio.h>

main()
{
    int mesec, godina, BrDana;
    printf(" Unesite godinu: ");
    scanf("%d", &godina);
    printf(" Unesite redni broj meseca u godini: ");
    scanf("%d", &mesec);
    switch(mesec)
    {
        case 1: case 3: case 5: case 7: case 8: case 10: case 12:
            BrDana=31;
            break;
        case 4: case 6: case 9: case 11:
            BrDana=30;
            break;
        case 2:
            BrDana=28+(godina%4==0 && godina%100!=0 || godina%400==0);
            break;
        default:
            BrDana=0;
            break;
    }
    if(BrDana!=0)
        printf(" Mesec ima %d dana.\n", BrDana);
    getche();
    return 0;
}

```

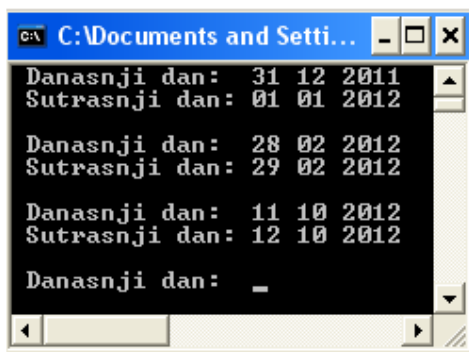


Испис на екрану

7.25. Саставити програм за исписивање наредног датума у односу на задати дан. Програм треба да чита датуме и исписује резултате све док за једну компоненту датума не прочита нулу.

```
#include <stdio.h>

main()
{
    int dan, mesec, godina, d;
    while (1)
    {
        printf(" Danasnji dan: ");
        scanf("%d%d%d", &dan, &mesec, &godina);
        if(dan==0 || mesec==0 || godina==0) break;
        switch (mesec)
        {
            case 1: case 3: case 5: case 7: case 8: case 10: case 12:
                d=31;
                break;
            case 4: case 6: case 9: case 11:
                d=30;
                break;
            case 2:
                d=28+(godina%4==0 && godina%100!=0 || godina%400==0);
                break;
        }
        if(dan<d) dan++;
        else
        {
            dan=1;
            if(mesec<12) mesec++;
            else
            {
                mesec=1;
                godina++;
            }
        }
        printf(" Sutrasnji dan: %2.2d %2.2d %d\n\n", dan, mesec, godina);
    }
    getch();
    return 0;
}
```

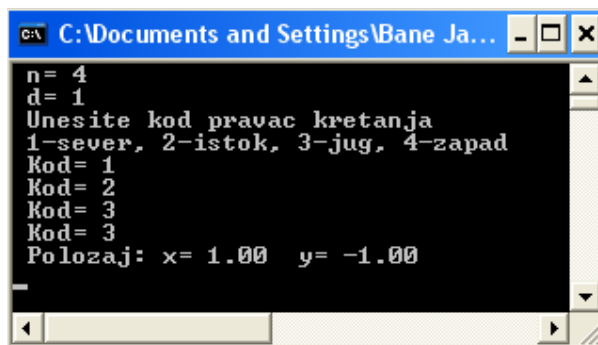



Испис на екрану

7.26. Дато је **n** инструкција кретања учитавањем **n** вредности променљиве $a \in \{1, 2, 3, 4\}$. Ако 1 представља кретање на север, 2 на исток, 3 на југ, 4 на запад, саставити програм који исписује позиције тачке која полази из координатног почетка. Свака инструкција задаје кретање за дужину **d**.

```
#include <stdio.h>

main()
{
    int i, n, a;
    float x=0, y=0, d;
    printf(" n= ");
    scanf("%d", &n);
    printf(" d= ");
    scanf("%f", &d);
    printf(" Unesite kod pravca kretanja");
    printf("\n 1-sever, 2-istok, 3-jug, 4-zapad\n");
    for(i=1; i<=n; i++)
    {
        printf(" Kod= ");
        scanf("%d", &a);
        switch (a)
        {
            case 1:
                y=y+d;
                break;
            case 2:
                x=x+d;
                break;
            case 3:
                y=y-d;
                break;
            case 4:
                x=x-d;
                break;
        }
    }
    printf(" Polozaj: x= %2.2f  y= %2.2f\n", x, y);
    getch();
    return 0;
}
```



Испис на екрану

8 КАРАКТЕРИ – ЗНАКОВНИ УЛАЗ И ИЗЛАЗ

Табела 8.1: Функције за читање и писање знакова без конверзије дефинисане у библиотеци `<stdio.h>`

функција	значење
<code>getchar()</code>	Функција чита следећи знак, укључујући и беле знакове са главног улаза рачунара (тастатуре). Вредност функције је код прочитаног знака или симболичка константа EOF (<i>end of file</i>) уколико је прочитани сигнал за крај датотеке као и у случају грешке у току читања. Резултат је типа int . Сигнал за крај датотеке преко тастатуре под оперативним системом <i>MS-DOS</i> се задаје управљачким знаком <i>ctrl+Z</i> .
<code>putchar(c)</code>	Функција исписује знак <i>c</i> на главном излазу рачунара (екрану). Тип аргумента је int . Вредност функције је код исписаног знака или симболичка константа EOF у случају грешке. Резултат је типа int .
<code>gets(s)</code>	Функција чита ред текста (до знака за прелазак у нови ред <code>\n</code>) са главног улаза рачунара (тастатуре) и смешта га, као бочни ефекат у ниску <i>s</i> (типа char [], односно char*). Уместо знака <code>\n</code> у низ <i>s</i> се поставља знак <code>\0</code> . Вредност функције у случају наилаз на крај датотеке, или у случају откривања грешке у току читања, једнака је симболичкој константи NULL, односно почетној адреси низа <i>s</i> у случају успешног читања.
<code>puts(s)</code>	Функција исписује ред ниске <i>s</i> (типа char [], односно char*) до завршног знака <code>\0</code> као ред текста на главном излазу рачунара (екрану), додајући знак за прелазак у нови ред (<code>\n</code>) иза последњег знака. Ако текст у низу <i>s</i> садржи и знакове <code>\n</code> , резултат ће бити више исписаних редова. Вредност функције (типа int) је не-негативни број, односно симболичка константа EOF у случају откривања грешке у току испитивања.

* Функције `gets(s)` и `puts(s)` применићемо у област о стринговима.

Табела 8.2: Функције за испитивање знакова дефинисане у библиотеци `<ctype.h>`

функција	значење
<code>isalnum(c)</code>	Да ли је <i>c</i> слово или цифра?
<code>isalpha(c)</code>	Да ли је <i>c</i> слово?
<code>islower(c)</code>	Да ли је <i>c</i> мало слово?
<code>isupper(c)</code>	Да ли је <i>c</i> велико слово?
<code>isdigit(c)</code>	Да ли је <i>c</i> децимална цифра?
<code>isxdigit(c)</code>	Да ли је <i>c</i> хексадецимална цифра?

isspace(c)	Да ли је c бели знак?
isgraph(c)	Да ли је c штампајући знак, али није размак?
isprint(c)	Да ли је c штампајући знак (укључујући и размак)?
ispunct(c)	Да ли је c специјални знак (штампајући али није слово ни цифра)?
isctrl(c)	Да ли је c управљачки знак?
tolower(c)	Ако је c велико слово, вредност функције је код одговарајућег малог слова, а иначе вредност функције је c.
toupper(c)	Ако је c мало слово, вредност функције је код одговарајућег великог слова, а иначе вредност функције је c.

* Вредност свих функција је типа *logical*, а тип аргумента c је **int** (вредност треба да је код неког знака). За логичку неистину дају неку ненулту вредност, а не обавезно вредност 1, као што дају релацијски и логички оператори.

8.1. Шта се испишује на екрану након извршавања следећег програмског кода:

```
#include <stdio.h>

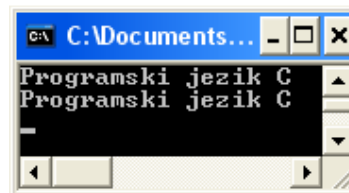
main()
{
    putchar('\n');
    putchar('a');
    putchar('\n');
    putchar(97);
    putchar('\n');
    putchar('A' + 10);
    putchar('\n');
    putchar('a' + 3);
    getch();
    return 0;
}
```



Испис на екрану
(Задатак 1)



Испис на екрану
(Задатак 2a)
Са тастатуре је унето: A



Испис на екрану
(Задатак 2б)
Са тастатуре је унето:
Programski jezik C

8.2. Саставити програм који:

- чита са тастатуре један знак и испишује га на екрану.
- чита са тастатуре неограничени број карактера све док се не прочита EOF и испишује унете карактере на екрану.

a)

```
#include <stdio.h>

main()
{
    int c;
    c=getchar();
    putchar(c);
    getch();
    return 0;
}
```

б)

```
#include <stdio.h>

main()
{
    int c;
    while(c!=EOF)
    {
        c=getchar();
        putchar(c);
    }
    getch();
    return 0;
}
```

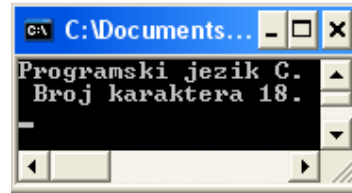
8.3. Саставити програм за:

- а) унос текста са тастатуре све док се не унесе тачка (.) и приказује број карактера до тачке.
 б) унос текста са тастатуре све док се не прочита EOF и приказује укупан број карактера.

а)

```
#include <stdio.h>

main()
{
    int c, n=0;
    c=getche();
    while(c != '.')
    {
        c=getche();
        n++;
    }
    printf("\n Broj karaktera %d.\n", n);
    getch();
    return 0;
}
```

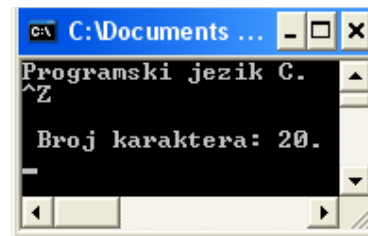


Испис на екрану а)
 Са тастатуре је унето:
 Programski jezik C.

б)

```
#include <stdio.h>

main()
{
    int c, n=0;
    while((c=getchar()) != EOF)
        n++;
    printf("\n Broj karaktera: %d.\n",n);
    getch();
    return 0;
}
```



Испис на екрану б)
 Са тастатуре је унето:
 Programski jezik C.

8.4. Саставити програм који броји децималне цифре у улазном тексту до ознаке за крај уноса EOF.

Први начин:

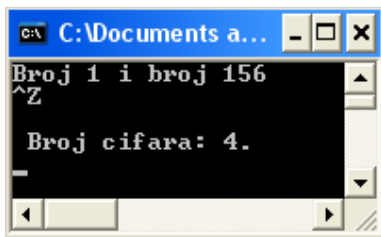
```
#include <stdio.h>

main()
{
    int c, n=0;
    while((c=getchar()) != EOF)
        if(c>='0' && c<='9')
            n++;
    printf("\n Broj cifara: %d.\n", n);
    getch();
    return 0;
}
```

Други начин:

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

main()
{
    int c, n=0;
    while((c=getchar()) != EOF)
        if(isdigit(c))
            n++;
    printf("\n Broj cifara: %d.\n",n);
    getch();
    return 0;
}
```



Испис на екрану
Са тастатуре је унето:
Broj 1 i broj 156

8.5. Саставити програм који броји празне знакове (размак, хоризонтални табулатор и нови ред), слова, децималне цифре као и све знакове улазног текста до ознаке за крај уноса EOF.

Први начин:

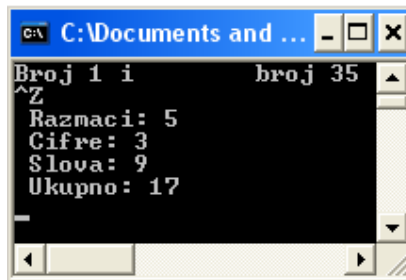
```
#include <stdio.h>

main()
{
    int c, nk=0, nr=0, nb=0, ns=0;
    while((c=getchar()) != EOF)
    {
        if ((c==' ') || (c=='\t') || (c=='\n')) nr++;
        if (c>='0' && c<='9') nb++;
        if ((c>='a' && c<='z') || (c>='A' && c<='Z')) ns++;
        nk++;
    }
    printf(" Razmaci: %d\n", nr);
    printf(" Cifre: %d\n", nb);
    printf(" Slova: %d\n", ns);
    printf(" Ukupno: %d\n", nk);
    getche();
    return 0;
}
```

Други начин:

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

main()
{
    int c, nk=0, nr=0, nb=0, ns=0;
    while((c=getchar()) != EOF)
    {
        if (isspace(c)) nr++;
        if (isdigit(c)) nb++;
        if (isalpha(c)) ns++;
        nk++;
    }
    printf(" Razmaci: %d\n", nr);
    printf(" Cifre: %d\n", nb);
    printf(" Slova: %d\n", ns);
    printf(" Ukupno: %d\n", nk);
    getche();
    return 0;
}
```

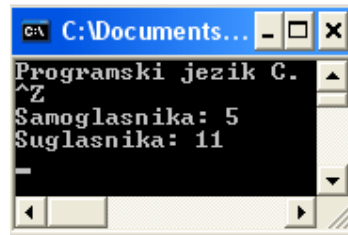


Испис на екрану

8.6. Саставити програм за одређивање броја самогласника и сугласника у улазном тексту. Унос текста се завршава сигналом EOF.

```
#include <stdio.h>

main()
{
    int c, sugl=0, samog=0;
    while((c=getchar()) != EOF)
    {
        if((c>='a' && c<='z') || (c>='A' && c<='Z'))
            if(c=='a' || c=='e' || c=='i' || c=='o' || c=='u'
               || c=='A' || c=='E' || c=='I' || c=='O' || c=='U')
                samog++;
            else
                sugl++;
    }
    printf("Samoglasnika: %d\n", samog);
    printf("Suglasnika: %d\n", sugl);
    getch();
    return 0;
}
```



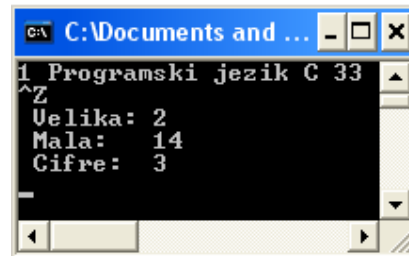
Испис на екрану
Са тастатуре је унето:
Programski jezik C

8.7. Саставити програм за одређивање броја великих слова, малих слова и цифара у улазном тексту. Унос текста се завршава сигналом EOF.

Први начин:

```
#include <stdio.h>

main()
{
    int c, veliko=0, malo=0, cifra=0;
    while((c = getchar()) != EOF)
    {
        if(c >= 'A' && c <= 'Z')    veliko++;
        if(c >= 'a' && c <= 'z')    malo++;
        if(c >= '0' && c <= '9')    cifra++;
    }
    printf(" Velika:  %d\n", veliko);
    printf(" Mala:    %d\n", malo);
    printf(" Cifre:   %d\n", cifra);
    getch();
    return 0;
}
```



Испис на екрану
Са тастатуре је унето:
1 Programski jezik C 33

Други начин:

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

main()
{
    int c, veliko=0, malo=0, cifra=0;
    while((c = getchar()) != EOF)
    {
        veliko += isupper(c) != 0;
        malo  += islower(c) != 0;
        cifra  += isdigit(c) != 0;
    }
}
```

```

    }
    printf(" Velika: %d\n", veliko);
    printf(" Mala:   %d\n", malo);
    printf(" Cifre:  %d\n", cifra);
    getch();
    return 0;
}

```

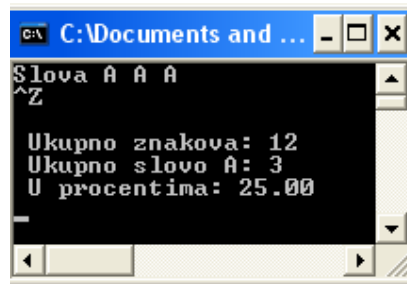
8.8. Саставити програм за одређивање броја појављивања слова **A** у улазном тексту и изразити ту вредност процентуално у односу на све унете знаке. Унос текста се завршава сигналом EOF.

```

#include <stdio.h>

main()
{
    int c, n=0, u=0;
    float p;
    while((c = getchar()) != EOF)
    {
        u++;
        if(c=='A')    n++;
    }
    p=(float)n/u*100;
    printf("\n Ukupno znakova: %d", u);
    printf("\n Ukupno slovo A: %d", n);
    printf("\n U procentima: %.2f%\n", p);
    getch();
    return 0;
}

```



Испис на екрану
Са тастатуре је унето:
Slova A A A

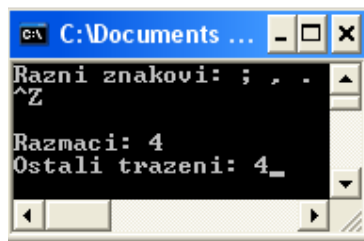
8.9. Саставити програм за одређивање броја празних карактера, као и броја знакова: тачка, зарез, двотачка и тачка-зарез. Унос текста се завршава сигналом EOF.

```

#include <stdio.h>

main()
{
    int c, razmak=0, n=0;
    while((c=getchar())!=EOF)
    switch(c)
    {
        case ' ':
            razmak++;
            break;
        case '.':
        case ',':
        case ':':
        case ';':
            n++;
            break;
        default:
            break;
    }
    printf("\nRazmaci: %d", razmak);
    printf("\nOstali trazeni: %d", n);
    getch();
    return 0;
}

```

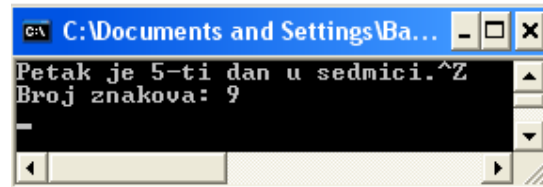


Испис на екрану
Са тастатуре је унето:
Razni znakovi: ; , .

8.10. Саставити програм који броји карактере улазног текста до прве децималне цифре. Унос текста се завршава сигналом EOF.

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

main()
{
    int c, n=0;
    while((c=getchar()) != EOF)
    {
        if(isdigit(c)) break;
        n++;
    }
    printf("Broj znakova: %d\n", n);
    getche();
    return 0;
}
```

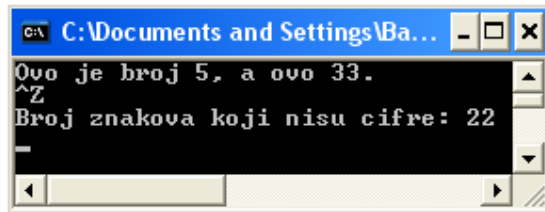


Испис на екрану
Са тастатуре је унето:
Petak je 5-ti dan u sedmici.

8.11. Саставити програм који броји карактере улазног текста различите од децималних цифара. Унос текста се завршава сигналом EOF.

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

main()
{
    int c, n=0;
    while((c=getchar()) != EOF)
    {
        if(isdigit(c)) continue;
        n++;
    }
    printf("Broj znakova koji nisu cifre: %d\n", n);
    getche();
    return 0;
}
```

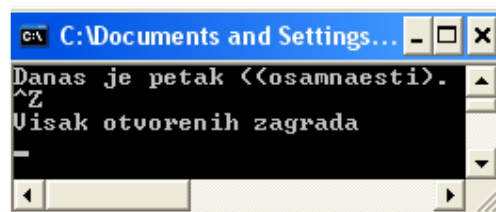


Испис на екрану
Са тастатуре је унето:
Ovo je broj 5, a ovo 33.

8.12. Саставити програм који испитује да ли су у унетом тексту заграде“(“ и “)” добро упарене. Унос текста се завршава сигналом EOF.

```
#include <stdio.h>

main()
{
    int c;
    int otv=0;
    while((c=getchar()) != EOF)
    {
        switch(c)
        {
            case '(':
                otv++;
                break;
        }
    }
}
```



Испис на екрану
Са тастатуре је унето:
Danas je petak (<osamnaesti).


```
        case ')':
            otv--;
            if (otv<0)
            {
                printf("Visak zatvorenih zagrada\n");
                break;
            }
        }
    }
    if (otv==0)
        printf("Zagrade su u redu\n");
    else
        printf("Visak otvorenih zagrada\n");
    getch();
    return 0;
}
```

8.13. Саставити програм који врши конверзију унетих великих слова у мала. Унос текста се завршава сигналом EOF.

Први начин:

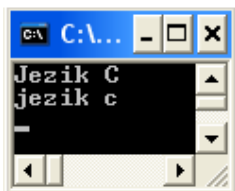
```
#include <stdio.h>

main()
{
    int c;
    while((c=getchar()) != EOF)
    {
        if(c >= 'A' && c <= 'Z')
            c = c - 'A' + 'a';
        putchar(c);
    }
    getch();
    return 0;
}
```

Други начин:

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

main()
{
    int c;
    while((c=getchar()) != EOF)
        putchar(tolower(c));
    getch();
    return 0;
}
```



Испис на екрану
Са тастатуре је унето:
Jezik C

8.14. Саставити програм који врши конверзију унетих малих слова у велика. Унос текста се завршава сигналом EOF.

Први начин:

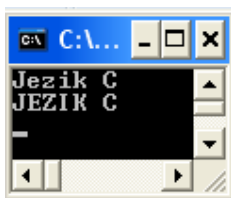
```
#include <stdio.h>

main()
{
    int c;
    while((c=getchar()) != EOF)
    {
        if(c >= 'a' && c <= 'z')
            c = c - 'a' + 'A';
        putchar(c);
    }
    getche();
    return 0;
}
```

Други начин:

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

main()
{
    int c;
    while((c=getchar()) != EOF)
        putchar(toupper(c));
    getche();
    return 0;
}
```

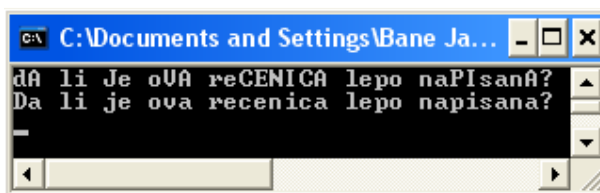


Испис на екрану
Са тастатуре је унето:
Jezik C

8.15. Саставити програм који врши конверзију првог слова у велико, а сва остала у мала унете реченице. Унос текста се завршава сигналом EOF.

