DIACONU DIANA-ELENA

ISTRATE NICOLAE-CECILIAN (in memoriam)

CHIRIAC BEATRICE-MIHAELA

PROGRAMAREA CALCULATOARELOR ÎN LIMBAJUL C++

Auxiliar curricular pentru disciplina Informatică

Clasa a IX-a

Decembrie 2023
Târgoviște

CUPRINS

Capitolul 1.

Elemente de bază ale Limbajului de programare C/C++

Noțiuni teoretice

Dintotdeauna oamenii şi-au pus diverse întrebări. De exemplu s-au întrebat *Cum a apărut universul?* sau *Dacă Ana bea două pahare de apă pe zi timp de şapte zile, câte pahare de apă a băut într-o săptămână?* La unele întrebări nu avem încă răspuns sau răspunsul este amplu şi bazat pe diverse teorii. La alte întrebări, precum cea în care trebuie să aflăm câte pahare de apă bea Ana, trebuie să ne gândim doar că având ca date cunoscute numărul de pahare şi numărul de zile şi prelucrând cu ajutorul unei înmulţiri datele cunoscute, aflăm într-un timp scurt rezultatul.

Algoritmul este o succesiune de operaţii de prelucrare a datelor de intrare pentru a obţine un rezultat.

Proprietăţile unui algoritm sunt: *generalitatea*, *claritatea*, *finitudinea* şi *eficienţa*. Generalitatea este proprietatea algoritmilor de a rezolva toate problemele de acelaşi tip, nu doar un caz particular. Claritatea este proprietatea algoritmilor de a rezolva clar problemele. Finitudinea este proprietatea algoritmilor de a se executa într-un timp finit. Eficienţa este proprietatea algoritmilor de a se executa într-un număr cât mai mic de paşi.

Pentru început, vom utiliza un limbaj natural numit ***pseudocod***, pe care îl vom scrie în limba română, urmat de limbajul de programare C++ pe care o să îl scriem în Code::Blocks. Cu ajutorul aplicaţiei Code::Blocks putem verifica şi rula programul scris în C++.

În interiorul codului programului, putem introduce comentarii pe o singură linie cu // şi pe mai multe linii cu /* */. Aceste comentarii sunt ignorate de program, dar sunt utilizate pentru a introduce câte un comentariu acolo unde considerăm necesar.

De exemplu putem scrie:

a=b; //atribuire
//alt comentariu
	/*enunţul
	unei probleme*/

În limbajul de programare C++ putem utiliza literele mari şi mici ale alfabetului englez, cifrele şi caracterele speciale: **A, B, ..., Z, a, b, ... , z, 0, 1,... , 9, +, -, *, /, _, <, >, =, ”, ;, :, (,), [,], {, }, #, $, @, ^, spaţiu (blank)**.

Anumite cuvinte, numite **cuvinte cheie**, au o anumită semnificaţie pentru limbajul de programare şi nu le putem utiliza în alt scop decât cel pentru care au

fost create. Printre acestea sunt: **cin**, **cout**, **int**, **float**, **main**, **include**, **iostream**, **if**, **while**, **for**.

Mai jos avem un exemplu de program scris în C++, care poate fi compilat de către Code::Blocks. Programul rulează, dar nu face nimic.

```
#include <iostream>
using namespace std;
int main()
{   return 0; }
```

Analizând codul de mai sus linie cu linie, reţinem următoarele:

#include se scrie pentru a include în program o bibliotecă în care sunt definite anumite funcţii şi operaţii necesare.

<iostream> este o bibliotecă de comenzi în care se găsesc de exemplu ***cin>>*** şi ***cout<<*** care ne permit citirea şi scrierea.

using namespace std; namespace ne permite să grupăm entităţile deja definite, precum variabilele, constantele, funcţiile şi să le folosim într-un anumit scop. Toate aceste entităţi ale librăriei C++ standard sunt declarate în interiorul std namespace.

main() este funcţia principală a programului.

int main() este funcţia principală a programului de tip întreg, care trebuie urmată de ***return*** ...

return 0 înseamnă că funcţia main întoarce (returnează) zero şi se termină programul.

{ } între acolade se scrie un bloc de cod. Obligatoriu orice acoladă deschisă trebuie închisă!