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

main()
{
    int c, prvi=0;
    while((c=getchar()) != EOF)
    {
        if(isalpha(c))
        {
            if(prvi==0)
                putchar(toupper(c));
            else putchar(tolower(c));
        }
        else putchar(c);
        prvi=1;
    }
    getche();
    return 0;
}
```

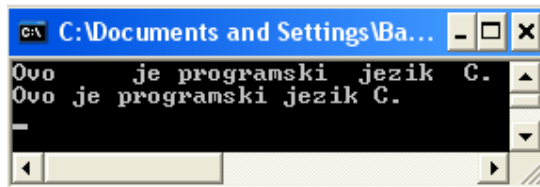


Испис на екрану

8.16. Саставити програм којим се копира улазни текст на екран уз раздвајање речи по једним знаком размака. Речи су на улазу раздвојене произвољним бројем знакова размака и/или табулације. Текст се завршава сигналом EOF.

```
#include <stdio.h>
#define NIJE_PRAZNO 'a'
#define PRAZNO ' '
#define HOR_TAB '\t'

main()
{
    int c, predh_c=NIJE_PRAZNO;
    while((c=getchar()) != EOF)
    {
        if(c==HOR_TAB)
            c=PRAZNO;
        if(c != PRAZNO || predh_c != PRAZNO)
            putchar(c);
        predh_c=c;
    }
    getche();
    return 0;
}
```

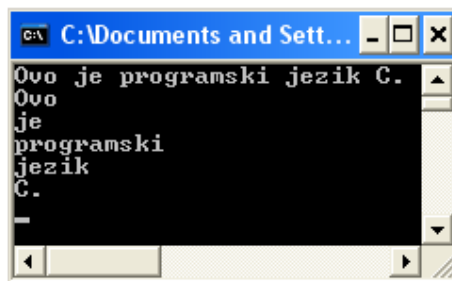


Испис на екрану

8.17. Саставити програм који копира улазни текст на екран, где сваку реч приказује у једној линији. Унос текста се завршава сигналом EOF.

```
#include <stdio.h>
#define U_RECI 1
#define VAN_RECI 0

main()
{
    int c, poz=VAN_RECI;
    while((c=getchar()) != EOF)
    {
        if(c==' ' || c=='\n' || c=='\t')
        {
            if(poz==U_RECI)
            {
                putchar('\n');
                poz=VAN_RECI;
            }
        }
        else
        {
            poz=U_RECI;
            putchar(c);
        }
    }
    getche();
    return 0;
}
```



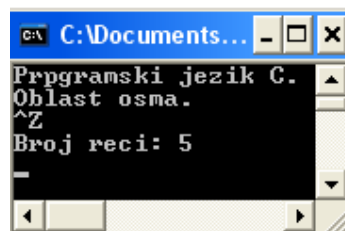
Испис на екрану

8.18. Саставити програм који учитава текст са улаза, и при том броји речи. Под речју се подразумева низ не-бланко карактера (бланко карактери су ' ', '\n' и '\t').

```
#include <stdio.h>
#define U_RECI 1
#define VAN_RECI 0

main()
{
    int c, n=0, stanje=VAN_RECI;
    while ((c=getchar()) != EOF)
    {
        switch(stanje)
        {
            case VAN_RECI:
                switch(c)
                {
                    case ' ':
                    case '\t':
                    case '\n':
                        break;
                    default:
                        n++;
                        stanje=U_RECI;
                        break;
                }
            break;
        }
    }
}
```

```
case U_RECI:
    switch(c)
    {
        case ' ':
        case '\t':
        case '\n':
            stanje=VAN_RECI;
            break;
        default:
            break;
    }
    break;
}
printf ("Broj reci: %d\n", n);
getche();
return 0;
}
```



Испис на екрану

9 ФУНКЦИЈЕ

9.1. Саставити функцију која врши сабирање два цела броја, а затим саставити програм који тестира функцију и исписује резултат.

Први начин:

```
#include <stdio.h>

/*Deklaracija i definicija funkcije*/
int zbir(int a, int b)
{
    int rezultat;
    rezultat=a+b;
    return rezultat;
}

/*Glavni program*/
main()
{
    int c;
    c=zbir(5,3); /*Poziv funkcije*/
    printf("\n%d\n", c);
    getch();
    return 0;
}
```

Други начин (скраћени запис):

```
#include <stdio.h>

/*Deklaracija i definicija funkcije*/
int zbir(int a, int b)
{
    return (a+b);
}

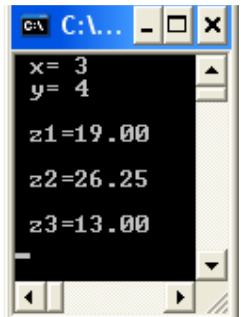
/*Glavni program*/
main()
{
    /*Poziv funkcije*/
    printf("\n%d\n", zbir(5,3));
    getch();
    return 0;
}
```



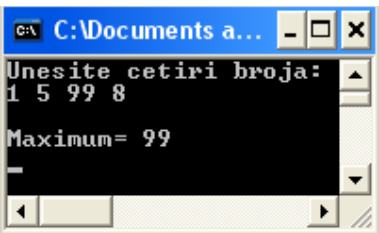
Испис на екрану

9.2. Саставити функције за рачунање збира, производа и количника два реална броја, као и функције за израчунавање квадрата и куба реалног броја. Затим саставити програм за израчунавање израза $z1 = x + y^2$, $z2 = x^3 - \frac{x}{y}$ и $z3 = (x * y) + (5 - y)$, користећи претходно формиране функције.

Променљиве x и y се уносе са тастатуре. Исписати резултате.

<pre>#include <stdio.h> /*Funkcija zbira*/ float zbir(float a, float b) { return(a+b); } /*Funkcija razlike*/ float razlika(float a, float b) { return(a-b); } /*Funkcija proizvoda*/ float proizvod(float a, float b) { return(a*b); } /*Funkcija kolicnika*/ float kolicnik(float a, float b) { if(b==0) return 0; else return(a/b); } /*Funkcija kvadrata*/ float kvadrat(float a) { return(a*a); }</pre>	<pre>/*Funkcija kuba*/ float kub(float a) { return(a*a*a); } /*Glavni program*/ main() { float x, y, z1, z2, z3; printf(" x= "); scanf("%f", &x); printf(" y= "); scanf("%f", &y); z1=zbir(x,kvadrat(y)); z2=razlika(kub(x),kolicnik(x,y)); z3=zbir(proizvod(x,y),razlika(5,y)); printf("\n z1=%.2f\n", z1); printf("\n z2=%.2f\n", z2); printf("\n z3=%.2f\n", z3); getch(); return 0; }</pre>  <p style="text-align: center;">Испис на екрану</p>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

9.3. Саставити функцију за одређивање максимума два цела броја, а затим саставити програм који ће за унета четири цела броја одредити највећи користећи претходну функцију, а затим исписати резултат.

<pre>#include <stdio.h> int max(int a, int b) { if(a==b) return 0; else if(a>b) return a; else return b; } main() { int a, b, c, d, m; printf("Unesite cetiri broja:\n"); scanf("%d %d %d %d", &a, &b, &c, &d); m=max(max(a,b),max(c,d)); if(m==0) printf("Zadati brojevi su isti."); else printf("\nMaximum= %d\n", m); getch(); return 0; }</pre>	 <p style="text-align: center;">Испис на екрану</p>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------

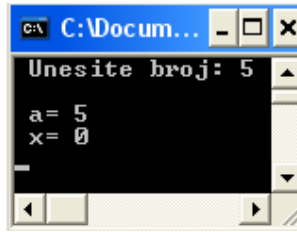
9.4. Шта се исписује на екрану након извршавања следећих програма.

a)

```
#include <stdio.h>

void fun1(int a)
{
    scanf("%d", &a);
    printf("\n a= %d\n", a);
}

main()
{
    int x=0;
    printf(" Unesite broj: ");
    fun1(x);
    printf(" x= %d\n", x);
    getche();
    return 0;
}
```



Испис на екрану а)

Пример приказује демонстрацију преноса параметра по вредности, где се пренети параметри не могу мењати. Функција покушава да прочита цео број. Због преноса параметра по вредности ово не успева.

б)

```
#include <stdio.h>

void fun2(int x)
{
    x*=2;
    x++;
}

main()
{
    int x=3;
    fun2(x);
    printf(" x= %d\n", x);
    getche();
    return 0;
}
```



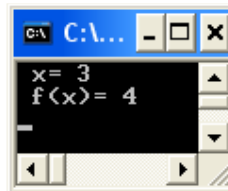
Испис на екрану б)

в)

```
#include <stdio.h>

int fun3(int x)
{
    return ++x;
}

main()
{
    int x=3;
    printf(" x= %d\n", x);
    printf(" f(x)= %d\n", fun3(x));
    getche();
    return 0;
}
```



Испис на екрану в)

9.5. Саставити програм за израчунавање површине и запремине лопте и исписивање резултата на основу унете вредности полупречника r . За рачунање запремине и површине, као и за испис резултата формирати одговарајуће функције.

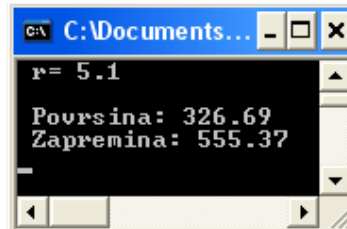
```
#include <stdio.h>
#define PI 3.14

double Povrsina(double r)
{
    return (4*r*r*PI);
}
```

```
double Zapremina(double r)
{
    return((4./3.)*r*r*r*PI);
}

void Ispis(double p, double v)
{
    printf("\n Povrsina: %.2f", p);
    printf("\n Zapremina: %.2f\n", v);
}

main()
{
    double r, p, v;
    printf(" r= ");
    scanf("%lf", &r);
    p=Povrsina(r);
    v=Zapremina(r);
    Ispis(p,v);
    getche();
    return 0;
}
```



Испис на екрану

9.6. Саставити програм за израчунавање површине троугла ако су задате координате његовог темена. Површину троугла рачунати помоћу следећих формула:

$$a = \sqrt{(x_B - x_C)^2 - (y_B - y_C)^2} \quad , \quad b = \sqrt{(x_C - x_A)^2 - (y_C - y_A)^2} \quad , \quad c = \sqrt{(x_A - x_B)^2 - (y_A - y_B)^2}$$

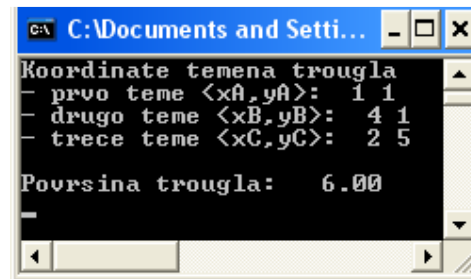
$$S = \frac{a+b+c}{2} \quad , \quad P = \sqrt{S(S-a)(S-b)(S-c)}$$

За израчунавање дужине страница **a, b и c** формирати одговарајућу функцију.

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

double Stranica(double x1, double x2, double y1, double y2)
{
    return(sqrt(pow(x1-x2,2)+pow(y1-y2,2)));
}

main()
{
    double xA, yA, xB, yB, xC, yC, a, b, c, s, P;
    printf("Koordinate temena trougla\n");
    printf("- prvo teme <xA,yA>: ");
    scanf("%lf%lf", &xA, &yA);
    printf("- drugo teme <xB,yB>: ");
    scanf("%lf%lf", &xB, &yB);
    printf("- trece teme <xC,yC>: ");
    scanf("%lf%lf", &xC, &yC);
    a=Stranica(xB,xC,yB,yC);
    b=Stranica(xC,xA,yC,yA);
    c=Stranica(xA,xB,yA,yB);
    s=(a+b+c)/2;
    P=sqrt(s*(s-a)*(s-b)*(s-c));
    printf("\nPovrsina trougla: %.2f\n", P);
    getche();
    return 0;
}
```



Испис на екрану

9.7. Саставити програм за израчунавање и исписивање суме квадрата свих бројева, само парних и само непарних за унете границе интервала. Користити функције за израчунавање збира квадрата.

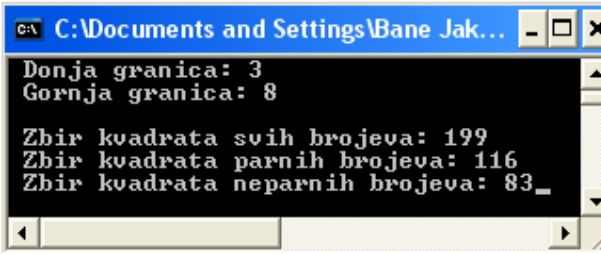
```
#include <stdio.h>

int ZbirKv(int a, int b)
{
    int i, suma=0;
    for(i=a; i<=b; i++)
        suma+=i*i;
    return suma;
}

int ZbirKvP(int a, int b)
{
    int i, suma=0;
    for(i=a; i<=b; i++)
    {
        if(i%2==0)
            suma+=i*i;
    }
    return suma;
}

int ZbirKvN(int a, int b)
{
    int i, suma=0;
    for(i=a; i<=b; i++)
    {
        if(i%2 == 1)
            suma += i*i;
    }
    return suma;
}

main()
{
    int a, b;
    printf(" Donja granica: "); scanf("%d", &a);
    printf(" Gornja granica: "); scanf("%d", &b);
    printf("\n Zbir kvadrata svih brojeva: %d", ZbirKv(a, b));
    printf("\n Zbir kvadrata parnih brojeva: %d", ZbirKvP(a, b));
    printf("\n Zbir kvadrata neparnih brojeva: %d", ZbirKvN(a, b));
    getche();
    return 0;
}
```



Испис на екрану

9.8. Саставити програм који исписује суму цифара за унете границе интервала. За рачунање суме цифара формирати одговарајућу функцију.

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

int Suma(int a)
{
    int s=0;
    while (a!=0)
    {
```

```

        s+=a%10;
        a/=10;
    }
    return s;
}

main()
{
    int i, d, g;
    printf(" Donja granica= ");
    scanf("%d", &d);
    printf(" Gornja granica= ");
    scanf("%d", &g);
    for(i=d; i<=g; i++)
        printf("\n%4d\t%3d", i, Suma(fabs(i)));
    getch();
    return 0;
}

```

```

C:\Documents a...
Donja granica= 203
Gornja granica= 214

203      5
204      6
205      7
206      8
207      9
208     10
209     11
210      3
211      4
212      5
213      6
214      7

```

Испис на екрану

9.9. Саставити програм који коришћењем функција одређује највећи заједнички делилац и најмањи заједнички садржалац два природна броја. Програм треба да захтева унос све док не прочита нулу за један од два унета природна броја.

```

#include <stdio.h>

unsigned nzd (unsigned a, unsigned b)
{
    unsigned c;
    while(b != 0)
    {
        c=b;
        b=a%b;
        a=c;
    }
    return a;
}

unsigned nzs (unsigned a, unsigned b)
{
    return a*b/nzd(a,b);
}

main()
{
    unsigned a, b;
    while(1)
    {
        printf ("\n a= "); scanf ("%u", &a);
        printf (" b= ");   scanf ("%u", &b);
        if (a==0 || b==0) break;
        printf (" nzd= %u", nzd(a,b));
        printf ("\n nzs= %u", nzs(a,b));
        printf ("\n\n");
    }
    getch();
    return 0;
}

```

```

C:\...
a= 15
b= 50
nzd= 5
nzs= 150

a= 4
b= 34
nzd= 2
nzs= 68

a= 0
b= 0

```

Испис на екрану

9.10. Саставити функцију за рачунање факторијела, затим саставити програм који рачуна и исписује број комбинација $C_{n,k} = \frac{n!}{k!(n-k)!}$, за дато **n** и **k** помоћу функције за рачунање факторијела.

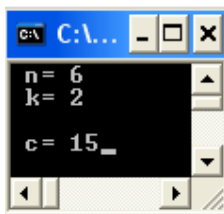
```
#include <stdio.h>
```

```
long Fakt(int n)
```

```
{
    long i, f=1;
    for(i=1; i<=n; i++)
        f *= i;
    return f;
}
```

```
main()
```

```
{
    int n, k, c;
    printf(" n= ");
    scanf("%d", &n);
    printf(" k= ");
    scanf("%d", &k);
    c=Fakt(n)/(Fakt(k)*Fakt(n-k));
    printf("\n c= %d", c);
    getche();
    return 0;
}
```



Испис на екрану

9.11. Саставити програм који за дато **n** рачуна и исписује суму $S = 1! + 2! + 3! + \dots + n!$. За рачунање факторијела користити одговарајућу функцију.

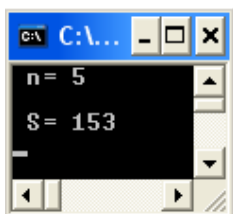
```
#include <stdio.h>
```

```
long Fakt(int n)
```

```
{
    long i, f=1;
    for(i=1; i<=n; i++)
        f *= i;
    return f;
}
```

```
main()
```

```
{
    long n, s=0;
    int i;
    printf(" n= ");
    scanf("%ld",&n);
    for(i=1; i<=n; i++)
        s+=Fakt(i);
    printf("\n S= %ld\n", s);
    getche();
    return 0;
}
```



Испис на екрану

9.12. Саставити програм којим се исписују сви троцифрени бројеви (ако их има) који су једнаки суми факторијела својих цифара.

```
#include <stdio.h>

long Fakt(int n)
{
    long i, f=1;
    for(i=1; i<=n; i++)
        f *= i;
    return f;
}

main()
{
    int a, b, c;
    for(a=1; a<=9; a++)
        for(b=0; b<=9; b++)
            for(c=0; c<=9; c++)
            {
                if((a*100+b*10+c)==(Fakt(a)+Fakt(b)+Fakt(c)))
                    printf("\n %d", a*100+b*10+c);
            }
    getche();
    return 0;
}
```



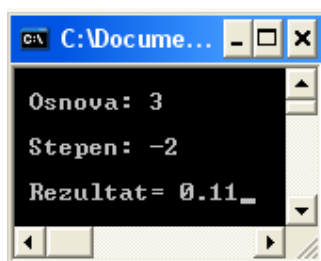
Испис на екрану

9.13. Саставити функцију за степеновање бројева. Затим саставити програм који помоћу формиране функције врши рачунање за дати број и степен и исписује резултат.

```
#include <stdio.h>

double Stepen(double x, int n)
{
    int i, negative;
    double s=1.0;
    negative = n<0;
    if(negative) n=-n;
    for(i=0; i<n; i++)
        s*=x;
    if(negative) return(1/s);
    else return(s);
}

main()
{
    int n;
    double x, s;
    printf("\n Osnova: ");
    scanf("%lf", &x);
    printf("\n Stepen: ");
    scanf("%d", &n);
    s=Stepen(x,n);
    printf("\n Rezultat= %.2f", s);
    getche();
    return 0;
}
```



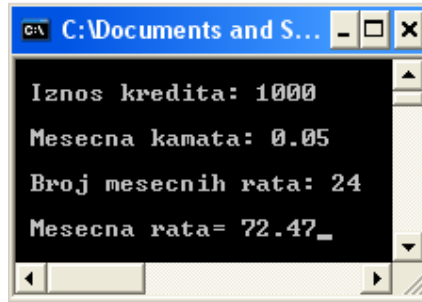
Испис на екрану

- 9.14.** Саставити програм за рачунање и исписивање месечне рате за отплату кредита по формули $r = \frac{A \cdot p \cdot k}{A - 1}$, где је $A = (1 + k)^m$, **p** – позајмица (износ кредита), **k** – месечна камата (нпр, за 6% уноси се 0.06), **m** – број месечних рата. За рачунање вредности **A** формирати одговарајућу функцију.

```
#include <stdio.h>

double Stepen(double x, int n)
{
    int i;
    double s=1.0;
    for(i=0; i<n; i++)
        s*=x;
    return(s);
}

main()
{
    int m;
    double p, k, r, A;
    printf("\n Iznos kredita: ");    scanf("%lf", &p);
    printf("\n Mesecna kamata: ");    scanf("%lf", &k);
    printf("\n Broj mesecnih rata: ");    scanf("%d", &m);
    A=Stepen(1+k,m);
    r=(A*p*k)/(A-1);
    printf("\n Mesecna rata= %.2f", r);
    getche();
    return 0;
}
```



Испис на екрану

- 9.15.** Саставити програм који исписује све просте бројеве мање од 500. Користити функцију за одређивање простих бројева.

```
#include <stdio.h>

int Prost(int broj)
{
    int i;
    for(i=2; i<broj; i++)
        if(broj%i == 0)
            return 0;
    return 1;
}

main()
{
    int i;
    for(i=1; i<=500; i++)
        if(Prost(i) == 1)
            printf("%d\t", i);
    getche();
    return 0;
}
```

Пошто у **C**-у не постоји логички тип података, уколико се ради о некој функцији која треба да врати вредност тачно или нетачно, као што је случај са овом, тада је она типа **int** и враћа 1 ако треба да врати тачно, а 0 ако треба да врати нетачно. Из тог разлога ова функција је типа **int** и у зависности од (не)испуњења услова враћа се вредност 0 или вредност 1.

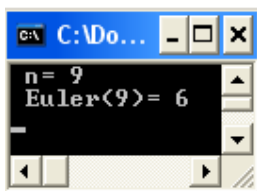
1	2	3	5	7	11	13	17	19	23
29	31	37	41	43	47	53	59	61	67
71	73	79	83	89	97	101	103	107	109
113	127	131	137	139	149	151	157	163	167
173	179	181	191	193	197	199	211	223	227
229	233	239	241	251	257	263	269	271	277
281	283	293	307	311	313	317	331	337	347
349	353	359	367	373	379	383	389	397	401
409	419	421	431	433	439	443	449	457	461
463	467	479	487	491	499	-			

Испис на екрану

9.16. Саставити програм који рачуна и испишује вредност Ојлерове функције позитивног целог броја унетог са улаза. Под Ојлеровом функцијом $\phi(n)$ подразумевамо број бројева m , таквих да је $1 \leq m < n$ и да су m и n узајамно прости. Ојлерова функција, као и $NZD(m, n)$ рачунати у посебним функцијама.

```
#include <stdio.h>

int nzd (int a, int b)
{
    int c;
    while(b != 0)
    {
        c=b;
        b=a%b;
        a=c;
    }
    return a;
}
```



Испис на екрану

```
int Ojler(int n)
{
    int br=0, m;
    /*Prolazimo kroz sve prirodne brojeve m
    koji su manji od n, i za svaki koji je
    uzajamno prost sa n uvecavamo brojac*/
    for(m = 1; m < n; m++)
        if(nzd(n,m) == 1) br++;
    /*Vracamo broj uzajamno prostih brojeva
    sa brojem n*/
    return br;
}

main()
{
    int n;
    printf(" n= ");
    scanf("%d", &n);
    printf(" Euler(%d)= %d\n", n, Ojler(n));
    getche();
    return 0;
}
```

9.17. Саставити програм којим се испишују сви троцифрени Амстронгови бројеви. Троцифрени број је Амстронгов ако је једнак суми кубова својих цифара. Формирати две функције, једна за рачунање суме кубова, а друга за одређивање да ли је дати број Амстронгов (ако јесте враћа 1, ако није враћа 0).

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

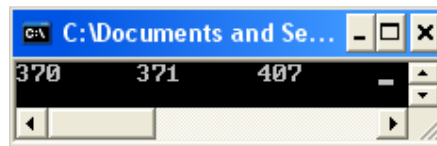
int SumaKubova(int n)
{
    int suma=0 ;
    for( ; n>0; n/=10)
        suma += pow(n%10,3);
    return suma;
}
```

```

int Amstrongov(int n)
{
    if(n==SumaKubova(n))
        return 1;
    return 0;
}

main()
{
    int i;
    for(i=100; i<=999; i++)
        if(Amstrongov(i))
            printf("%d\t", i);
    getche();
    return 0;
}

```



Испис на екрану

9.18. Саставити програм којим се исписују сви Нивенови бројеви друге стотице. Нивенов број је број који је дељив са сумом својих цифара. Формирати две функције, једна за рачунање суме цифара, а друга за одређивање да ли је дати број Нивенов (ако јесте враћа 1, ако није враћа 0).

```

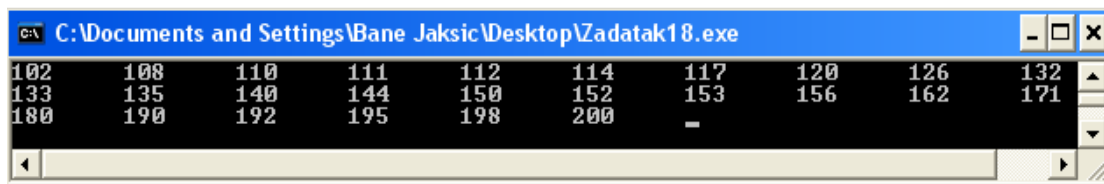
#include <stdio.h>
#include <math.h>

int SumaCifara(int n)
{
    int suma=0 ;
    for( ; n>0; n/=10)
        suma += n%10;
    return suma;
}

int Nivenov(int n)
{
    if(n % SumaCifara(n) == 0)
        return 1;
    return 0;
}

main()
{
    int i;
    for(i=101; i<=200; i++)
        if(Nivenov(i))
            printf("%d\t", i);
    getche();
    return 0;
}

```



Испис на екрану

9.19. Кондензатор капацитета C се пуни из једносмерног извора напона U_0 преко отпорника отпора R . Напон на кондензатору временски расте по релацији $u_c(t) = U_0 \left(1 - e^{-\frac{t}{RC}}\right)$ док напон на отпорнику опада по изразу $u_r(t) = U_0 - u_c(t)$. Саставити програм који ће учитати вредност капацитета C у микрофарадима, отпора R у килоомима и напона U_0 у волтима и израчунати и исписати вредности напона на кондензатору и отпорнику за t од 0 до $R \cdot C$ са кораком $R \cdot C / 10$.

```

#include <stdio.h>
#include <math.h>

float R, C, U0;

float uc(float t)
{
    return U0*(1-exp(-t/(R*C)));
}

float ur(float t)
{
    return U0-uc(t);
}

main()
{
    float t;
    printf("\n R[kOhm]= ");
    scanf("%f",&R);
    printf("\n C[uF]= ");
    scanf("%f",&C);
    printf("\n U0[V]= ");
    scanf("%f",&U0);
    R*=1e3;
    C*=1e-6;
    printf("\n   Vreme       N a p o n   n a");
    printf("\n   [ms]      kondenzatoru  otporniku");
    for(t=0; t<=R*C; t=t+R*C/10)
        printf("\n%7.2f    %9.3f    %9.3f",t*1e3,uc(t),ur(t));
    getch();
    return 0;
}

```

Vreme [ms]	N a p o n kondenzatoru	n a otporniku
0.00	0.000	3.000
0.20	0.285	2.715
0.40	0.544	2.456
0.60	0.778	2.222
0.80	0.989	2.011
1.00	1.180	1.820
1.20	1.354	1.646
1.40	1.510	1.490
1.60	1.652	1.348
1.80	1.780	1.220
2.00	1.896	1.104

Испис на екрану

9.20. Саставити програм који ће табеларно приказати вредности функција $f_1(x) = \frac{\sin(x)}{x}$ и

$f_2(x) = \left(\frac{\sin(x)}{x}\right)^2 = (f_1(x))^2$. Променљива x треба да узима вредности од 1 до уčitаног **xmin**

(**xmin<1**) тако да се свако наредно x смањи за уčitану вредност корака **dx** (**dx<0.1**).

```

#include <stdio.h>
#include <math.h>

float x, xmin, dx;

float f1(float x)
{
    return sin(x)/x;
}

float f2(float x)
{
    return f1(x)*f1(x);
}

main()

```

x	f1(x)	f2(x)
1.00	0.84147	0.70807
0.98	0.84745	0.71817
0.96	0.85332	0.72816
0.94	0.85910	0.73806
0.92	0.86478	0.74785
0.90	0.87036	0.75753

Испис на екрану


```

{
    printf("\n xmin[xmin<1] = ");
    scanf("%f",&xmin);
    printf("\n dx[xkor<0.1] = ");
    scanf("%f",&dx);
    printf("\n x      f1(x)      f2(x)");
    for(x=1; x>=xmin; x-=dx)
        printf("\n%5.2f%9.5f%9.5f",x,f1(x),f2(x));
    getch();
    return 0;
}

```

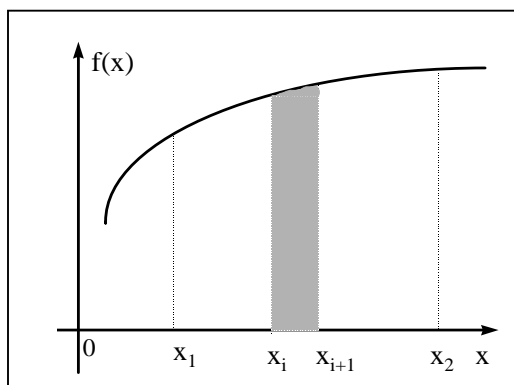
9.21. Трапезна метода за рачунање површине:

Површина испод криве $y = f(x)$ у интервалу од x_1 до x_2 може се приближно израчунати трапезном методом тако, да се цела површина апроксимира одређеним бројем трапеза (као што је приказано на слици). Површина појединачног трапеза је

$$P_i = \frac{f(x_i) + f(x_i + dx)}{2} \cdot dx.$$

Када се површине свих n трапеза саберу добија се

$$\text{укупна површина } P = \left(\frac{f(x_1)}{2} + \frac{f(x_2)}{2} + S \right) \cdot dx$$



где је S сума ордината $y(x_i)$ за свако x_i од x_1+dx до x_2-dx . То произлази због тога што се при рачунању свих површина ординате $f(x_i)$ узимају два пута, као задња страница i -тог трапеза и прва страница $i+1$ -ог трапеза, док су $f(x_1)$ и $f(x_2)$ странице првог и последњег трапеза.

Саставити програм који ће трапезном методом наћи и ишисати површину испод криве $y = a \cdot x \cdot |\sin(b \cdot x)|$ у интервалу од задатог x_1 до задатог x_2 . Апроксимирати површину помоћу n трапеза. Параметре a и b учитати као реалне бројеве, а број трапеза n као цели број.

```

#include <stdio.h>
#include <math.h>

float a, b;

float f(float x)
{
    return(a*x*fabs(sin(b*x)));
}

main()
{
    int n;
    float x, x1, x2, dx, S, P;
    printf(" Parametar a= "); scanf("%f",&a);
    printf(" Parametar b= "); scanf("%f",&b);
    printf(" Broj tacaka n= "); scanf("%d",&n);
    printf(" Pocetak intervala x1= "); scanf("%f",&x1);
    printf(" Kraj intervala x2= "); scanf("%f",&x2);
    dx=(x2-x1)/n;
    S=(f(x1)+f(x2))/2;

```

Ишис на екрану

```

    for(x=x1+dx; x<=x2-dx; x+=dx)
        S+=f(x);
    P=S*dx;
    printf("\n Povrsina= %.4f",P);
    getch();
    return 0;
}

```

9.22. Напон на неком потрошачу се мења временски по функцији $u(t) = e^{-b \cdot t} \cdot \sin(a \cdot t)$ док отпор потрошача зависи од напона по функцији $r(u) = r_0 + \ln\left(1 + \frac{|u|}{10}\right)$. Саставити програм који ће учитати параметре **a**, **b** и **r₀**, као и почетно и крајње време **t₁** и **t₂** и наћи и исписати енергију утрошену на потрошачу у временском интервалу од **t₁** до **t₂**.

Енергија је временски интеграл напона и струје, па треба наћи површину испод криве

$$f(t) = \frac{(u(t))^2}{r(t)} \text{ у интервалу од } t_1 \text{ до } t_2.$$

```

#include <stdio.h>
#include <math.h>

float a, b, r0;

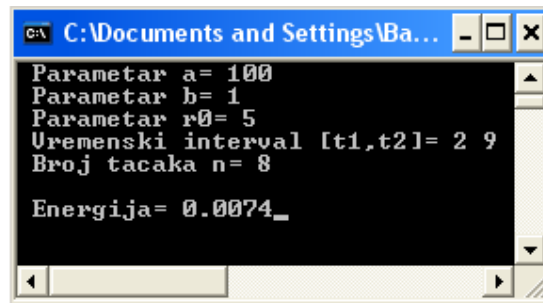
float u(float t)
{
    return exp(-b*t)*sin(a*t);
}

float r(float t)
{
    return r0+log(1+fabs(u(t)/10));
}

float f(float t)
{
    return u(t)*u(t)/r(t);
}

main()
{
    int n;
    float t1, t2, t, dt, E;
    printf(" Parametar a= "); scanf("%f",&a);
    printf(" Parametar b= "); scanf("%f",&b);
    printf(" Parametar r0= "); scanf("%f",&r0);
    printf(" Vremenski interval [t1,t2]= "); scanf("%f%f",&t1,&t2);
    printf(" Broj tacaka n= "); scanf("%d",&n);
    dt=(t2-t1)/n;
    E=f(t1)+f(t2);
    for(t=t1+dt; t<=t2-dt; t+=dt)
        E+=2*f(t);
    E=2*E*dt;
    printf("\n Energija= %.4f",E);
    getch();
    return 0;
}

```



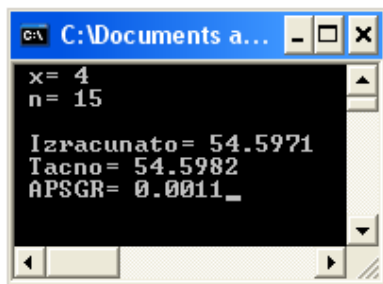
Испис на екрану

9.23. Саставити функцију која израчунава e^x на основу првих неколико (**n**) чланова Тејлоровог реда $e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots$, а затим саставити главни програм који за дато **n** и **x** користећи претходну функцију рачуна и исписује вредност израза, а затим одређује апсолутну грешку између добијене вредности и тачне вредности (за тачну вредност користити уграђену функцију **exp()** за исто **x**).

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

double Ekspon (double x, int n)
{
    double ex=1, clan=1;
    int i;
    for(i=1; i<n; i++)
    {
        clan *= x/i;
        ex += clan;
    }
    return ex;
}

main()
{
    double x, ex, ext, apsg;
    int n;
    printf (" x= ");
    scanf ("%lf", &x);
    printf (" n= ");
    scanf ("%d", &n);
    ex=Ekspon(x,n); /*Izracunata vrednost*/
    ext=exp(x);      /*Tacna vrednost*/
    apsg = fabs(ex-ext);
    printf("\n Izracunato= %.4f", ex);
    printf("\n Tacno= %.4f", ext);
    printf("\n APSGR= %.4f", apsg);
    getch();
    return 0;
}
```



Испис на екрану

9.24. Саставити програм за рачунање суме $S = \binom{n}{k} - \binom{n}{k+1} + \binom{n}{k+2} - \dots + (-1)^t \binom{n}{k+t}$ и исписивање њене вредности за дато **n**, **k** и **t** употребом функције за рачунање комбинација $\binom{n}{k}$.

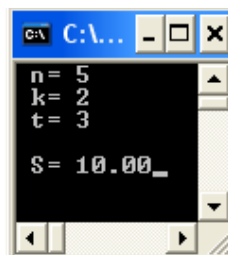
```
#include <stdio.h>

int Komb(int n, int m)
{
    int i, p=1;
    for(i=1; i<=m; i++)
        p *= (n-i+1)/i;
    return p;
}
```

```

main()
{
    float s;
    int n, k, i, t, znak=1;
    printf (" n= ");
    scanf ("%d", &n);
    printf (" k= ");
    scanf ("%d", &k);
    printf (" t= ");
    scanf ("%d", &t);
    for(i=0; i<t; i++)
    {
        s += znak*Komb(n,k);
        znak=-znak;
    }
    printf("\n S= %.2f", s);
    getch();
    return 0;
}

```



Испис на екрану

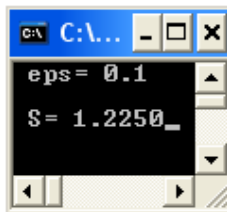
9.25. Саставити програм којим се рачуна сума $S = 1 + \frac{1}{2!!} - \frac{1}{3!!} + \frac{1}{4!!} - \dots$ рачунајући до члана који је по апсолутној вредности мањи од **eps**. Двоструки факторијел рачунати коришћењем функције:

$$dfakt(n) = n!! = \begin{cases} n \cdot (n-2) \cdot \dots \cdot 3 & \text{n непарно} \\ n \cdot (n-2) \cdot \dots \cdot 4 \cdot 2 & \text{n парно} \end{cases}$$

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

```
int DFakt(int n)
```

```
{
    int p=1;
    while(n>=2)
    {
        p=p*n;
        n=n-2;
    }
    return p;
}
```



Испис на екрану

```

main()
{
    int i=2, znak=1;
    float s=1, clan=1, eps;
    printf (" eps= ");
    scanf ("%f", &eps);
    printf("\n");
    while(fabs(clan)>=eps)
    {
        clan=(float)znak/DFakt(i);
        s=s+clan;
        i++;
        znak=-znak;
    }
    printf(" S= %.4f", s);
    getch();
    return 0;
}

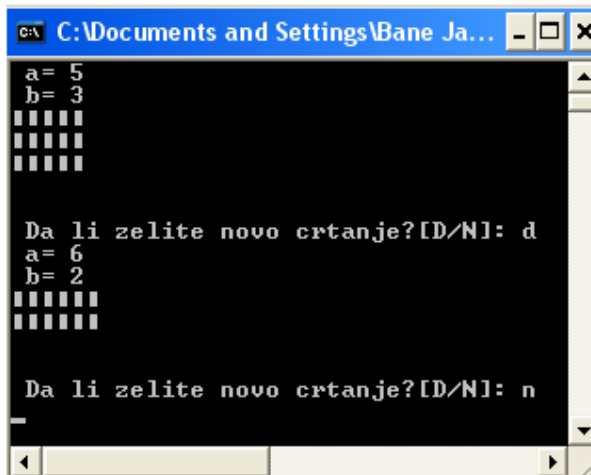
```

9.26. Саставити програм који помоћу функције исцртава правоугаоник дужине **a** и ширине **b** од знакова чији је ASCII код 254. После цртања правоугаоника поставља се питање "Da li zelite novo crtanje?[D/N]:" и понавља исти поступак.

```
#include <stdio.h>

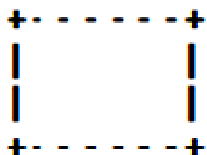
void Crtaj(int a,int b)
{
    int i, j;
    for(i=0; i<b; i++)
    {
        for(j=0; j<a; j++)
            printf("%c",254);
        printf("\n");
    }
}

main()
{
    int a, b, ch;
    ch='D';
    while(ch=='D')
    {
        printf(" a= ");
        scanf("%d",&a);
        printf(" b= ");
        scanf("%d",&b);
        Crtaj(a,b);
        printf("\n\n Da li zelite novo crtanje?[D/N]: ");
        ch=getchar();
        ch=toupper(ch);
        while(ch!='D' && ch!='N')
        {
            ch=getchar();
            ch=toupper(ch);
        }
    }
    getch();
    return 0;
}
```



Испис на екрану

9.27. Саставити програм који ће исцртати оквир приказан на слици за унету дужину и ширину. За цртање вертикалних и хоризонталних делова оквира формирати одговарајуће функције. На слици је приказан оквир дужине 6 и ширине 2.



```

#include <stdio.h>
#define HLIN '-'
#define VLIN '|'
#define UGAO '+'
#define PRAZNO ' '

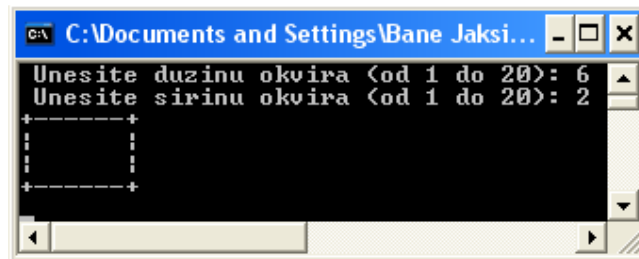
void Znakovi(int n, char c)
{
    while(n>0)
    {
        n--;
        putchar(c);
    }
}

void Hokvir(int k)
{
    putchar(UGAO);
    Znakovi(k, HLIN);
    putchar(UGAO);
    putchar('\n');
}

void Vokvir(int k)
{
    putchar(VLIN);
    Znakovi(k, PRAZNO);
    putchar(VLIN);
    putchar('\n');
}

main()
{
    int i, duzina, sirina;
    do
    {
        printf(" Unesite duzinu okvira (od 1 do 20): ");
        scanf("%d", &duzina);
    }
    while(duzina<1 || duzina>20);
    do
    {
        printf(" Unesite sirinu okvira (od 1 do 20): ");
        scanf("%d", &sirina);
    }
    while(sirina<1 || sirina>20);
    Hokvir(duzina); /*Stampanje gornjeg dela okvira*/
    for(i=0; i<sirina; i++)
        Vokvir(duzina); /*Stampanje srednjeg dela okvira*/
    Hokvir(duzina); /*Stampanje donjeg dela okvira*/
    getche();
    return 0;
}

```



Испис на екрану

9.28. а) Саставити функцију која врши конверзију малих слова у велика. Затим тестирати функцију за унети текст. Унос текста се завршава сигналом EOF.

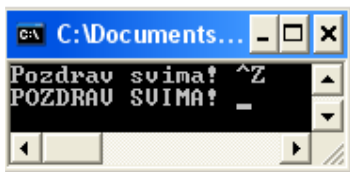
б) Саставити функцију која врши конверзију великих слова у мала. Затим тестирати функцију за унети текст. Унос текста се завршава сигналом EOF.

а)

```
#include <stdio.h>

char UVeliko(char c)
{
    if ('a'<=c && c<='z')
        return c-'a'+'A';
    return c;
}

main()
{
    int c;
    while((c=getchar()) != EOF)
        putchar(UVeliko(c));
    getche();
    return 0;
}
```



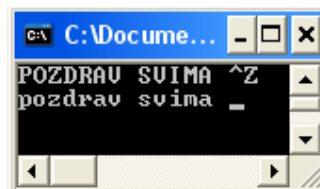
Испис на екрану

б)

```
#include <stdio.h>

char UMalo(char c)
{
    if ('A'<=c && c<='Z')
        return c-'A'+'a';
    return c;
}

main()
{
    int c;
    while((c=getchar()) != EOF)
        putchar(UMalo(c));
    getche();
    return 0;
}
```



Испис на екрану

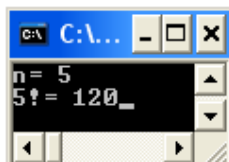
9.1 Рекурзивне функције

9.29. Саставити рекурзивну функцију за одређивање факторијела целог броја, а затим је тестирати у главном програму за унети цео број **n** и исписати добијени резултат.