Exemplu: Afişaţi pe monitor un brăduţ format din caractere speciale.

pseudocod	**limbaj de programare C++**
scrie " * "; scrie " *|* "; scrie " **|** "; scrie"***|***"; stop	#include <iostream> using namespace std; int main() { cout<<" * "<<'\n'; cout<<" *|* "<<'\n'; cout<<" **|** "<<'\n'; cout<<"***|***"<<'\n'; return 0;}

Observaţie: Am utilizat ***'\n'*** pentru a marca sfârşitul afişării pe linia curentă, urmând ca afişarea următoare să se realizeze pe altă linie.

cout<< permite afişarea conţinutului unui şir de caractere, a conţinutului unei variabile sau expresii. Un şir de caractere, "***|***" se scrie între ghilimele duble, toate sus.

cin>> permite citirea unei variabile.

cin şi **cout** se citesc „si in” şi „si aut”. Tastatura este un dispozitiv de intrare cu ajutorul căreia citim (cin) datele de intrare (in). Monitorul este un dispozitiv de ieşire cu ajutorul căruia afişăm (cout) datele de ieşire (out).

Variabilele utilizate în C++ sunt entităţi care îşi pot modifica valoarea de-a lungul rulării unui program. Numele unei variabile poate fi format din litere, cifre şi caracterul special _, dar primul caracter nu poate fi cifră. De exemplu, o variabilă poate fi x, a1, var_7.

Exemplu: Scrieţi un program în C++ care citeşte un număr şi îl afişează.

pseudocod	limbaj de programare C++
întreg x; scrie „Dati un numar:”; citeşte x; scrie „Ati citit numarul:”; scrie x;	#include <iostream> using namespace std; int main() {int x; cout<<"Dati un numar:";cin>>x; cout<<"Ati citit numarul:"<<x; return 0;}

În cadrul algoritmilor, pentru efectuarea operaţiilor se utilizează expresiile care reprezintă succesiuni de operatori şi operanzi legaţi între ei.

Operatorii limbajului C++ se clasifică în:

- <u>operatori aritmetici</u>: + (plus), - (minus), * (înmulţire), / (împărţire), % (restul împărţirii întregi)

Operatorul / poate genera un rezultat diferit în funcţie de tipul de date al operanzilor. De exemplu, dacă ambii operanzi sunt întregi, operaţia are ca rezultat câtul împărţirii, iar dacă cel puţin un operand este de tip real, operaţia va avea ca rezultat o valoare reală.

Mai jos avem două exemple de utilizare a operatorului /:

int x, y, z;	**float x, y, z;**	**declararea variabilelor**
x=5; y=2;	x=5.0; y=2.0;	atribuirea unor valori variabilelor
z=x/y;	z=x/y;	rezultatul expresiei x/y este atribuit variabilei z
//z=2	//z=2.5	variabila z va avea valori diferite în funcţie de tipul de date declarat

Exemplu: Scrieţi un program în C++ care citeşte două numere şi afişează suma lor.

pseudocod	limbaj de programare C++
întreg x, y; scrie “x=”;	#include <iostream> using namespace std; int main()

citeşte x;
scrie “y=”;
citeşte y;
scrie “x+y=”;
scrie x+y;
stop
	{int x, y;
	cout<<"x=";cin>>x;
	cout<<"y=";cin>>y;
	cout<<"x+y="<<x+y;
	return 0; }

Exemplu: Scrieţi un program în C++ în care se citeşte un număr de trei cifre, se formează un număr cu cifrele inversate şi se afişează rezultatul. De exemplu, pentru x=123 se afişează 321.

pseudocod	limbaj de programare C++

întreg x, y;
scrie “x=”;
citeşte x;
y← x mod10*100+xdiv10mod10*10+ xdiv100;
scrie “y=”;
scrie y;
stop
	#include <iostream>
	using namespace std;
	int main()
	{int x,y;
	cout<<"x=";cin>>x;
	y=x%10*100+x/10%10*10+x/100;
	cout<<"y="<<y;
	return 0;}

Observaţie: Cu x%10 obţinem ultima cifră a numărului x, iar cu x/10 obţinem un număr care se obţine din primul număr după ce „tăiem” ultima cifră.