```
#include <stdio.h>

int Faktorijel(int n)
{
    if(n == 1) return 1;
    return n*Faktorijel(n-1);
}

main()
{
    int n;
    printf("n= ");
    scanf("%d", &n);
    printf("%d!= %d", n, Faktorijel(n));
    getche();
    return 0;
}
```



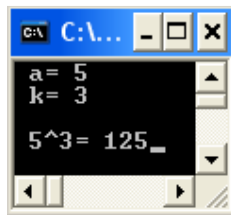
Испис на екрану

9.30. Саставити рекурзивну функцију која степеновање целог броја на целобројни изложилац, а затим тестирати функцију за дати природни број и изложилац и исписати добијени резултат.

```
#include <stdio.h>

int Stepen(int a, int k)
{
    if(k==0)
        return 1;
    else
        return a*Stepen(a,k-1);
}

main()
{
    int a,k;
    printf(" a= ");
    scanf("%d", &a);
    printf(" k= ");
    scanf("%d", &k);
    printf("\n %d^%d= %d", a, k, Stepen(a,k));
    getch();
    return 0;
}
```



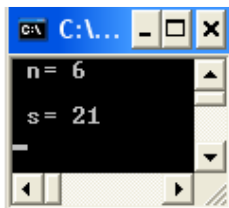
Испис на екрану

9.31. Саставити рекурзивну функцију која врши сабирање првих **n** бројева, а затим тестирати функцију за дато **n** и исписати добијени резултат.

```
#include <stdio.h>

int Suma(int n)
{
    if(n==0)
        return 0;
    else
        return(n + Suma(n-1));
}

main()
{
    int s,n;
    printf(" n= ");
    scanf("%d", &n);
    s=Suma(n);
    printf("\n s= %d\n",s);
    getch();
    return 0;
}
```



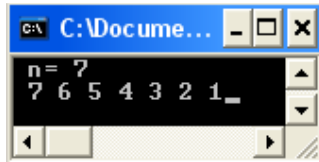
Испис на екрану

9.32. Саставити рекурзивну функцију која исписује првих **n** бројева у обрнутом редолседу, а затим тестирати функцију за дато **n**.

```
#include <stdio.h>

void Stampaj(unsigned int n)
{
    if(n == 0) return;
    printf(" %d", n);
    Stampaj(n-1);
}

main()
{
    int n;
    printf(" n= ");
    scanf("%d",&n);
    Stampaj(n);
    getch();
    return 0;
}
```



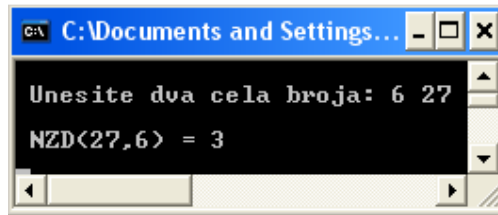
Испис на екрану

9.33. Саставити рекурзивну функцију за одређивање највећег заједничког делилаца за два цела броја. Затим тестирати функцију у главном програму и исписати добијени резултат.

```
#include <stdio.h>

int NZD(int x, int y)
{
    if(x%y == 0) return y;
    return NZD(y, x%y);
}

main()
{
    int n, m, t;
    printf("\n Unesite dva cela broja: ");
    scanf("%d %d",&n, &m);
    if(n<m)
    {
        t=n;
        n=m;
        m=t;
    }
    printf("\n NZD(%d,%d) = %d\n", n, m,NZD(n,m));
    getch();
    return 0;
}
```



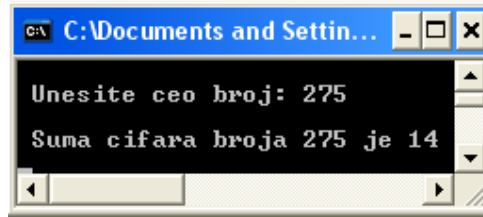
Испис на екрану

9.34. Саставити рекурзивну функцију која сабира декадне цифре целог броја. Затим тестирати функцију у главном програму за унети цео број и исписати добијени резултат.

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

int SaberiCifre(int x)
{
    if(x<0) x=abs(x);
    if(x<10) return x;
    return(x%10 + SaberiCifre(x/10));
}

main()
{
    int n;
    printf("\n Unesite ceo broj: ");
    scanf("%d",&n);
    printf("\n Suma cifara broja %d je %d\n", n, SaberiCifre(n));
    getch();
    return 0;
}
```



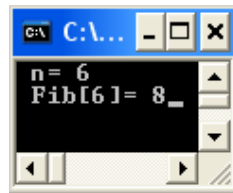
Испис на екрану

9.35. Саставити рекурзивну функцију која одређује и исписује **n**-ти члан Фибоначијевог низа, а затим тестирати функцију за дато **n**. Фибоначијев низ: $f_1=1, f_2=1, f_i=f_{i-1}+f_{i-2}, i=3, 4, 5, \dots$

```
#include <stdio.h>

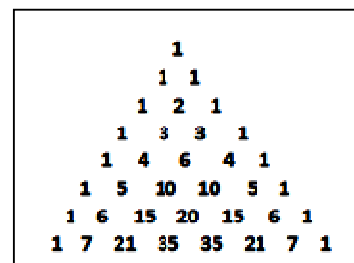
int Fib(int n)
{
    if((n==0) || (n==1))
        return 1;
    else
        return(Fib(n-1)+Fib(n-2));
}

main()
{
    int n;
    printf(" n= ");
    scanf("%d", &n);
    printf(" Fib[%d]= %d", n, Fib(n-1));
    getch();
    return 0;
}
```



Испис на екрану

9.36. Саставити рекурзивну функцију за рачунање биномног коефицијента. Затим саставити програм који користећи претходну функцију исписује Паскалов троугао за унети цео број **n**. Изглед Паскаловог троугла за **n=7** је приказан на следећој слици:

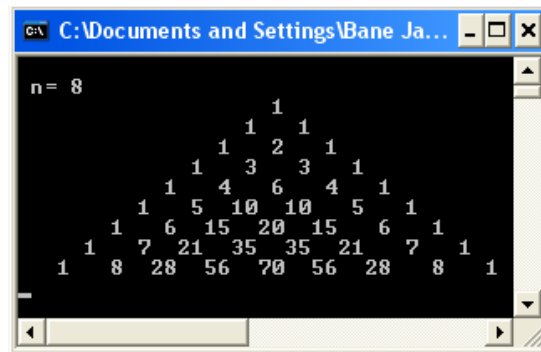


```
#include <stdio.h>

/*Rekurzivna funkcija za racunanje binomnog koeficijenta.*/
int BinKoeffREK(int n, int k)
{
    return (0<k && k<n) ? BinKoeffREK(n-1, k-1)+BinKoeffREK(n-1, k) : 1;
}

/*Iterativna funkcija za racunanje binomnog koeficijenta.*/
int BinKoeffIT(int n, int k)
{
    int i, j, b;
    for(b=i=1, j=n; i<=k; b=b*j--/i++);
    return b;
}

/*Glavni program.*/
main ()
{
    int n, i, j, k;
    printf("\n n= ");
    scanf("%d", &n);
    for(i=0; i<=n; i++)
    {
        for(j=0; j<n-i; j++)
            printf(" ");
        for(k=0; k<=i; k++)
            printf("%4d", BinKoeffREK(i, k));
        printf("\n");
    }
    getche();
    return 0;
}
```



Испис на екрану

9.37. Ханојске куле: Дата су три штапа, и на једном од њих **n** дискова различитих пречника, поређаних тако да мањи диск лежи на већем. Потребно је преместити све дискове на трећи штап у истом поретку, тако што се премешта један по један диск, коришћењем сва три штапа, и у сваком тренутку на сваком од штапова диск може да лежи само на већем диску.

Саставити рекурзивну функцију која омогућује пребацивање дискова по горе наведеном правилу, а затим тестирати функцију за унети број (**n**) дискова.

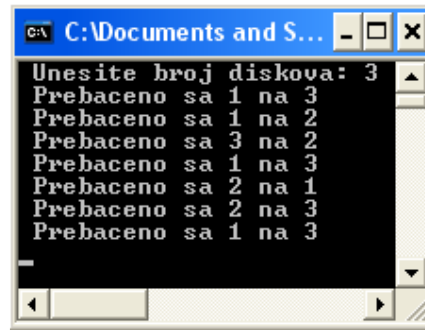
```
#include <stdio.h>

void Prebaci(int n, int i, int j)
{
    int k;
    /*Orediti pomocni stap k*/
    switch(i+j)
    {
        case 3: k=3; break;
        case 4: k=2; break;
        case 5: k=1; break;
    }
    if(n==1)
        printf(" Prebaceno sa %d na %d\n", i, j);
    else
    {

```

```
        Prebaci(n-1,i,k);
        Prebaci(1,i,j);
        Prebaci(n-1,k,j);
    }
    return;
}

main()
{
    int broj;
    printf(" Unesite broj diskova: ");
    scanf("%d", &broj);
    Prebaci(broj, 1, 3);
    getche();
    return 0;
}
```



Испис на екрану

10 НИЗОВИ

10.1 Основне конструкције програма са низовима

10.1. а) Која је разлика између следећа два низа:

```
int a[15] = {1,2,3};
```

```
int b[] = {1,2,3};
```

б) На који други начин можемо доделити вредности елементима низа?

в) Колико елемената и које вредности ће имати ти елементи низова **a** и **b** након извршавања следећег дела програма:

```
int a [10];
```

```
int b [10] = {0};
```

а) Променљиве (низови) **a** и **b** се разликују у дужини. Низ **a** има 15 елемената где прва три имају вредности 1, 2 и 3, док су остали имају вредности нуле. Низ **b** има три елемента са вредностима 1, 2 и 3.

б)

```
int a[17], b[3], i;  
a[0]=1; a[1]=2; a[2]=3;  
for(i=3;i<15;i++) a[i]=0;  
b[0]=1; b[1]=2; b[2]=3;
```

в) Низ **a** има 10 елемената чије су вредности непознате (цели бројеви). Низ **b** има 10 елемената и сви имају вредности нула.

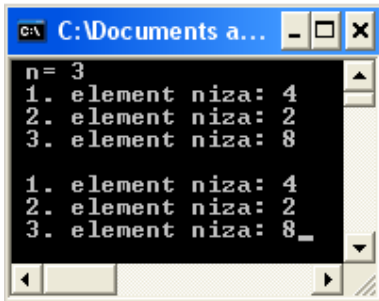
10.2. Шта се исписује на екрану након извршавања следећег програма:

```
#include <stdio.h>  
#define MAX 10  
  
main()  
{  
    int i, n, niz[MAX];  
    printf(" n= ");  
    scanf("%d", &n);  
    for(i=0; i<n; i++)  
    {  
        printf(" %d. element niza: ", i+1);
```

```

    scanf("%d", &niz[i]);
}
for(i=0; i<n; i++)
    printf("\n %d. element niza: %d", i+1, niz[i]);
getche();
return 0;
}

```



Испис на екрану

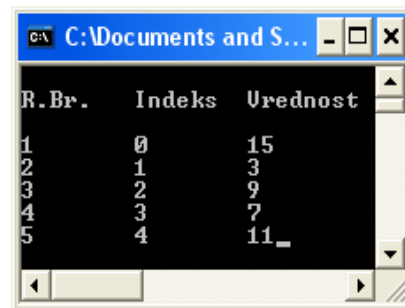
10.3. Шта се исписује на екрану након извршавања следећег програма:

```

#include <stdio.h>
#define MAX 5

main()
{
    int i, j, niz[]={15,3,9,7,11};
    printf("\nR.Br.\tIndeks\tVrednost\n");
    for(i=0; i<MAX; i++)
        printf("\n%d\t%d\t%d", i+1, i, niz[i]);
    getche();
    return 0;
}

```



Испис на екрану

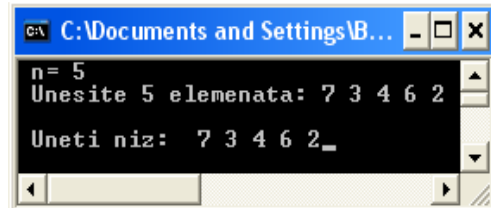
10.4. Саставити програм који ће учитати низ од **n** елемената и исписати их оним редоследом којим су учитани.

```

#include <stdio.h>
#define MAX 100

main()
{
    int niz[MAX], i,n;
    printf(" n= ");
    scanf("%d", &n);
    printf(" Unesite %d elemenata: ",n);
    for(i=0 ; i<n ; i++)
        scanf("%d", &niz[i]);
    printf("\n Uneti niz: ");
    for(i=0 ; i<n ; i++)
        printf(" %d", niz[i]);
    getche();
    return 0;
}

```

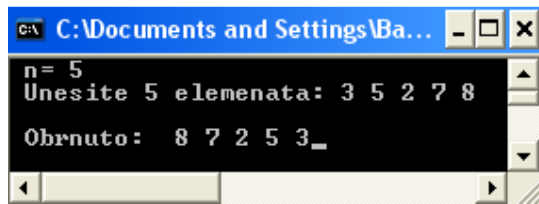


Испис на екрану

10.5. Саставити програм који ће учитати низ од **n** елемената и исписати их обрнутим редоследом.

```
#include <stdio.h>
#define MAX 100

main()
{
    int niz[MAX];
    int i,n;
    printf(" n= ");
    scanf("%d", &n);
    printf(" Unesite %d elemenata: ",n);
    for(i=0; i<n; i++)
        scanf("%d", &niz[i]);
    printf("\n Obrnuto: ");
    for(i=n-1; i>=0; i--)
        printf(" %d", niz[i]);
    getch();
    return 0;
}
```



Испис на екрану

10.6. Шта се исписује на екрану након извршавања следећих програма:

a)

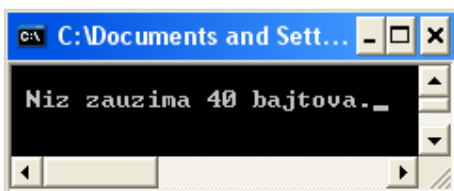
```
#include <stdio.h>
#define MAX 10

main()
{
    int niz[MAX];
    printf("\n Niz zauzima %d bajtova.", sizeof(niz));
    getch();
    return 0;
}
```

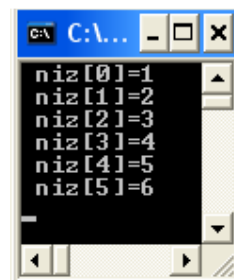
б)

```
#include <stdio.h>

main()
{
    int i, BrElem;
    int niz[] = {1, 2, 3, 4, 5, 6};
    BrElem = sizeof(niz)/sizeof(int);
    for(i=0; i<BrElem; i++)
        printf(" niz[%d]=%d\n",i, niz[i]);
    getch();
    return 0;
}
```



Испис на екрану a)



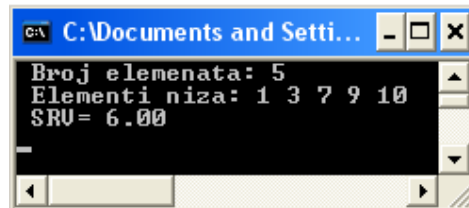
Испис на екрану

10.2 Операције са низовима и разврставање елемената

10.7. Саставити програм за израчунавање и исписивање аритметичке средине задатог низа (дужине **n**) целих бројева.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, n, niz[MAX], suma=0;
    printf(" Broj elemenata: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
    {
        scanf ("%d", &niz[i]);
        suma+=niz[i];
    }
    printf(" SRV= %.2f\n", (float)suma/n);
    getche();
    return 0;
}
```

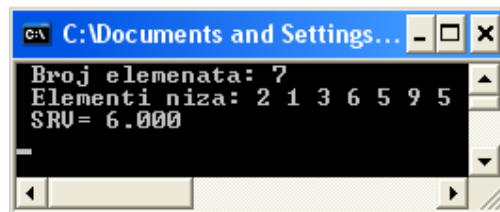


Испис на екрану

10.8. Саставити програм који за унети низ (дужине **n**) целих бројева израчунава и исписујеј аритметичку средину оних елемената низа који су дељиви са 3.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, n, k=0, niz[MAX];
    double suma=0;
    printf(" Broj elemenata: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
    {
        scanf("%d", &niz[i]);
        if(niz[i]%3 == 0)
        {
            suma+=niz[i];
            k++;
        }
    }
    printf(" SRV= %.3f\n", suma/k);
    getche();
    return 0;
}
```



Испис на екрану

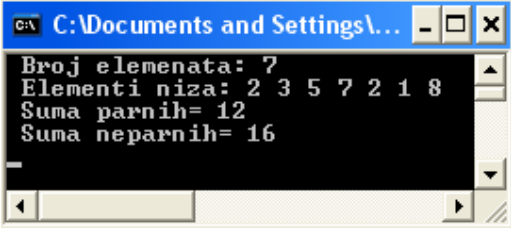
10.9. Саставити програм који ће за унети низ (дужине **n**) целих бројева одредити и исписати:

- а) суму парних и суму непарних бројева;
- б) суму елемената са парним индексима и суму елемената са непарним индексима.

а)

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, n, niz[MAX], sumap=0, suman=0;
    printf(" Broj elemenata: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
    {
        scanf("%d", &niz[i]);
        if(niz[i]%2==0)
            sumap += niz[i];
        else
            suman += niz[i];
    }
}
```

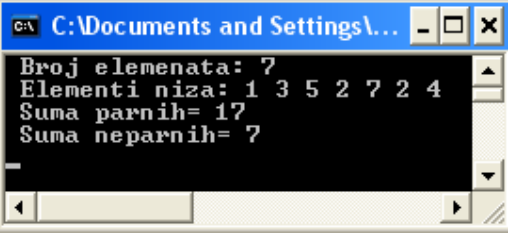


Испис на екрану

б)

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, n, niz[MAX], sumap=0, suman=0;
    printf(" Broj elemenata: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
    {
        scanf("%d", &niz[i]);
        if(i%2==0)
            sumap += niz[i];
        else
            suman += niz[i];
    }
}
```



Испис на екрану

10.10. Саставити програм који прочита два низа, дужине **n**, са реалним компонентама,

$A = (A_1, A_2, \dots, A_n)$ и $B = (B_1, B_2, \dots, B_n)$ израчунава њихов скаларни производ $s = \sum_{i=0}^{n-1} A_i B_i$ и

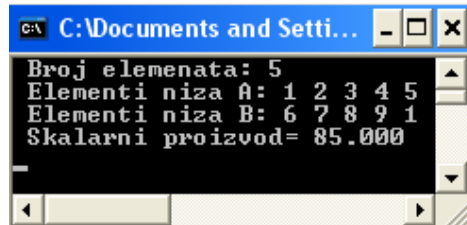
исписује добијени резултат.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, n;
    double proizvod=0, niz1[MAX], niz2[MAX];
    printf(" Broj elemenata: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
        scanf("%lf", &niz1[i]);
    printf(" Elementi niza B: ");
    for(i=0; i<n; i++)
        scanf("%lf", &niz2[i]);
    for(i=0; i<n; i++)
        proizvod+=niz1[i]*niz2[i];
    printf(" Skalarni proizvod= %.3f\n", proizvod);
    getche();
    return 0;
}

```



Испис на екрану

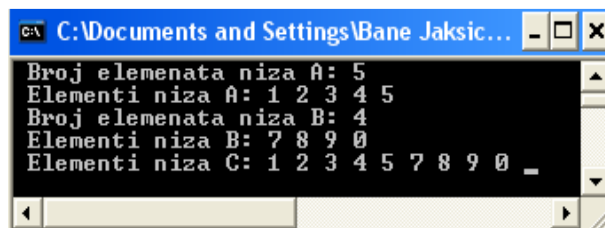
10.11. Саставити програм који учита елементе низа $A = (A_1, A_2, \dots, A_n)$ дужине **n** и низа $B = (B_1, B_2, \dots, B_m)$ дужине **m** и спаја у један низ $C = (A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m)$ дужине **n+m**. Исписати новокреирани низ.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, n, m, nizA[MAX], nizB[MAX], nizC[MAX];
    printf(" Broj elemenata niza A: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
        scanf("%d", &nizA[i]);
    printf(" Broj elemenata niza B: ");
    scanf("%d", &m);
    printf(" Elementi niza B: ");
    for(i=0; i<m; i++)
        scanf("%d", &nizB[i]);
    printf(" Elementi niza C: ");
    for(i=0; i<(n+m); i++)
    {
        if (i<n) nizC[i]=nizA[i];
        else nizC[i]=nizB[i-n];
        printf("%d ", nizC[i]);
    }
    getche();
    return 0;
}

```

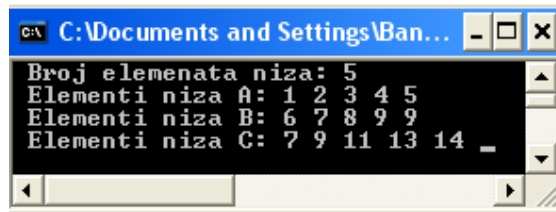


Испис на екрану

10.12. Саставити програм који учита елементе два низа дужине n $A = (A_1, A_2, \dots, A_n)$ и $B = (B_1, B_2, \dots, B_n)$ и формира и исписује нови низ чији су елементи $C = (A_1 + B_1, A_2 + B_2, \dots, A_n + B_n)$.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, n, nizA[MAX], nizB[MAX], nizC[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
        scanf("%d", &nizA[i]);
    printf(" Elementi niza B: ");
    for(i=0; i<n; i++)
        scanf("%d", &nizB[i]);
    printf(" Elementi niza C: ");
    for(i=0; i<n; i++)
    {
        nizC[i]=nizA[i]+nizB[i];
        printf("%d ", nizC[i]);
    }
    getche();
    return 0;
}
```



Испис на екрану

10.13. Саставити програм за формирање низа C од два задата низа реалних бројева A и B (сваки дужине 5) на следећи начин $C_i = \frac{(A_i)^3}{3} + 2A_iB_i$. Исписати низ C .

```
#include <stdio.h>
#define MAX 5

main()
{
    float nizA[MAX], nizB[MAX], nizC[MAX];
    int i;
    printf(" Elementi niza A: ");
    for(i=0; i<MAX; i++)
        scanf("%f", &nizA[i]);
    printf(" Elementi niza B: ");
    for(i=0; i<MAX; i++)
        scanf("%f", &nizB[i]);
    printf("\n Elementi niza C: ");
    for(i=0; i<MAX; i++)
    {
        nizC[i]=pow(nizA[i],3)/3+2*nizA[i]*nizB[i];
        printf("%.2f ", nizC[i]);
    }
    getche();
    return 0;
}
```

```

C:\Documents and Settings\Bane Jaksic\Desktop\Zadata...
Elementi niza A: 1 2 3 4 5
Elementi niza B: 2 4 3 8 7

Elementi niza C: 4.33 18.67 27.00 85.33 111.67 _

```

Испис на екрану

10.14. Саставити програм за формирање низа **C** од два задата низа целих бројева **A** и **B** (сваки дужине 5) на следећи начин: $A[0]+B[4], \dots, A[4]+B[0]$. Исписати низ **C**.

```

#include <stdio.h>
#define MAX 5

main()
{
    int nizA[MAX], nizB[MAX], nizC[MAX], i;
    printf (" Elementi niza A: ");
    for(i=0; i<MAX; i++)
        scanf ("%d", &nizA[i]);
    printf (" Elementi niza B: ");
    for(i=0; i<MAX; i++)
        scanf ("%d", &nizB[i]);
    printf (" Elementi niza C: ");
    for(i=0; i<MAX; i++)
    {
        nizC[i]=nizA[i]+nizB[MAX-1-i];
        printf ("%d ", nizC[i]);
    }
    getch();
    return 0;
}

```

```

C:\Documents and Settings\...
Elementi niza A: 1 2 3 4 5
Elementi niza B: 5 4 3 2 1
Elementi niza C: 2 4 6 8 10 _

```

Испис на екрану

10.15. Саставити програм који учита елементе низа $A = (A_1, A_2, \dots, A_n)$ дужине **n** и низа $B = (B_1, B_2, \dots, B_m)$ дужине **m**, а затим формира и исписује низ **C** који се састоји од парних елемената низа **A** и низа **B**.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, n, m, k=0, nizA[MAX], nizB[MAX], nizC[MAX];
    printf(" Broj elemenata niza A: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
        scanf("%d", &nizA[i]);
    printf(" Broj elemenata niza B: ");
    scanf("%d", &m);
    printf(" Elementi niza B: ");
    for(i=0; i<m; i++)
        scanf("%d", &nizB[i]);
    printf("\n Elementi niza C: ");
    for(i=0; i<n; i++)
        if(nizA[i]%2==0)

```

```

    nizC[k]=nizA[i];
    k++;
}
for(i=0; i<m; i++)
    if(nizB[i]%2==0)
    {
        nizC[k]=nizB[i];
        k++;
    }
for(i=0; i<k; i++)
    printf("%d ",nizC[i]);
getche();
return 0;
}

```

Испис на екрану

10.16. Саставити програм који ће учитати два низа целих бројева **A** и **B** једнаких дужина **n** и на основу њих формирати низ **C** тако да **i**-ти елемент низа **C** буде једнак мањем од **i**-тих елемената низа **A** и **B**. Ако су **i**-ти елементи низа **A** и **B** једнаки онда **i**-ти елемент низа **C** треба да добије вредност нула. Исписати низ **C**.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, n, nizA[MAX], nizB[MAX], nizC[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
        scanf("%d", &nizA[i]);
    printf(" Elementi niza B: ");
    for(i=0; i<n; i++)
        scanf("%d", &nizB[i]);
    printf("\n Elementi niza C: ");
    for(i=0; i<n; i++)
    {
        if (nizA[i]<nizB[i]) nizC[i]=nizA[i];
        else if(nizA[i]>nizB[i]) nizC[i]=nizB[i];
        else nizC[i]=0;
        printf("%d ",nizC[i]);
    }
    getche();
    return 0;
}

```

Испис на екрану

10.17. Саставити програм који за учитани низ целих бројева **A** дужине **n** формира и исписује два низа: низ **B** који садржи негативне елементе низа **A** и низ **C** који садржи позитивне елементе и нуле низа **A**.

```

#include <stdio.h>
#define MAX 100

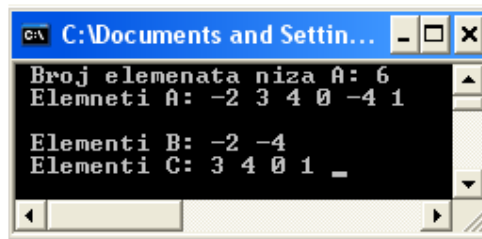
main()
{
    int nizA[MAX], nizB[MAX], nizC[MAX];
    int n, i, j=0, k=0;
    printf(" Broj elemenata niza A: ");
}

```

```

scanf("%d", &n);
printf(" Elemneti A: ");
for(i=0; i<n; i++)
    scanf ("%d", &nizA[i]);
for(i=0; i<n; i++)
{
    if (nizA[i]<0)
    {
        nizB[j]=nizA[i];
        j++;
    }
    else
    {
        nizC[k]=nizA[i];
        k++;
    }
}
printf("\n Elementi B: ");
for(i=0; i<j; i++)
    printf ("%d ", nizB[i]);
printf("\n Elementi C: ");
for(i=0; i<k; i++)
    printf ("%d ", nizC[i]);
getche();
return 0;
}

```



Испис на екрану

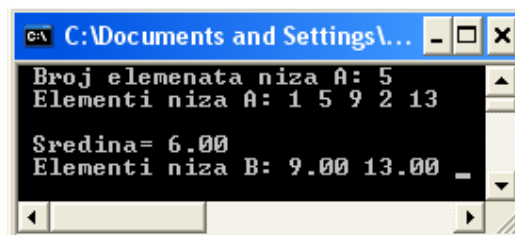
10.18. Саставити програм који за унети низ реланих бројева **A** дужине **n** формира и исписује нови низ **B** кога чине елементи низа **A** који су већи од аритемтичке средине свих елемената низа **A**.

```

#include <stdio.h>
#define MAX 100

main()
{
    float s=0, as, nizA[MAX], nizB[MAX];
    int i, j=0, n;
    printf(" Broj elemenata niza A: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
    {
        scanf("%f", &nizA[i]);
        s+=nizA[i];
    }
    as=s/n;
    printf("\n Sredina= %.2f", as);
    printf("\n Elementi niza B: ");
    for(i=0; i<n; i++)
    {
        if (nizA[i]>as)
        {
            nizB[j]=nizA[i];
            printf("%.2f ", nizB[j]);
            j++;
        }
    }
    getche();
    return 0;
}

```

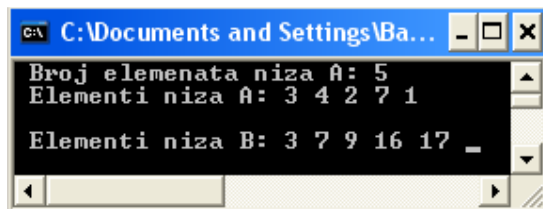


Испис на екрану

10.19. Саставити програм који за унети низ целих бројева **A**, дужине **n**, формира и исписује нови низ **B** чији се елементи формирају по следећем принципу: $B_0=A_0$, $B_1=A_0+A_1$, $B_2=A_0+A_1+A_2$, ..., $B_i=A_0+A_1+A_2+...+A_i$.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, n, s=0, nizA[MAX], nizB[MAX];
    printf(" Broj elemenata niza A: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
    {
        scanf("%d", &nizA[i]);
        s+=nizA[i];
        nizB[i]=s;
    }
    printf("\n Elementi niza B: ");
    for(i=0; i<n; i++)
        printf("%d ", nizB[i]);
    getch();
    return 0;
}
```



Испис на екрану

10.3 Низови и функције

10.20. Шта се исписује извршавањем следећег програмског кода:

```
#include <stdio.h>

void StampaNiz(int a[], int n)
{
    int i;
    for(i=0; i<n; i++)
        printf("%d ", a[i]);
    printf("\n");
    printf("sizeof(a) - u okviru fje : %d\n", sizeof(a));
}

main()
{
    int a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9};
    printf("sizeof(a) - u okviru main : %d\n", sizeof(a));
    StampaNiz(a, sizeof(a)/sizeof(int));
    getch();
    return 0;
}
```

```

C:\Documents and Settings\B...
sizeof(a) - u okviru main : 36
1 2 3 4 5 6 7 8 9
sizeof(a) - u okviru fje : 4

```

Испис на екрану

Низови се преносе тако што се пренесе адреса њиховог почетка. Низове је неопходно преносити заједно са димензијом низа (осим ниски карактера), јер тамо важи конвенција да се крај низа обележава знаком '\0'.

10.21. Саставити функцију за израчунавање скаларног производа два низа реалних бројева

$$s = \sum_{i=0}^{n-1} A_i B_i$$

а затим саставити главни програм који ће учитати два низа једнаких дужина **n** и

применом формирани функције исписати скаларни производ два низа.

```

#include <stdio.h>
#define MAX 100

double SkalPro(double a[], double b[], int n)
{
    double zbir=0;
    int i;
    for(i=0; i<n; i++)
        zbir+=a[i]*b[i];
    return zbir;
}

main()
{
    double nizA[MAX], nizB[MAX];
    int i, n;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
        scanf("%lf", &nizA[i]);
    printf(" Elementi niza B: ");
    for(i=0; i<n; i++)
        scanf("%lf", &nizB[i]);
    printf("\n A*B= %.2f", SkalPro(nizA, nizB, n));
    getch();
    return 0;
}

```

```

C:\Documents and Settings\B...
Broj elemenata niza: 5
Elementi niza A: 1 2 3 4 7
Elementi niza B: 3 2 1 1 6
A*B= 56.00_

```

Испис на екрану

10.22. Саставити функцију којом се одређује број различитих елемената у задатом целобројном низу. Затим саставити програм који чита низ целих бројева, и одређује број различитих елемената користећи претходну функцију и исписује резултат.

```

#include <stdio.h>
#define MAX 100

int Razliciti(const int niz[], int n)
{
    int i, j, brojac=0;
}

```

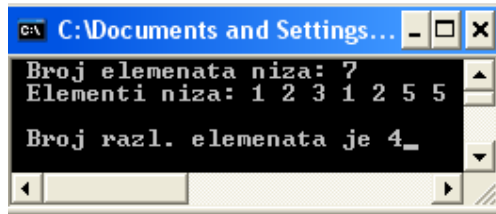


```

    for(i=0; i<n; i++)
    {
        for(j=0; j<i && niz[j]!=niz[i]; j++);
        if(j==i) brojac++;
    }
    return brojac;
}

main()
{
    int niz[MAX], n, i;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for (i=0; i<n; i++)
        scanf ("%d",&niz[i]);
    printf ("\n Broj razl. elemenata je %d", Razliciti(niz,n));
    getche();
    return 0;
}

```



Испис на екрану

10.23. Саставити програм за израчунавање статистике полагања испита која обухвата следеће функције: израчунавање укупне просечне оцене (сви који су полгали испит), израчунавање просечне оцене оних који су положили (оцена већа од 5), израчунавање броја који су положили испит, израчунавање броја који нису положили испит и израчунавање броја који имају оцену већу од просечне. У главном програму се уноси број студената и оцене студената у облику низа. Исписати добијене резултате

```

#include <stdio.h>
#define MAX 100

/*Izracunavanje prosečne ocene*/
float Prosek(int x[], int n)
{
    int i;
    float suma =0;
    for(i=0; i<n; i++)
        suma+=x[i];
    return(suma/n);
}

/*Izracunavanje proseka položenih*/
float ProsekPoloženih(int x[], int n)
{
    int i, j=0;
    float suma=0;
    for(i=0; i<n; i++)
        if(x[i]>5)
        {
            suma+=x[i];
            j++;
        }
    return(suma/j);
}

/*Izracunavanje broja položenih na ispitu*/
int BrojPoloženih (int x[], int n)
{

```

```

    int i, broj=0;
    for(i=0; i<n; i++)
        if(x[i]>5)
            broj++;
    return broj;
}

/*Izracunavanje broja nepolozenih na ispitu*/
int BrojNepolozenih (int x[], int n)
{
    int i, broj=0;
    for(i=0; i<n; i++)
        if(x[i] == 5)
            broj++;
    return broj;
}

/*Izracunavanje broja studenata iznad proseka*/
int BrojNadprosecnih (int x[], int y[], int n, float m)
{
    int i, j=0;
    for(i=0; i<n; i++)
    {
        if(x[i] > m)
        {
            y[j]=i;
            j++;
        }
    }
    return j;
}

/*Glavni program*/
main()
{
    int student[MAX], ocena[MAX];
    int i, n;
    printf(" Broj studenata: ");
    scanf("%d",&n);
    printf(" Ocene studenata [5 do 10]: \n");
    for(i=0; i<n; i++)
    {
        printf(" Ocena studenta %d = ",i);
        scanf("%d",&ocena[i]);
    }
    printf("\n Ukupno polozenih na ispitu: %d\n",
           BrojPolozenih(ocena,n));
    printf(" Ukupno nepolozenih na ispitu: %d\n",
           BrojNepolozenih(ocena,n));
    printf(" Prosek ocena na ispitu: %.2f\n", Prosek(ocena,n));
    printf(" Prosek ocena polozenih na ispitu: %.2f\n",
           ProsekPolozenih(ocena,n));
    printf(" Broj studenata iznad proseka: %d\n",
           BrojNadprosecnih(ocena, student, n, Prosek(ocena,n)));
    getch();
    return 0;
}

```

```

C:\Documents and Settings\Bane Jaksic\...
Broj studenata: 7
Ocene studenata [5 do 10]:
Ocena studenta 0 = 5
Ocena studenta 1 = 7
Ocena studenta 2 = 8
Ocena studenta 3 = 5
Ocena studenta 4 = 8
Ocena studenta 5 = 10
Ocena studenta 6 = 9
Ukupno polozenih na ispitu: 5
Ukupno nepolozenih na ispitu: 2
Prosek ocena na ispitu: 7.43
Prosek ocena polozenih na ispitu: 8.40
Broj studenata iznad proseka: 4

```

Испис на екрану

10.24. Саставити функцију која генерише првих n чланова Фибоначијевог низа и функцију која исписује чланове низа, а затим те функције тестирати у главном програму.

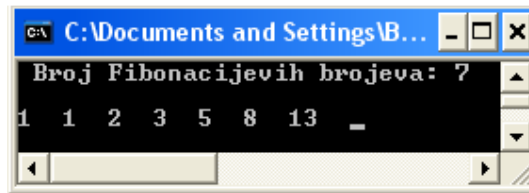
Фибоначијев низ: $f_1=1, f_2=1, f_i=f_{i-1}+f_{i-2}, i=3, 4, 5, \dots$

```
#include <stdio.h>
#define MAX 100

void Fibonaci(int niz[], int n)
{
    int i;
    for(i=0; i<n; i++)
        if(i<2) niz[i]=1;
        else niz[i]=niz[i-1]+niz[i-2];
}

void Prikazi(int niz[], int n)
{
    int i;
    for(i=0; i<n; i++)
        printf("%d ", niz[i]);
}

main()
{
    int niz[MAX], n;
    printf(" Broj Fibonacijevih brojeva: ");
    scanf("%d", &n);
    printf("\n");
    Fibonaci(niz, n);
    Prikazi(niz, n);
    getch();
    return 0;
}
```



Испис на екрану

10.25. Саставити рекурзивну функцију за израчунавање скаларног производа два низа реалних

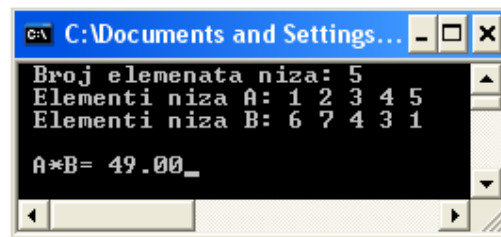
бројева $s = \sum_{i=0}^{n-1} A_i B_i$, а затим саставити главни програм који ће учитати два низа једнаких дужина n и

применом формирани функције исписати скаларни производ два низа.

```
#include <stdio.h>
#define MAX 100

float SkalPro (const float a[], const float b[], int n)
{
    return (n>0) ? a[0]*b[0]+SkalPro(a+1,b+1,n-1) : 0;
}

main()
{
    float nizA[MAX], nizB[MAX];
    int i, n;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
        scanf("%f", &nizA[i]);
    printf(" Elementi niza B: ");
```



Испис на екрану

```

    for(i=0; i<n; i++)
        scanf("%f",&nizB[i]);
    printf("\n A*B= %.2f", SkalPro(nizA,nizB,n));
    getch();
    return 0;
}

```

10.26. Саставити рекурзивну функцију која испишује све пермутације скупа $\{1, 2, \dots, n\}$. Затим тестирати функцију у главном програму за унуту дужину пермуатција n .

```

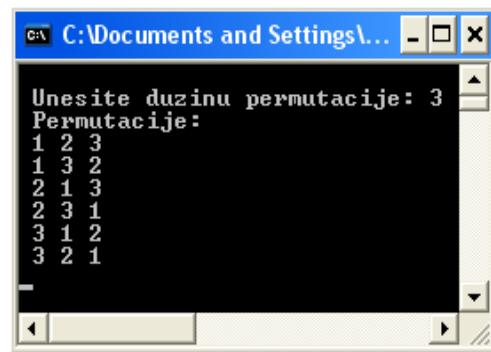
#include <stdio.h>
#define MAX 100

/*Funkcija koja ispisuje elemente niza*/
void IspisiNiz(int a[], int n)
{
    int i;
    for(i=1; i<=n; i++)
        printf(" %d", a[i]);
    printf("\n");
}

/*Funkcija koja proverava da li se x vec nalazi
u permutaciji na prethodnih 1...n mesta*/
int Koriscen(int a[], int n, int x)
{
    int i;
    for(i=1; i<=n; i++)
        if(a[i]==x) return 1;
    return 0;
}

/*Funkcija koja ispisuje sve permutacije od skupa {1,2,...,n}.
a[] - je niz u koji smesta permutacije,
m - oznacava da se na m-tu poziciju u permutaciji
smesta jedan od preostalih celih brojeva,
n - je velicina skupa koji se permutuje,
Funkciju pozivamo sa argumentom m=1 jer krecemo da
formiramo permutaciju od 1. pozicije.*/
void Permutacija(int a[], int m, int n)
{
    int i;
    /*Ako je pozicija na koju treba smestiti broj premasila
velicinu skupa, onda se svi brojevi vec nalaze u
permutaciji i ispisujemo permutaciju.*/
    if(m>n) IspisiNiz(a,n);
    for(i=1; i<=n; i++)
    {
        /*Ako se broj i nije do sada pojavio u permutaciji
od 1 do m-1 pozicije, onda ga stavljamo na poziciju m
i pozivamo funkciju da napravi permutaciju za jedan
vece duzine, tj. m+1. Inace nastavljamo dalje, trazeci
broj kojis e nije pojavio do sada u permutaciji.*/
        if(! Koriscen(a,m-1,i))
        {
            a[m]=i;
            Permutacija(a,m+1,n);
        }
    }
}

```



Испис на екрану

```

main()
{
    int n, a[MAX];
    printf("\n Unesite duzinu permutacije: ");
    scanf("%d", &n);
    if(n < 0 || n >= MAX)
    {
        printf("Duzina permutacije mora biti broj od 0 do %d!\n", MAX);
        return 1;
    }
    printf(" Permutacije: \n");
    Permutacija(a,1,n);
    getch();
    return 0;
}

```

10.4 Претраживање низова

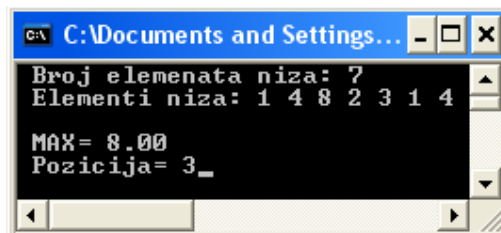
10.27. Саставити програм који за унети низ реалних бројева, дужине **n**, исписује елемент највеће вредности, као и његове позиције у низу.

```

#include <stdio.h>
#define MAX 100

main()
{
    double niz[MAX], max;
    int n, i, imax=0;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf (" Elementi niza: ");
    for (i=0; i<n; i++)
        scanf ("%lf", &niz[i]);
    max=niz[0];
    for(i=1; i<n; i++)
        if(niz[i] > max)
        {
            max=niz[i];
            imax=i;
        }
    printf ("\n MAX= %.2f", max);
    printf ("\n Pozicija= %d", imax+1);
    getch();
    return 0;
}