Exemplu: Scrieţi un program în C++ în care se citesc două numere întregi şi se afişează media aritmetică a lor.

pseudocod	limbaj de programare C++

întreg x, y;
scrie “x=”;
citeşte x;
scrie real (x+y)/2;
stop
	#include <iostream>
	using namespace std;
	int main()
	{int x, y;
	cout<<"x=";cin>>x;
	cout<<"y=";cin>>y;
	cout<<(float)(x+y)/2;
	return 0;}

Observaţie: În exemplul de mai sus, suma a două numere întregi este un număr întreg, care împărţit la alt număr întreg va avea ca rezultat câtul împărţirii. Pentru că noi dorim media aritmetică a două numere întregi, vom face o conversie explicită a unui operand către un număr real. O altă soluţie ar fi fost ca 2 să fie convertit la real, adăugând după el un punct (2.) sau (float).

- operatori relaţionali: > (mai mare), >= (mai mare sau egal), < (mai mic), <= (mai mic sau egal), = = (egalitate), != (inegalitate).

Operatorii = = şi != au ca rezultat 1 (adevărat) sau 0 (fals).

De exemplu: 7= =7 are rezultat 1, iar 5!=5 are rezultat 0.

- operatori de atribuire: =

Unei variabile i se poate atribui o constantă, conţinutul altei variabile sau rezultatul unei expresii.

```
x=2            //variabilei x i se atribuie valoarea 2
x=y            //variabilei x i se atribuie conţinutul variabilei y
x=2+7*y        // variabilei x i se atribuie rezultatul expresiei 2+7*y
```

- operatori de incrementare şi decrementare: + + (incrementare), - - (decrementare)

Incrementarea unei variabile reprezintă creşterea variabilei cu o unitate. De exemplu x++ este echivalent cu x=x+1 şi se spune că x a fost incrementat cu o unitate.

În funcţie de poziţia operatorului faţă de variabilă, se va numi operator prefixat (+ + x) sau operator postfixat (x + +) şi va acţiona asupra valorii variabilei înainte sau după evaluarea unei expresii.

Exemplu: În secvenţele de program C++ de mai jos sunt exemplificate incrementările şi decrementările prefixate şi postfixate.

int x=5,y=7;
cout<<"++x="<<++x<<'\n'; //creste x cu 1 inainte de a fi afisat
cout<<"y++="<<y++<<'\n'; //creste y cu 1 dupa ce va fi afisat
cout<<"y="<<y<<'\n';
cout<<"++x*++y="<<++x*++y<<'\n'; //cresc x si y cu 1 inainte de //efectuarea inmultirii
cout<<"2+x--="<<2+x--<<'\n'; //scade x cu 1 dupa ce va fi afisat
cout<<"x="<<x<<'\n';
	
	6
	7
	8
	63
	9
	6

int x=2,y=1; cout<<x-++y; //incrementarea prefixată ++y se face înainte de afişare	 0

int x=2,y=1;
cout<<x-y++; //incrementarea postfixată y++ se face după afişare
cout<<y;
	
	1
	2

int x=5,y=2;
cout<<x--*y; //decrementarea postfixată x-- se face după afişare
cout<<x;
	
	10
	4

- operatori logici: ! negare logică, && şi logic, || sau logic

Dacă avem un operand egal cu zero, operatorul ! aplicat în faţa operandului va schimba valoarea din zero în unu. Zero este valoarea utilizată pentru fals şi diferit de zero pentru adevărat.

Sau logic (||) este operator binar. Dacă cel puţin un operand are valoarea 1, rezultatul este 1, altfel, rezultatul este 0.

Şi logic (&&) este operator binar. Dacă cel puţin un operand are valoarea 0, rezultatul este 0, altfel, rezultatul este 1.

x	y	!x	x || y	x&&y
1	1	0	1	1
1	0	0	1	0
0	1	1	1	0
0	0	1	0	0

Exemplu: Care este valoarea de adevăr rezultată în urma operaţiilor de mai jos?

2+3>=5	1
(2+3>=5)&&(3+4>7)	0 deoarece din 1&&0 obţinem 0
!(2+3= =5)	0

Exemplu: Ştiind că x=2 şi y=5, scrieţi care este valoarea de adevăr rezultată în urma operaţiilor de mai jos?

y%x!=0	1
y/x!=2	0
(y>x)&&(x<2)||(y-3!=x)	0 provine de la 1&&0||0

- <u>operatorul condiţional</u> se utilizează atunci când dorim să evaluăm prin adevărat sau fals.