```

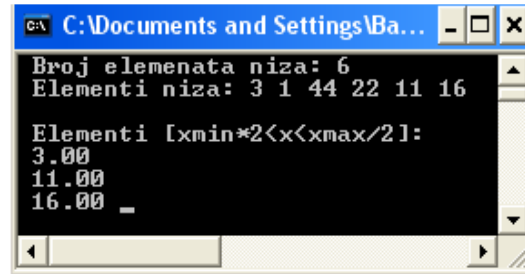


Испис на екрану

10.28. Саставити програм који учита низ реалних бројева, дужине **n**, налази најмањи и највећи члан низа, **xmin** и **xmax**, и исписује све елементе низа који су мањи од **xmax/2** и већи од **xmin*2**.

```
#include <stdio.h>
#define MAX 100

main()
{
    int n, i;
    float x[MAX], xmin, xmax;
    printf(" Broj elemenata niza: ");
    scanf("%d",&n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf("%f",&x[i]);
    xmin=x[0];
    xmax=x[0];
    for(i=1; i<n; i++)
    {
        if(x[i]<xmin) xmin=x[i];
        if(x[i]>xmax) xmax=x[i];
    }
    printf("\n Elementi [xmin*2<x<xmax/2]: ");
    for(i=0; i<n; i++)
        if(x[i]<xmax/2 && x[i]>xmin*2) printf("\n %.2f ",x[i]);
    getch();
    return 0;
}
```

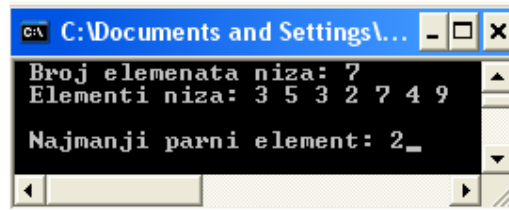


Испис на екрану

10.29. Саставити програм који за унети низ целих бројева, дужине **n**, исписује елемент најмање вредности међу парним бројевима.

```
#include <stdio.h>
#define MAX 100

main()
{
    int niz[MAX], i, n, min;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf ("%d",&niz[i]);
    for(i=0; i<n; i++)
        if(niz[i]%2==0)
        {
            min=niz[i];
            break;
        }
    if(i==n) printf("\n Nema parnih brojeva!");
    else
    {
        for(i=0; i<n; i++)
            if(niz[i]%2==0 && niz[i]<min) min=niz[i];
        printf("\n Najmanji parni element: %d", min);
    }
    getch();
    return 0;
}
```



Испис на екрану

10.30. Саставити програм који за унети низ целих бројева, дужине **n**, проналази и на екрану испишује елементе на парним позицијама и међу њима проналази онај који има максималну вредност. Минимална дужина низа је 2.

```
#include <stdio.h>
#define MAX 100

main()
{
    int niz[MAX], i, n, max;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf ("%d",&niz[i]);
    max=niz[1];
    printf("\n Parne pozicije: ");
    for(i=0; i<n; i++)
        if(i%2!=0 )
        {
            printf("%d ",niz[i]);
            if(niz[i]>max) max=niz[i];
        }
    printf("\n MAX= %d", max);
    getch();
    return 0;
}
```

Испис на екрану

10.31. Саставити програм који испишује обавештење да ли уčitани низ бројева одговара Фибоначијевом низу. Низ бројева који се уноси мора имати најмање три елемента.

Фибоначијев низ: $f_1=1, f_2=1, f_i=f_{i-1}+f_{i-2}, i=3, 4, 5, \dots$

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, pom=2, niz[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    while(n>=3 && n<=MAX)
    {
        for(i=1; i<=n; i++)
        {
            scanf("%d", &j);
            niz[i-1]=j;
        }
        for(i=2; i<n; i++)
        {
            if(niz[i]==(niz[i-1]+niz[i-2]))
                pom++;
            else
            {
                printf("\n Niz nije Fibonacijev!");
                break;
            }
        }
    }
}
```

Испис на екрану

```

    }
    if(pom==n)
        printf("\n Niz jeste Fiboacijev!");
    break;
}
getche();
return 0;
}

```

10.32. Саставити програм који за унети низ целих бројева, дужине **n**, проналази позицију траженог елемента или исписује обавештење да тражени елемент не постоји у низу. Користити методу *Линеарног претраживања*:

- без употреба функција;
- коршћењем функције која линеарно претражује низ.

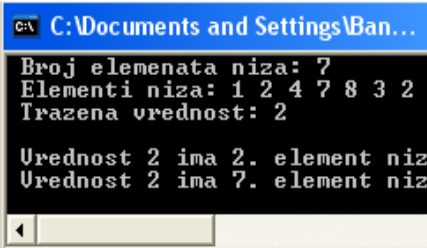
a)

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, n, broj, nadjen=0, niz[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf("%d", &niz[i]);
    printf(" Trazena vrednost: ");
    scanf("%d", &broj);
    for(i=1; i<n; i++)
        if(niz[i] == broj)
        {
            nadjen=1;
            printf ("\n Vrednost %d ima %d. element niza.", broj, i+1);
        }
    if(!nadjen)
        printf ("\n Vrednost %d nije nadjena u nizu.", broj);
    getche();
    return 0;
}

```



Испис на екрану

б)

```

#include <stdio.h>
#define MAX 100

/*Funkcija proverava da li se dati broj nalazi u datom nizu celih brojeva.
 Funkcija vraca poziciju u nizu na kojoj broj pronadjen
 odnosno -1 ukoliko trazenog broja nema.*/
int Trazi(int niz[], int n, int broj)
{
    int i;
    for(i=0; i<n; i++)
        if(niz[i]==broj) return i;
    return -1;
}

main()
{
    int i, n, broj, niz[MAX];
    printf(" Broj elemenata niza: ");

```



```

scanf("%d", &n);
printf(" Elementi niza: ");
for(i=0; i<n; i++)
    scanf("%d", &niz[i]);
printf(" Trazena vrednost: ");
scanf("%d", &broj);
i=Trazi(niz, n, broj);
if(i==-1)
    printf("\n Vrednost %d nije nadjena u nizu.", broj);
else
    printf("\n Vrednost %d ima %d. element niza.", broj, i+1);
getche();
return 0;
}

```

10.33. Саставити програм који за унети низ целих бројева, дужине **n**, проналази позицију траженог елемента или исписује обавештење да тражени елемент не постоји у низу. Претпоставља се да је низ уређен у растућем поретку. Користити методу Бинарног претраживања:

- а) без употреба функција;
- б) коришћењем функције која бинарно претражује низ;
- в) коришћењем рекурзивне функције која бинарно претражује низ.

Метода Бинарне претраге:

Нека је дат низ **a[0], a[1], ..., a[n-1]** и вредност елемента који се тражи **b**. Најпре се **b** са средњим елементом низа (или елементом око средине). Ако су једнаки, претраживање је завршено. Ако је **b** мање од средњег елемента, тада се претраживање наставља у левој половини низа, а супротно у десну. У изабраној половини се примењује исти алгоритам.

а)

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, iSrednji, iMin=0, iMax;
    int n, broj, nadjen=0, niz[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf("%d", &niz[i]);
    printf(" Trazena vrednost: ");
    scanf("%d", &broj);
    iMax=n-1;
    while(iMin<=iMax)
    {
        iSrednji = (iMin+iMax)/2;
        if(broj==niz[iSrednji])
        {
            nadjen=1;
            printf("\n Vrednost %d je %d. element.", broj, iSrednji+1);
            break;
        }
        else if(broj<niz[iSrednji])
            iMax=iSrednji-1;
        else
            iMin=iSrednji+1;
    }
}

```

```

    }
    if(!nadjen)
        printf ("\n Vrednost %d nije nadjena u nizu.", broj);
    getch();
    return 0;
}

```

6)

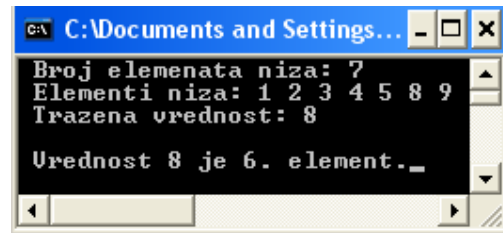
```

#include <stdio.h>
#define MAX 100

/*Funkcija proverava da li se trazeni broj nalazi
unutar niza celih brojeva a.
Funkcija vraca poziciju na kojoj je element nadjen
odnosno -1 ako ga nema.*/
int Trazi(int niz[], int n, int broj)
{
    /*Pretrazujemo interval [iMin, iMax]*/
    int iMin=0, iMax=n-1;
    /*Sve dok interval [iMin, iMax] nije prazan*/
    while(iMin<=iMax)
    {
        /*Srednja pozicija intervala [iMin, iMax]*/
        int iSrednji=(iMin+iMax)/2;
        /* Ispitujemo odnos trazenog broja i srednjeg elementa*/
        if(broj==niz[iSrednji])
            return iSrednji; /*Element je pronadjen*/
        else if(broj<niz[iSrednji])
            iMax=iSrednji-1; /*Pretrazujemo interval [iMin,iSrednji-1]*/
        else
            iMin=iSrednji+1; /*Pretrazujemo interval [iSrednji+1,iMax]*/
    }
    return -1; /*Element je nadjen*/
}

main()
{
    int i, n, broj, niz[MAX];
    printf (" Broj elemenata niza: ");
    scanf ("%d", &n);
    printf (" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf ("%d", &niz[i]);
    printf (" Trazena vrednost: ");
    scanf ("%d", &broj);
    i=Trazi(niz, n, broj);
    if(i== -1)
        printf ("\n Vrednost %d nije nadjena u nizu.", broj);
    else
        printf ("\n Vrednost %d ima %d. element niza.", broj, i+1);
    getch();
    return 0;
}

```



Испис на екрану

B)

```

#include <stdio.h>
#define MAX 100

/*Funkcija proverava da li se trazeni broj nalazi unutar niza celih brojeva a.
Funkcija vraca poziciju na kojoj je element nadjen odnosno -1 ako ga nema.*/
int Trazi(int niz[], int iMin, int iMax, int broj)
{
    /*Ukoliko je interval prazan, elementa nema*/
    if(iMin>iMax) return -1;
    /*Srednja pozicija intervala [iMin, iMax]*/
    int iSrednji=(iMin+iMax)/2;
    /*Ispitujemo odnos trazenog broja i srednjeg elementa*/
    if(broj==niz[iSrednji])
        return iSrednji; /*Element je pronadjen*/
    else if(broj<niz[iSrednji])
        /*Pretrazujemo interval [iMin, iSrednji-1]*/
        return Trazi(niz, iMin, iSrednji-1, broj);
    else
        /*Pretrazujemo interval [iSrednji+1, iMax] */
        return Trazi(niz, iSrednji+1, iMax, broj);
    return -1; /*Element je nadjen*/
}

main()
{
    int i, n, broj, niz[MAX];
    printf (" Broj elemenata niza: ");
    scanf ("%d", &n);
    printf (" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf ("%d", &niz[i]);
    printf (" Trazena vrednost: ");
    scanf ("%d", &broj);
    i=Trazi(niz, 0, n-1, broj);
    if(i!=-1)
        printf("\n Vrednost %d nije nadjena u nizu.", broj);
    else
        printf("\n Vrednost %d ima %d. element niza.", broj, i+1);
    getch();
    return 0;
}

```

10.5 Уређивање и сортирање низова

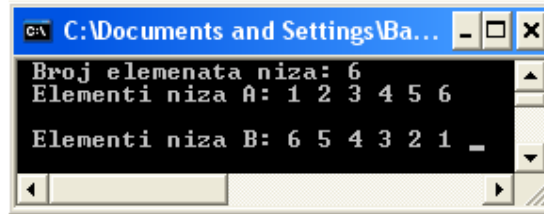
Табела 10.1: Облици сортирања низова

поредак	изглед низа
опадајући	$a[0] > a[1] > \dots > a[i] > a[i+1] > \dots > a[n-1]$
растући	$a[0] < a[1] < \dots < a[i] < a[i+1] < \dots < a[n-1]$
неопадајући	$a[0] \leq a[1] \leq \dots \leq a[i] \leq a[i+1] \leq \dots \leq a[n-1]$
нерастући	$a[0] \geq a[1] \geq \dots \geq a[i] \geq a[i+1] \geq \dots \geq a[n-1]$

10.34. Саставити програм који од унетог низа **A** целих бројева дужине **n** формира и исписује низ **B** са обрнутим распоредом елемената.

```
#include <stdio.h>
#define MAX 100

main()
{
    int a[MAX], i, n;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza A: ");
    for(i=0; i<n; i++)
        scanf("%d",&a[i]);
    printf("\n Elementi niza B: ");
    for(i=n-1; i>=0; i--)
        printf("%d ", a[i]);
    getch();
    return 0;
}
```

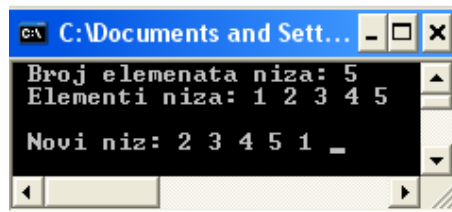


Испис на екрану

10.35. Саставити програм за циклично премештање елемената задатог низа целих бројева дужине **n** за једно место у лево и исписивање новодобијеног низа.

```
#include <stdio.h>
#define MAX 100

main()
{
    int niz[MAX], i, n, pom;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf("%d",&niz[i]);
    pom=niz[0];
    for(i=0; i<n-1; i++)
        niz[i]=niz[i+1];
    niz[n-1]=pom;
    printf("\n Novi niz: ");
    for(i=0; i<n; i++)
        printf("%d ", niz[i]);
    getch();
    return 0;
}
```



Испис на екрану

10.36. Саставити програм који за унети низ целих бројева дужине **n** врши ротирање чланова низа за **x** места у лево и исписује новодобијени низ.

```
#include <stdio.h>
#define MAX 100

void CitajNiz(int a[],int n)
{
```

```

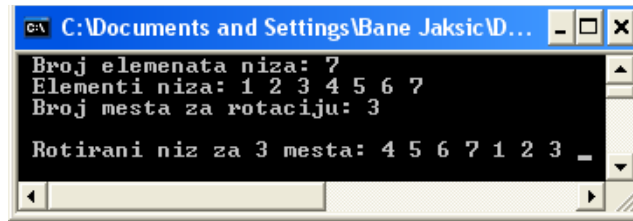
    int i;
    for(i=0; i<n; i++)
        scanf("%d",&a[i]);
}

void PisiNiz(int a[],int n)
{
    int i;
    for(i=0; i<n; i++)
        printf("%d ", a[i]);
}

void Rotiraj(int a[], int n)
{
    int pom, i;
    pom=a[0];
    for(i=1; i<n; i++)
        a[i-1]=a[i];
    a[n-1]=pom;
}

main()
{
    int i, x, n, a[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d",&n);
    printf(" Elementi niza: ");
    CitajNiz(a, n);
    printf(" Broj mesta za rotaciju: ");
    scanf("%d",&x);
    for(i=0; i<x; i++)
        Rotiraj(a, n);
    printf("\n Rotirani niz za %d mesta: ", x);
    PisiNiz(a, n);
    getch();
    return 0;
}

```



Испис на екрану

10.37. Саставити програм који за унети низ целих бројева дужине **n** врши замену суседних елемената низа на парним и непарним позицијама и исписује новодобијени низ.

```

#include <stdio.h>
#define MAX 100

void CitajNiz(int a[],int n)
{
    int i;
    for(i=0; i<n; i++)
        scanf("%d",&a[i]);
}

void PisiNiz(int a[],int n)
{
    int i;
    for(i=0; i<n; i++)
        printf("%d ", a[i]);
}

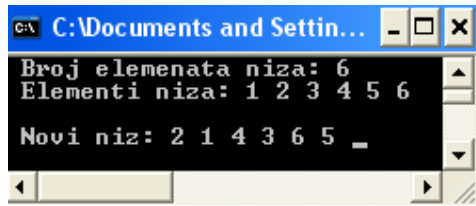
void Zameni(int a[], int n)
{

```

```

int i, pom;
for(i=0; i<n-1; i+=2)
{
    pom=a[i];
    a[i]=a[i+1];
    a[i+1]=pom;
}

```



Испис на екрану

```

main()
{
    int i, x, n, a[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d",&n);
    printf(" Elementi niza: ");
    CitaNiz(a, n);
    Zameni(a, n);
    printf("\n Novi niz: ");
    PisiNiz(a, n);
    getche();
    return 0;
}

```

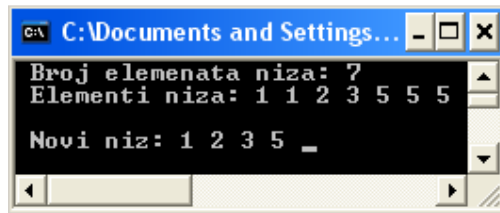
10.38. Саставити програм који за унети низ целих бројева дужине **n** формира и приказује нови низ који је састављен од елемената без понављања унетог низа.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, nadjeniIsti, a[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf ("%d", &a[i]);
    printf("\n Novi niz: ");
    for(i=0; i<n-1; i++)
    {
        nadjeniIsti=0;
        for(j=i+1; j<n; j++)
            if(a[i] == a[j])
            {
                nadjeniIsti=1;
                break;
            }
        if(!nadjeniIsti)
            printf ("%d ",a[i]);
    }
    printf ("%d ",a[n-1]);
    getche();
    return 0;
}

```



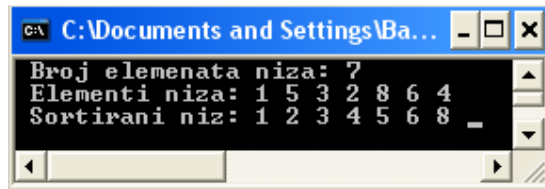
Испис на екрану

10.39. Саставити програм за сортирање унетог низа целих бројева дужине **n** у неоппадајући поредак методом избора (*Selection Sort*). Исписати сортирани низ.

Selection Sort подразумева да минимални елемент низа размени са **a[0]**, минимални елемент одсечка **a[1], a[2], ..., a[n-1]** разменити са **a[1]**, минимални елемент одсечка **a[2], a[3], ..., a[n-1]** разменити са **a[2]**; исти поступак применити на преостале елементе осим последњег који се налази на свом месту.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, pom, niz[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf ("%d", &niz[i]);
    for(i=0; i<n-1; i++)
        for(j=i+1; j<n; j++)
            if(niz[i] > niz[j])
            {
                pom=niz[i];
                niz[i]=niz[j];
                niz[j]=pom;
            }
    printf(" Sortirani niz: ");
    for(i=0; i<n; i++)
        printf ("%d ", niz[i]);
    getch();
    return 0;
}
```



Испис на екрану

10.40. Саставити програм за сортирање унетог низа целих бројева дужине **n** у неоппадајући поредак методом уметања (*Insert Sort*). Исписати сортирани низ.

Insert Sort: Нека је првих **k** елемената већ уређено у неоппадајућем поретку, тада се узима (**k+1**)-ви елемент и уметне на одговарајуће место међу првих **k** елемената тако да првих **k+1** елемената буде уређено. Овај се метод примењује за **k** од **0** до **n-2**.

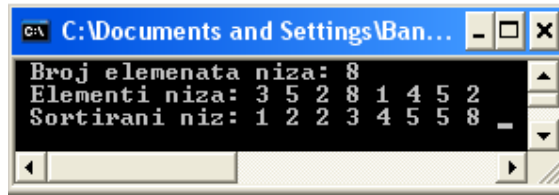
```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, pom, niz[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf("%d", &niz[i]);
    for(i=1; i<n; i++)
    {
        pom=niz[i];
        for(j=i-1; j>=0; j--)
            if (niz[j] > pom)
                niz[j+1]=niz[j];
            else break;
    }
}
```

```

        niz[j+1]=pom;
    }
    printf (" Sortirani niz: ");
    for(i=0; i<n; i++)
        printf ("%d ",niz[i]);
    getch();
    return 0;
}

```



Испис на екрану

10.41. Саставити програм за сортирање унетог низа целих бројева дужине **n** у неоппадајући поредак методом мехурића (*Bubble Sort*). Исписати сортирани низ.

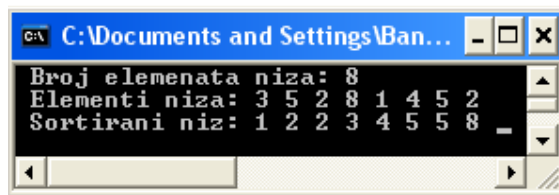
Bubble Sort: Пролазимо кроз низ редом поредећи суседне елементе, и при том их замењујући ако су у погрешном поретку. Овим се највећи елемент попут мехурића истискује на "површину", тј. на крајњу десну позицију. Након тога је потребно овај поступак поновити над низом **a[0],...,a[n-2]**, тј. над првих **n-1** елемената низа без последњег који је постављен на праву позицију. Након тога се исти поступак понавља над све краћим и краћим префиксима низа, чиме се један по један истискују елементи на своје праве позиције.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, pom, niz[MAX];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf("%d", &niz[i]);
    for(i=n-1; i>0; i--)
        for(j=0; j<i; j++)
            if(niz[j] > niz[j+1])
            {
                pom=niz[j];
                niz[j]=niz[j+1];
                niz[j+1]=pom;
            }
    printf(" Sortirani niz: ");
    for(i=0; i<n; i++)
        printf("%d ",niz[i]);
    getch();
    return 0;
}

```



Испис на екрану

10.42. Саставити програм којим се у уређени низ бројева умеће нови број тако да низ и даље буде уређен. Исписати новодобијени низ.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, n, b, niz[MAX+1];
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");

```



```

for(i=0; i<n; i++)
    scanf("%d", &niz[i]);
printf(" Element koji treba umetnuti: ");
scanf("%d", &b);
for(i=n-1; i>=0 && niz[i]>b; i--)
    niz[i+1]=niz[i];
niz[i+1]=b;
n++;
printf("\n Novi niz: ");
for(i=0; i<n; i++)
    printf("%d ", niz[i]);
getche();
return 0;
}

```

Испис на екрану

10.43. Саставити програм који учитава два низа целих бројева који су уређени по неопадајућем редоследу (различитих дужина) и исписује њихову фузију која је такође неопадајућа.

```

#include <stdio.h>
#define MAX 100

main()
{
    int nizA[MAX], nizB[MAX], nizC[2*MAX];
    int na, nb, nc, ia, ib, ic;
    printf(" Broj elemenata niza A: ");
    scanf("%d", &na);
    printf(" Elemneti A: ");
    for(ia=0; ia<na; ia++)
        scanf ("%d", &nizA[ia]);
    printf(" Broj elemenata niza B: ");
    scanf("%d", &nb);
    printf(" Elemneti B: ");
    for(ib=0; ib<nb; ib++)
        scanf("%d", &nizB[ib]);
    nc=na+nb;
    ia=0;
    ib=0;
    if(na==0)
    {
        for(ic=0; ic<nc; ic++)
            nizC[ic]=nizB[ic];
    }
    else if (nb==0)
    {
        for(ic=0; ic<nc; ic++)
            nizC[ic]=nizA[ic];
    }
}

```

```

else
{
    for(ic=0; ic<nc; ic++)
    {
        if(nizA[ia]<=nizB[ib])
        {
            nizC[ic]=nizA[ia];
            ia++;
        }
        else
        {
            nizC[ic]=nizB[ib];
            ib++;
        }
    }
    printf("\n Elementi niza C: ");
    for(ic=0; ic<nc; ic++)
        printf("%d ", nizC[ic]);
    getche();
    return 0;
}

```

Испис на екрану

10.44. Саставити програм који исписује број који се највећи број пута појављује у низу целих бројева, као и број појављивања. Прво сортирати низ у растућем поретку, а затим пронаћи најдужу секвенцу једнаких елемената. Низ се уноси са тастатуре.

```
#include <stdio.h>
#define MAX 100

void Sortiraj(int a[], int n)
{
    int i, j, pom, min;
    for(i=0; i<n-1; i++)
    {
        min=i;
        for(j=i+1; j<n; j++)
            if(a[j]<a[min])
                min=j;
        if(min!=i)
        {
            pom=a[i];
            a[i]=a[min];
            a[min]=pom;
        }
    }
}

main()
{
    int a[MAX], i, j, n, indeks=-1, duzina=-1, brojac;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)
        scanf("%d", &a[i]);
    Sortiraj(a,n);
    printf("\n Sortiran niz: ");
    for(i=0; i<n; i++)
        printf("%d ", a[i]);
    for(i=0; i<n; i++)
    {
        brojac=1;
        for(j=i+1; j<n && a[i]==a[j]; j++)
            brojac++;
        if(brojac > duzina)
        {
            duzina=brojac;
            indeks=i;
        }
    }
    printf("\n Najveci broj puta se pojavljuje broj %d i to %d puta.",
           a[indeks], duzina);
    getche();
    return 0;
}
```

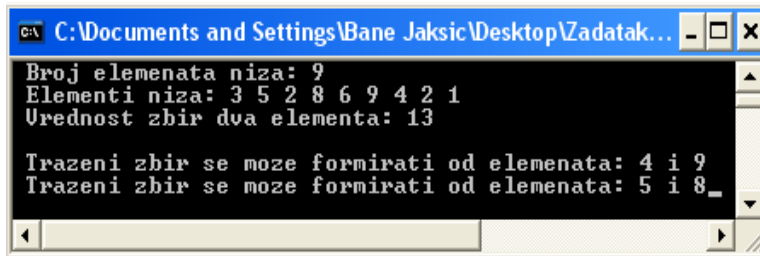
```
C:\Documents and Settings\Bane Jaksic\Desktop\Zadatak7...
Broj elemenata niza: 9
Elementi niza: 2 4 7 2 2 1 9 2 3
Sortiran niz: 1 2 2 2 2 3 4 7 9
Najveci broj puta se pojavljuje broj 2 i to 4 puta.
```

Испис на екрану

10.45. Саставити програм који испитује да ли у унетом низу целих бројева постоје два елемента чији је збир једнак датом броју. Користити бинарну претрагу за проналажење допуне до датог броја. Елементи низа се уносе са тастатуре.

```
#include <stdio.h>
#define MAX 100
```

```
void Sortiraj(int a[], int n)
{
    int i, j, pom, min;
    for(i=0; i<n-1; i++)
    {
        min=i;
        for(j=i+1; j<n; j++)
            if(a[j]<a[min])
                min=j;
        if(min!=i)
        {
            pom=a[i];
            a[i]=a[min];
            a[min]=pom;
        }
    }
}
```



Испис на екрану

```
int BinarnaPretraga(int a[], int n, int x)
{
    int s, l=0, d=n-1;
    while(l<=d)
    {
        s=(l+d)/2;
        if(a[s]==x)    return s;
        if(a[s]>x)    d=s-1;
        else    l=s+1;
    }
    return -1;
}

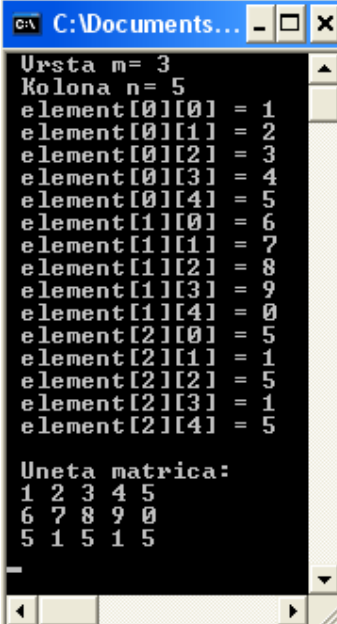
main()
{
    int a[MAX], n, i, zbir, drugi, nadjen =0;
    printf(" Broj elemenata niza: ");
    scanf("%d", &n);
    printf(" Elementi niza: ");
    for(i=0; i<n; i++)    scanf("%d", &a[i]);
    Sortiraj(a, n);
    printf(" Vrednost zbir dva elementa: ");
    scanf("%d", &zbir);
    /*Uslov drugi>i stoji samo da bi se izbegla nepotreban ponavljanja u ispisu.*/
    for(i=0; i<n; i++)
        if((drugi=BinarnaPretraga(a, n, zbir-a[i])) != -1 && drugi>i)
        {
            printf("\n Trazeni zbir se moze formirati od elemenata: %d i %d",
                a[i], a[drugi]);
            nadjen++;
        }
    if(!nadjen)    printf(" Trazeni zbir se ne moze dobiti!\n");
    getch();
    return 0;
}
```

11 МАТРИЦЕ

11.1. Саставити програм који учитава, а затим исписује елементе матрице **m**x**n**. Елементи матрице су цели бројеви.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, m, n, mat[MAX][MAX];
    printf(" Vrsta m= ");
    scanf("%d", &m);
    printf(" Kolona n= ");
    scanf("%d", &n);
    /*Citanje matrice sa tastature*/
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
        {
            printf(" element[%d][%d] = ", i, j);
            scanf("%d", &mat[i][j]);
        }
    /*Prikaz matrice*/
    printf("\n Uneta matrica:\n");
    for(i=0; i<m; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d", mat[i][j]);
        printf("\n");
    }
    getch();
    return 0;
}
```



```
C:\Documents...
Vrsta m= 3
Kolona n= 5
element[0][0] = 1
element[0][1] = 2
element[0][2] = 3
element[0][3] = 4
element[0][4] = 5
element[1][0] = 6
element[1][1] = 7
element[1][2] = 8
element[1][3] = 9
element[1][4] = 0
element[2][0] = 5
element[2][1] = 1
element[2][2] = 5
element[2][3] = 1
element[2][4] = 5

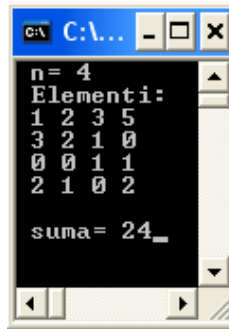
Uneta matrica:
1 2 3 4 5
6 7 8 9 0
5 1 5 1 5
```

Испис на екрану

11.2. Саставити програм који за унуту матрицу димензија $n \times n$ врши сабирање њених елемената и исписује добијени резултат. Елементи су цели бројеви.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, s=0, mat[MAX][MAX];
    printf(" n= ");
    scanf("%d", &n);
    printf(" Elementi:\n");
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            scanf("%d", &mat[i][j]);
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            s+=mat[i][j];
    printf("\n suma= %d", s);
    getch();
    return 0;
}
```

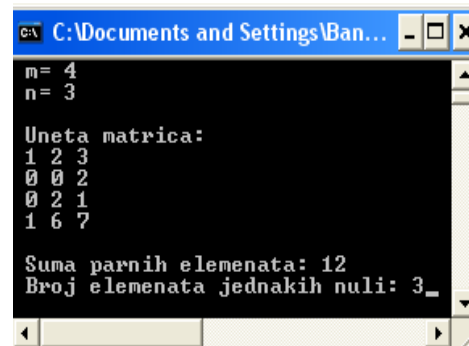


Испис на екрану

11.3. Саставити програм који учита матрицу димензија $m \times n$, а затим врши сабирање елемената који су парни бројеви. На крају исписати суму парних бројева и број елемената који су једнаки нули. Елементи матрице су цели бројеви од 0 до 9.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, m, n, s=0, nule=0, mat[MAX][MAX];
    printf(" m= ");
    scanf("%d",&m);
    printf(" n= ");
    scanf("%d", &n);
    printf("\n Uneta matrica:\n");
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
            scanf("%d",&mat[i][j]);
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
        {
            if(mat[i][j]%2==0) s+=mat[i][j];
            if(mat[i][j]==0) nule++;
        }
    printf("\n Suma parnih elemenata: %d", s);
    printf("\n Broj elemenata jednakih nuli: %d", nule);
    getch();
    return 0;
}
```



Испис на екрану

11.4. Саставити програм који учита две матрице целих бројева, **a** и **b**, обе димензија **m****x****n**, а затим врши сабирање ове две матрице и исписује нову матрицу **c**. Матрице се сабирају тако што се сабирају елементи матрица са истим индексима.

$$c = a + b = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \dots & b_{mn} \end{bmatrix} =$$

$$\begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} & \dots & a_{1n} + b_{1n} \\ a_{21} + b_{21} & a_{22} + b_{22} & \dots & a_{2n} + b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & a_{m2} + b_{m2} & \dots & a_{mn} + b_{mn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \dots & c_{mn} \end{bmatrix}$$

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, m, mat1[MAX][MAX], mat2[MAX][MAX];
    printf(" m= ");
    scanf("%d", &m);
    printf(" n= ");
    scanf("%d", &n);
    printf("\n Elementi prve matrice: \n");
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
            scanf("%d", &mat1[i][j]);
    printf("\n Elementi druge matrice: \n");
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
            scanf("%d", &mat2[i][j]);
    printf("\n Zbir dve matrice:\n");
    for(i=0; i<m; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d", mat1[i][j]+mat2[i][j]);
        printf("\n");
    }
    getche();
    return 0;
}
```

```
C:\Documents and S...
m= 3
n= 4

Elementi prve matrice:
1 1 1 2
0 0 0 1
1 1 1 3

Elementi druge matrice:
2 3 4 5
5 2 1 1
1 1 1 2

Zbir dve matrice:
3 4 5 7
5 2 1 2
2 2 2 5
```

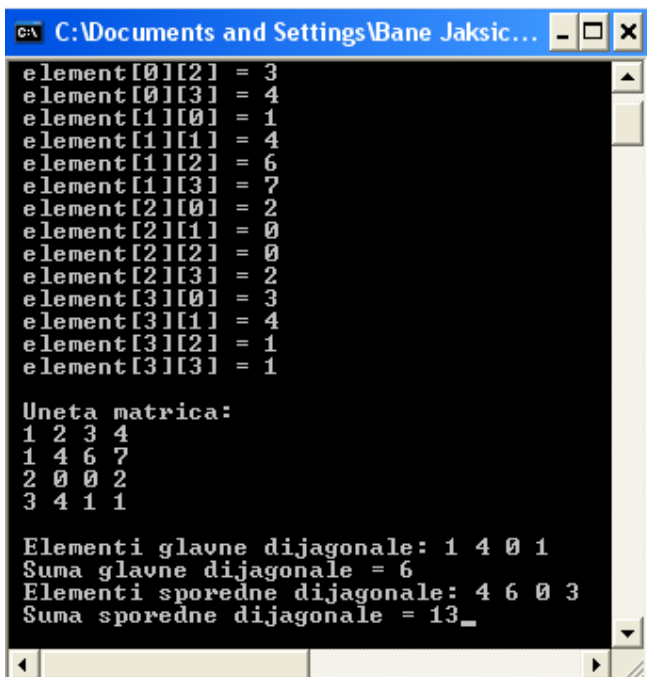
Испис на екрану

11.5. Саставити програм који ће учитати матрицу димензија **n****x****n**, а затим исписати матрицу у облику таблице, исписати све елементе на главној и споредној дијагонали, као и суме елемената на главној и споредној дијагонали. Елементи матрице су цели бројеви.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, sgd=0, ssd=0, mat[MAX][MAX];
    printf(" n= ");
    scanf("%d", &n);
```

```
printf(" Elementi:\n");
for(i=0; i<n; i++)
    for(j=0; j<n; j++)
    {
        printf(" element[%d][%d] = ", i, j);
        scanf("%d", &mat[i][j]);
    }
printf("\n Uneta matrica:\n");
for(i=0; i<n; i++)
{
    for(j=0; j<n; j++)
        printf(" %d",mat[i][j]);
    printf("\n");
}
printf("\n Elementi glavne dijagonale: ");
for(i=0; i<n; i++)
{
    printf("%d ",mat[i][i]);
    sgd+=mat[i][i];
}
printf("\n Suma glavne dijagonale = %d", sgd);
printf("\n Elementi sporedne dijagonale: ");
for(i=0; i<n; i++)
{
    printf("%d ",mat[i][n-i-1]);
    ssd+=mat[i][n-i-1];
}
printf("\n Suma sporedne dijagonale = %d", ssd);
getche();
return 0;
}
```



```
C:\Documents and Settings\Bane Jaksic...
element[0][2] = 3
element[0][3] = 4
element[1][0] = 1
element[1][1] = 4
element[1][2] = 6
element[1][3] = 7
element[2][0] = 2
element[2][1] = 0
element[2][2] = 0
element[2][3] = 2
element[3][0] = 3
element[3][1] = 4
element[3][2] = 1
element[3][3] = 1

Uneta matrica:
1 2 3 4
1 4 6 7
2 0 0 2
3 4 1 1

Elementi glavne dijagonale: 1 4 0 1
Suma glavne dijagonale = 6
Elementi sporedne dijagonale: 4 6 0 3
Suma sporedne dijagonale = 13_
```

Испис на екрану

11.6. Саставити програм који учита матрицу димензија $n \times n$, а затим је исписује у облику таблице и израчунава и исписује суму елемената у свакој врсти. Елементи матрице су цели бројеви.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, s=0, mat[MAX][MAX];
    printf(" n= ");
    scanf("%d", &n);
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
        {
            printf(" element[%d][%d] = ", i, j);
            scanf("%d", &mat[i][j]);
        }
    printf("\n Uneta matrica:\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d", mat[i][j]);
        printf("\n");
    }
    printf("\n Suma elememenata vrste:\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            s+=mat[i][j];
        printf(" %d", s);
    }
    getch();
    return 0;
}
```

```
n= 3
element[0][0] = 1
element[0][1] = 2
element[0][2] = 1
element[1][0] = 7
element[1][1] = 3
element[1][2] = 2
element[2][0] = 0
element[2][1] = 1
element[2][2] = 2

Uneta matrica:
1 2 1
7 3 2
0 1 2

Suma elememenata vrste:
4 16 19_
```

Испис на екрану

11.7. Саставити програм који учита матрицу димензија $m \times n$, а затим је исписује у облику таблице и на основу унетог редног броја врсте врши сабирање елемената у тој врсти. Елементи матрице су цели бројеви.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, m, vrsta, s=0, mat[MAX][MAX];
    printf(" m= ");
    scanf("%d", &m);
    printf(" n= ");
    scanf("%d", &n);
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
        {
            printf(" element[%d][%d] = ", i, j);
            scanf("%d", &mat[i][j]);
        }
    printf("\n Uneta matrica:\n");
    for(i=0; i<m; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d", mat[i][j]);
    }
```



```

    printf("\n");
}
printf("\n Redni broj vrste ciji zbir elemenata zelite: ");
scanf("%d",&vrsta);
for(j=0; j<n; j++)
    s=s+mat[vrsta-1][j];
printf("\n Zbir elememenata %d. vrste: %d", vrsta, s);
getche();
return 0;
}

```

```

C:\Documents and Settings\Bane Jaksic\Desktop\Z...
m= 4
n= 3
element[0][0] = 1
element[0][1] = 2
element[0][2] = 3
element[1][0] = 1
element[1][1] = 5
element[1][2] = 6
element[2][0] = 2
element[2][1] = 0
element[2][2] = 2
element[3][0] = 3
element[3][1] = 3
element[3][2] = 1

Uneta matrica:
1 2 3
1 5 6
2 0 2
3 3 1

Redni broj vrste ciji zbir elemenata zelite: 2
Zbir elememenata 2. vrste: 12_

```

Испис на екрану

11.8. Саставити програм који учита матрицу целих бројева **A** димензија **m x n**, а затим исписује њене елементе у редоследу као што је приказано на следећој слици:

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 11 & 12 & 13 & 14 \\ 21 & 22 & 23 & 24 \\ 31 & 32 & 33 & 34 \end{bmatrix}$$

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, j, m, n, a[MAX][MAX];
    printf(" m= ");
    scanf("%d", &m);
    printf(" n= ");
    scanf("%d", &n);
    printf(" Matrica:\n");
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
            scanf("%d", &a[i][j]);
}

```

```

printf("\n Elementi:\n");
for(j=0; j<n; j++)
{
    if(j%2==0)
        for(i=0; i<m; i++)
            printf(" %d",a[i][j]);
    else
        for(i=m-1; i>=0; i--)
            printf(" %d",a[i][j]);
}
getche();
return 0;
}

```

Испис на екрану

11.9. Саставити програм који учита матрицу реалних бројева **A** димензија **n x n**, а затим формира нову матрицу тако што све чланове врсте (укључујући и дијагонални) дели са дијагоналним чланом. Уколико је дијагонални члан једнак нули, све чланове у том реду поставља на нулу, осим дијагоналног који поставља на 1. Исписати добијену матрицу.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n;
    float a[MAX][MAX], t;
    printf(" n= ");
    scanf("%d", &n);
    printf(" Matrica:\n");
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            scanf("%f",&a[i][j]);
    for(i=0; i<n; i++)
        if(a[i][i]!=0)
        {
            t=a[i][i];
            for(j=0; j<n; j++)
                a[i][j]/=t;
        }
        else
        {
            for(j=0; j<n; j++)
                a[i][j]=0;
            a[i][i]=1;
        }
    printf("\n Nova matrica:\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            printf(" %.2f",a[i][j]);
        printf("\n");
    }
    getche();
    return 0;
}

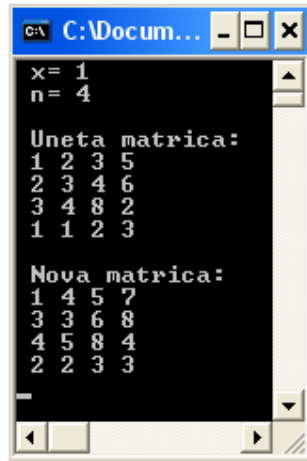
```

Испис на екрану

11.10. Саставити програм који учита један цео број x и матрицу целих бројева A димензија $n \times n$, а затим формира нову матрицу тако што све елементе испод главне дијагонале увећа за вредност x , а елементе изнад главне дијагонале увећава за $2x$. Елементи на главној дијагонали се не мењају. Исписати добијену матрицу.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, x, a[MAX][MAX];
    printf(" x= ");
    scanf("%d",&x);
    printf(" n= ");
    scanf("%d", &n);
    printf("\n Uneta matrica: \n");
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            scanf("%d",&a[i][j]);
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
        {
            if(i>j) a[i][j]+=x;
            if(i<j) a[i][j]+=2*x;
        }
    printf("\n Nova matrica:\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d",a[i][j]);
        printf("\n");
    }
    getch();
    return 0;
}
```



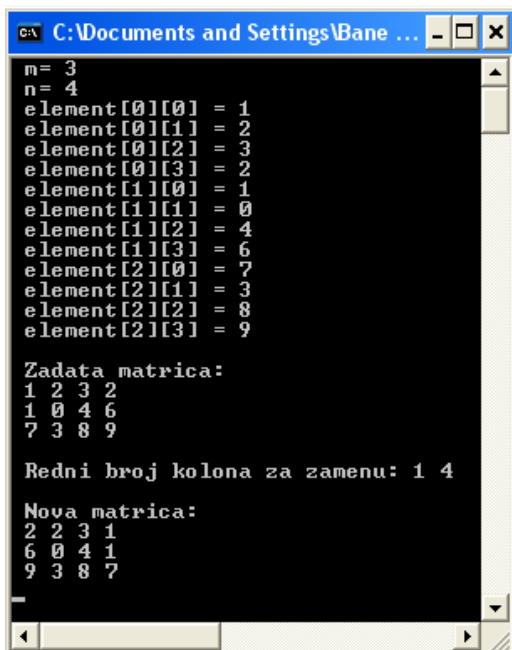
Испис на екрану

11.11. Саставити програм који учита матрицу целих бројева A димензија $m \times n$, а затим исписује матрицу у облику таблице и врши замену места двама колонама на основу унета два цела броја који представљају редне бројеве колона.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, m, k1, k2, pom, a[MAX][MAX];
    printf(" m= ");
    scanf("%d", &m);
    printf(" n= ");
    scanf("%d", &n);
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
        {
            printf(" element[%d][%d] = ", i, j);
            scanf("%d", &a[i][j]);
        }
    printf("\n Zadata matrica:\n");
```

```
for(i=0; i<m; i++)
{
    for(j=0; j<n; j++)
        printf(" %d", a[i][j]);
    printf("\n");
}
do
{
    printf("\n Redni broj kolona za zamenu: ");
    scanf("%d%d", &k1, &k2);
}
while(k1<1 || k1>n || k2<1 || k2>n);
for(i=0; i<n; i++)
{
    pom=a[i][k1-1];
    a[i][k1-1]=a[i][k2-1];
    a[i][k2-1]=pom;
}
printf("\n Nova matrica:\n");
for(i=0; i<m; i++)
{
    for(j=0; j<n; j++)
        printf(" %d", a[i][j]);
    printf("\n");
}
getche();
return 0;
}
```



```
C:\Documents and Settings\Bane ...
m= 3
n= 4
element[0][0] = 1
element[0][1] = 2
element[0][2] = 3
element[0][3] = 2
element[1][0] = 1
element[1][1] = 0
element[1][2] = 4
element[1][3] = 6
element[2][0] = 7
element[2][1] = 3
element[2][2] = 8
element[2][3] = 9

Zadata matrica:
1 2 3 2
1 0 4 6
7 3 8 9

Redni broj kolona za zamenu: 1 4

Nova matrica:
2 2 3 1
6 0 4 1
9 3 8 7
```

Испис на екрану

11.12. Саставити програм који учита матрицу целих бројева **A** димензија **nхn**, а затим исписује матрицу у облику таблице и врши њено транспоновање. Транспонована матрица је матрица која се добија када се врсте почетне матрице поређају по колонама.

а) без употребе функција;

б) употребом функција за учитавања, исписивање и транспоновање матрице.

Исписати транспоновану матрицу.

а)

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, pom, mat[MAX][MAX];
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
        {
            printf(" element[%d][%d] = ", i, j);
            scanf("%d", &mat[i][j]);
        }
    printf("\n Zadana matrica: \n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d",mat[i][j]);
        printf("\n");
    }
    for(i=0; i<n-1; i++)
        for(j=i+1; j<n; j++)
        {
            pom=mat[i][j];
            mat[i][j]=mat[j][i];
            mat[j][i]=pom;
        }
    printf (" \n Transponovana matrica:\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d",mat[i][j]);
        printf("\n");
    }
    getch();
    return 0;
}
```

б)

```
#include <stdio.h>
#define MAX 100

void Citaj(int mat[MAX][MAX], int n)
{
    int i, j;
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
        {
            printf(" element[%d][%d] = ", i, j);
            scanf("%d", &mat[i][j]);
        }
}
```

```

    }
}

void Pisi(int mat[MAX][MAX], int n)
{
    int i, j;
    for (i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d",mat[i][j]);
        printf("\n");
    }
}

void Transp(int mat[MAX][MAX], int n)
{
    int i, j, pom;
    for(i=0; i<n-1; i++)
        for(j=i+1; j<n; j++)
        {
            pom=mat[i][j];
            mat[i][j]=mat[j][i];
            mat[j][i]=pom;
        }
}

main()
{
    int i, j, n, pom, mat[MAX][MAX];
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    Citaj(mat,n);
    printf("\n Zadata matrica: \n");
    Pisi(mat,n);
    Transp(mat,n);
    printf (" \n Transponovana matrica:\n");
    Pisi(mat,n);
    getch();
    return 0;
}

```

```

C:\Documents and ...
n= 3
element[0][0] = 1
element[0][1] = 2
element[0][2] = 3
element[1][0] = 4
element[1][1] = 5
element[1][2] = 6
element[2][0] = 7
element[2][1] = 8
element[2][2] = 9

Zadata matrica:
1 2 3
4 5 6
7 8 9

Transponovana matrica:
1 4 7
2 5 8
3 6 9

```

Испис на екрану

11.13. Саставити програм који учита матрицу целих бројева **A** димензија **nхn**, а затим исписује матрицу у оквиру таблице. Програм треба да одређује највећи и најмањи елемент у свакој врсти и колони, а добијене елементе сместити у једнодимензионалне низове. Исписати формиране низове.

```

#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, a[MAX][MAX];
    int maxv[MAX], minv[MAX], maxk[MAX], mink[MAX];
    printf(" n= ");
    scanf("%d", &n);
    printf("\n");
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)

```

```

    {
        printf(" element[%d][%d] = ", i, j);
        scanf("%d", &a[i][j]);
    }
    printf("\n Zadata matrica: \n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d", a[i][j]);
        printf("\n");
    }
    for(i=0; i<n; i++)
    {
        minv[i]=a[i][0];
        maxv[i]=a[i][0];
        mink[i]=a[0][i];
        maxk[i]=a[0][i];
        for(j=1; j<n; j++)
        {
            if(a[i][j]<minv[i]) minv[i]=a[i][j];
            if(a[i][j]>maxv[i]) maxv[i]=a[i][j];
            if(a[j][i]<mink[i]) mink[i]=a[j][i];
            if(a[j][i]>maxk[i]) maxk[i]=a[j][i];
        }
    }
    printf("\n Najveci u vrstama: ");
    for(i=0; i<n; i++)
        printf("%d ", maxv[i]);
    printf("\n Najmanji u vrstama: ");
    for(i=0; i<n; i++)
        printf("%d ", minv[i]);
    printf("\n Najveci u kolonama: ");
    for(i=0; i<n; i++)
        printf("%d ", maxk[i]);
    printf("\n Najmanji u kolonama: ");
    for(i=0; i<n; i++)
        printf("%d ", mink[i]);
    getch();
    return 0;
}

```

```

C:\Documents and Settings...
n= 3
element[0][0] = 1
element[0][1] = 2
element[0][2] = 5
element[1][0] = 7
element[1][1] = 3
element[1][2] = 0
element[2][0] = 2
element[2][1] = 8
element[2][2] = 1

Zadata matrica:
1 2 5
7 3 0
2 8 1

Najveci u vrstama: 5 7 8
Najmanji u vrstama: 1 0 1
Najveci u kolonama: 7 8 5
Najmanji u kolonama: 1 2 0

```

Испис на екрану

11.14. Саставити програм који учита две матрице целих бројева, **A** димензија **n x m**, и матрицу **B** димензија **m x k**, а затим формира и исписује матрицу **C** добијену множењем матрица **A** и **B**.

$$c = a \cdot b = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1k} \\ b_{21} & b_{22} & \dots & b_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \dots & b_{mk} \end{bmatrix} =$$

$$\begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} + \dots + a_{1m}b_{m1} & \dots & a_{11}b_{1k} + a_{12}b_{2k} + \dots + a_{1m}b_{mk} \\ a_{21}b_{11} + a_{22}b_{21} + \dots + a_{2m}b_{m1} & \dots & a_{21}b_{1k} + a_{22}b_{2k} + \dots + a_{2m}b_{mk} \\ \vdots & \ddots & \vdots \\ a_{n1}b_{11} + a_{n2}b_{21} + \dots + a_{nm}b_{m1} & \dots & a_{n1}b_{1k} + a_{n2}b_{2k} + \dots + a_{nm}b_{mk} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \dots & c_{mn} \end{bmatrix}$$

Две матрице могу да се множе ако је број врсти друге матрице једнак броју колона прве матрице.

```
#include <stdio.h>
#define MAX 100

void Citaj(int mat[MAX][MAX], int n, int m)
{
    int i, j;
    for(i=0; i<n; i++)
        for(j=0; j<m; j++)
        {
            printf(" element[%d][%d] = ", i, j);
            scanf("%d", &mat[i][j]);
        }
}

void Pisi(int mat[MAX][MAX], int n, int m)
{
    int i, j;
    for (i=0; i<n; i++)
    {
        for(j=0; j<m; j++)
            printf(" %3d",mat[i][j]);
        printf("\n");
    }
}

main()
{
    int a[MAX][MAX], b[MAX][MAX];
    int c[MAX][MAX], d[MAX][MAX];
    int i, j, n, m, k, t;
    printf(" Broj vrsta matrice A: ");
    scanf("%d",&n);
    printf(" Broj kolona matrice A: ");
    scanf("%d",&m);
    printf(" Broj kolona matrice B: ");
    scanf("%d",&k);
    printf("\n Matrica A:\n");
    Citaj(a,n,m);
    printf("\n Matrica B:\n");
    Citaj(b,m,k);
    for(i=0; i<n; i++)
        for(j=0; j<k; j++)
        {
            c[i][j]=0;
            for(t=0; t<m; t++)
                c[i][j]=c[i][j]+a[i][t]*b[t][j];
        }
    printf("\n Matrica A:\n");
    Pisi(a,n,m);
    printf("\n Matrica B:\n");
    Pisi(b,m,k);
    printf("\n Matrica C:\n");
    Pisi(c,n,k);
    getch();
    return 0;
}
```

```
C:\Documents and Settings...
Broj vrsta matrice A: 5
Broj kolona matrice A: 3
Broj kolona matrice B: 2

Matrica A:
element[0][0] = 1
element[0][1] = 0
element[0][2] = 3
element[1][0] = 0
element[1][1] = 2
element[1][2] = 2
element[2][0] = 3
element[2][1] = 3
element[2][2] = 0
element[3][0] = 2
element[3][1] = 1
element[3][2] = 1
element[4][0] = 1
element[4][1] = 2
element[4][2] = 1

Matrica B:
element[0][0] = 1
element[0][1] = 2
element[1][0] = 2
element[1][1] = 1
element[2][0] = 3
element[2][1] = 2

Matrica A:
1  0  3
0  2  2
3  3  0
2  1  1
1  2  1

Matrica B:
1  2
2  1
3  2

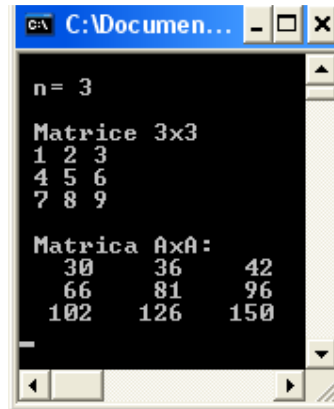
Matrica C:
10  8
10  6
9  9
7  7
8  6
```

Испис на екрану

11.15. Саставити програм који учита матрицу целих бројева **A** димензија **nхn**, а затим исписује матрицу која представља производ **AхA**.

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, n, k;
    int a[40][40], b[40][40];
    printf("\n n= ");
    scanf("%d", &n);
    printf("\n Matrice %dx%d\n", n, n);
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            scanf("%d", &a[i][j]);
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
        {
            b[i][j]=0;
            for(k=0; k<n; k++)
                b[i][j]+=a[i][k]*a[k][j];
        }
    printf("\n Matrica AxA:\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
            printf(" %4d ", b[i][j]);
        printf("\n");
    }
    getch();
    return 0;
}
```



Испис на екрану

11.16. Саставити програм који учита матрицу целих бројева **A** димензија **mхm** и цео број **n**, а затим исписује матрицу која представља **n**-ти производ матрице **A** (**n**-ти степен матрице **A**).

```
#include <stdio.h>
#define MAX 100

main()
{
    int i, j, m, n, k, p;
    int a[MAX][MAX], b[MAX][MAX], c[MAX][MAX];
    printf("\n Stepen n= ");
    scanf("%d", &n);
    printf(" Dimanzija m= ");
    scanf("%d", &m);
    printf("\n Matrica %dx%d:\n", m, m);
    for(i=0; i<m; i++)
        for(j=0; j<m; j++)
        {
            scanf("%d", &a[i][j]);
            b[i][j]=a[i][j];
        }
}
```

```

/*Stepenovanje matrice mnozeci je sa samom sobom n-1 puta*/
for(p=1; p<n; p++)
{
    for(i=0; i<m; i++)
        for(j=0; j<m; j++)
        {
            c[i][j]=0;
            for(k=0; k<m; k++)
                c[i][j]+=a[i][k]*b[k][j];
        }
    /*Prenos matrice C u matricu B*/
    for(i=0; i<m; i++)
        for(j=0; j<m; j++)
            b[i][j]=c[i][j];
}

/*Ispis matrice*/
printf("\n Rezultujuca matrica:\n");
for(i=0; i<m; i++)
{
    for(j=0; j<m; j++)
        printf("%6d ",c[i][j]);
    printf("\n");
}
getche();
return 0;
}

```

```

C:\Documents a...
Stepen n= 4
Dimanzija m= 3

Matrica 3x3:
1 2 3
4 5 0
1 2 1

Rezultujuca matrica:
636      894      312
1200     1689     588
496      698      244

```

Испис на екрану

11.17. Саставити програм који учита матрицу реалних бројева **A** димензија **nхn** и, а затим дату матрицу своди на горњу десну троугласту. Исписати новодобијену матрицу.

```

#include <stdio.h>
#define MAX 100

main()
{
    float a[MAX][MAX], z, xm;
    int n, i, j, k, s=1;
    printf("\n n= ");
    scanf("%d",&n);
    printf("\n Matrica %dx%d:\n",n,n);
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            scanf("%f",&a[i][j]);
    /*Iteracija za svaku kolonu: svodjenje na nulu clanova ispod dijagonale*/
    for(k=0; k<n-1; k++)
    {
        if(a[k][k]==0)
            /*Potrebno je izvrstiti zamenu vrsti; nadji clan u k-toj koloni razlicit od nule; ako ga nema, ispisi poruku*/
            {
                for(i=k+1; i<n-1 && a[i][k]==0; i++)
                    if(i==n-1 && a[i][k]==0)
                    {
                        printf("\n Matrica se ne moze svesti na gornju trouglastu.");
                        return 1;
                    }
            }
        /*Zameniti i-tu i k-tu vrstu*/
    }
}

```

```

        for(j=k; j<n; j++)
        {
            z=a[k][j];
            a[k][j]=a[j][k];
            a[j][k]=z;
        }
        s=-s;
    }
    /*Svodjenje na nulu clanova ispod dijagonale */
    for(i=k+1; i<n; i++)
    {
        xm=a[i][k]/a[k][k];
        for(j=k; j<n; j++)
            a[i][j]-=xm*a[k][j];
    }
}
if(a[n][n]==0)
{
    printf("\n Matrica se ne moze svesti na gornju trouglastu.");
    return 1;
}
a[n][n]=a[n][n]*s; /*Postavi ispravni predznak*/

/*Ispis rezultata*/
printf("\n Gornja trouglasta matrica:\n");
for(i=0; i<n; i++)
{
    for(j=0; j<n; j++)
        printf("%7.2f",a[i][j]);
    printf("\n");
}
getche();
return 0;
}

```

```

C:\Documents and Settings...
n = 4
Matrica 4x4:
1 2 3 4
5 6 7 8
9 0 1 2
3 4 5 6

Gornja trouglasta matrica:
1.00 2.00 3.00 4.00
0.00 -4.00 -8.00 -12.00
0.00 0.00 10.00 20.00
0.00 0.00 0.00 0.00

```

Испис на екрану

Решење се своди на итеративни поступак у петљи у којој се повећава варијабла **k**. Свођење чланова на нулу обавља се за све врсте испод дијагонале и то у свакој колони од прве до предпоследње (**k** иде од **0** до **n-2**) јер у последњој колони више нема чланова испод дијагоналног. Ако је дијагонални члан једнак нули па није могуће с њим делити, проналази се врста испод дијагонале која у тој (**k**-тој) колони има вредност различиту од нуле па се ти елементи замене и варијабли **s** се промени предзнак. Ако таква врста не постоји и замена није могућа, матрица се не може свести на горњу троугласту па се исписује одговарајућа порука и завршава даље рачунање. За дијагонални члан различит од нуле обавља се множење и умањење да би се добиле нуле у свим врстама испод дијагонале. При томе се не узимају колоне лево од дијагонале (**j** иде од **k** до **n-1**) јер су ти чланови већ сведени на нулу у претходним итерацијама. Након обављене итерације по **k** (до предпоследње колоне), може се догодити да је последњи дијагонални члан једнак нули. Тада се матрица опет не може свести на горњу троугласту па се исписује одговарајућа порука.

11.18. Саставити програм који учита матрицу целих бројева димензија $m \times n$, а затим врши уређивање колона матрице по неоппадајућем редоследу збинова елемената по колонама. Исписати новодобијену матрицу.

```
#include <stdio.h>
#define MAX 100

main()
{
    int mat[MAX][MAX], s[MAX], p;
    int m, n, i, j, min;
    printf(" m= ");
    scanf("%d", &m);
    printf(" n= ");
    scanf("%d", &n);
    for(i=0; i<m; i++)
        for(j=0; j<n; j++)
        {
            printf(" element[%d][%d] = ", i, j);
            scanf("%d", &mat[i][j]);
        }

    /*Ispisivanje zadate matrice*/
    printf("\n Zadata matrica: \n");
    for(i=0; i<m; i++)
    {
        for(j=0; j<n; j++)
            printf(" %d", mat[i][j]);
        printf("\n");
    }

    /*Racunanje zbirova elemenata po kolonama*/
    for(j=0; j<n; j++)
        for(s[j]=i=0; i<m; i++)
            s[j]+=mat[i][j];

    /*Uredjivanje kolona po velicinama zbirova*/
    for(i=0; i<n-1; i++)
    {
        /*Trazenje najmanjeg zbira*/
        for(min=i, j=i+1; j<n; j++)
            if(s[j]<s[min]) min=j;
        if(min != i)
        {
            p=s[i];
            s[i]=s[min];
            s[min]=p;
            /*Zamena kolona*/
            for(j=0; j<n; j++)
            {
                p=mat[j][i];
                mat[j][i]=mat[j][min];
                mat[j][min]=p;
            }
        }
    }

    /*Ispisivanje uredjene matrice*/
    printf("\n Uredjena matrica:\n");
    for(i=0; i<m; i++)
```

```
m= 3
n= 4
element[0][0] = 1
element[0][1] = 2
element[0][2] = 3
element[0][3] = 5
element[1][0] = 1
element[1][1] = 8
element[1][2] = 2
element[1][3] = 0
element[2][0] = 2
element[2][1] = 3
element[2][2] = 1
element[2][3] = 6

Zadata matrica:
1 2 3 5
1 8 2 0
2 3 1 6

Uredjena matrica:
1 3 5 2
1 2 0 8
2 1 6 3

Suma po kolonama:
4 6 11 13_
```

Испис на екрану

```
{
    for(j=0; j<n; j++)
        printf(" %d", mat[i][j]);
    printf("\n");
}
printf("\n Suma po kolonama:\n");
for(j=0; j<n; j++)
    printf("% d", s[j]);
getche();
return 0;
}
```

ЛИТЕРАТУРА

- [1] Brian W. Kernighan, Dennis M. Ritchie: **The C Programming Language**, New Jersey, 1988.
- [2] Laslo Kraus: **Programski jezik C sa rešenim zadacima**, Akademska misao, Beograd, 2006.
- [3] Laslo Kraus: **Rešeni zadaci iz programskog jezika C**, Akademska misao, Beograd, 2005.
- [4] Ivo Mateljan: **Programiranje C jezikom**, Split, 2005/2006.
- [5] Jozo J. Dujmović: **Programski jezici i metode programiranja**, Naučna knjiga, Beograd, 1990.
- [6] B. S. Gottfried: **Theory and Problems of Programming with C**, Schaum's outline series, McGraw-Hill, 1996.
- [7] Clovis Tondo, Scott Gimpel: **Programski jezik C – rešenja zadataka**, CET, Beograd, 2004.
- [8] A.Hansen: **Programiranje na jeziku C – potpuni vodič za programski jezik C**, Mikroknjiga, Beograd, 2000.
- [9] Igor Đurović, Slobodan Đukanović, Vesna Popović: **Programski jezik C sa zbirkom riješenih zadataka**, ETF Podgorica, Podgorica, 2006.
- [10] Milan Čabarkapa: **C osnovi programiranja**, Krug, Beograd, 1996.
- [11] Milan Čabarapa, Stanka Matković: **C/C++ zbirka zadataka**, Krug, Beograd, 2003.
- [12] Milan Čabarkapa, Nevenka Spalević: **Metodička zbirka zadataka iz programiranja**, Sova, Novi Beograd, 1997.
- [13] Абрамов С.А., Гнездилова Г.Г., Капустина Е.Н., Селъун М.И.: **Задачи по программированию**, Наука, Москва, 1988.
- [14] Абрамов В.Г., Тирфионов Н.П., Трифонава Г.Н., **Введение в язык Паскаль**, Наука, Москва, 1988.