De exemplu cout<<(2<3?1:0) va afişa 1, deoarece se evaluează 2<3, dacă este adevărat se afişează prima valoare care se află după ?, iar dacă este fals se afişează a doua valoare.

Exemplu: Scrieţi un program în C++ în care se citeşte un număr întreg x. Dacă numărul este par se afişează mesajul *număr par* altfel se afişează *număr impar*.

```
#include <iostream>
using namespace std;
int main()
{int x;
   cout <<"x=";cin>>x;
   cout<<(x%2==0?"numar par":"numar impar");
   return 0;}
```

Un bit poate avea una dintre valorile: 1 sau 0. **Un octet (Byte sau B)** este format din 8 biţi şi este cea mai mică unitate de memorie care poate fi accesată în funcţie de adresa ei.

De exemplu pentru a memora numărul 5_{10} scriem 0 0 0 0 0 1 0 1

Pentru că cea mai mică unitate de memorie este octetul, atunci când dorim să reţinem un număr mic utilizăm o suprafaţă de memorie de 9 biţi, după cum vedem mai jos:

$1_{10} = 00000001_2$ (1 în baza 10 este egal cu 0000000 în baza 2)
$2_{10} = 00000010_2$
$3_{10} = 00000011_2$
$4_{10} = 00000100_2$
$5_{10} = 00000101_2$
$6_{10} = 00000110_2$
$7_{10} = 00000111_2$
$8_{10} = 00001000_2$
$9_{10} = 00001001_2$
$10_{10} = 00001010_2$
.....
$16_{10} = 00010000_2$
...
$128_{10} = 10000000_2$

Operaţia de adunare în baza doi este:

+	0	1
0	0	1
1	1	10

Operaţia de scădere în baza doi:
$0_2-0_2=0_2$
$1_2-0_2=1_2$
$0_2-1_2=1_2$ – împrumut 1
$1_2-1_2=0_2$
$10_2-0_2=10_2$
$10_2-1_2=1_2$
$10_2-10_2=0_2$

Exemplu: $1100_2-1_2=1011_2$

Operaţia de înmulţire în baza 2:
$0_2 * 0_2=0_2$
$1_2 * 0_2=0_2$
$0_2 * 1_2=0_2$
$1_2 * 1_2=1_2$

Exemplu: $11_2*11_2=1001_2$

- **operatori logici pe biţi** sunt cei care permit accesarea biţilor.
Ei sunt:

<<	operator de deplasare shift left	operator binar	deplasează spre stânga un număr de biţi egal cu al doilea operand 3<<1 returnează 6 unde $3_{10}=0011_2$, iar după deplasare $6_{10}=0110_2$
>>	operator de deplasare shift right	operator binar	deplasează spre dreapta un număr de biţi egal cu al doilea operand 3>>1 returnează 1 unde $3_{10}=0011_2$, iar după deplasare $1_{10}=0001_2$
&	şi pe biţi	operator binar	0&0=0 0&1=0 1&0=0 1&1=1
|	sau pe biţi	operator binar	0|0=0 0|1=1 1|0=1 1|1=1
^	sau exclusiv pe biţi	operator binar	disjuncţie exclusivă pe biţi 0^0=0 0^1=1 1^0=1 1^1=0
~	negare pe biţi	operator unar	complementariază ~2 returnează -3 ~-3 returnează 2

Exemplu: Deplasarea spre stânga a unui bit schimbă valoarea acelui bit. Mai jos avem deplasarea la stânga cu 1 şi cu 2.

```
#include <iostream>
using namespace std;
int main()
{  int x;
   x=2<<1; cout<<x<<'\n';
   x=4<<2; cout<<x<<'\n';
   x=3<<2; cout<<x<<'\n';
   return 0;}
```

2_{10} = 00000010_2
3_{10} = 00000011_2
4_{10} = 00000100_2

00000010 deplasare stânga cu 1=> 00000100 afişează 4
00000100 deplasare stânga cu 2=> 00010000 afişează 16
00000011 deplasare stânga cu 2=> 00001100 afişează 12

Exemplu: Deplasarea spre dreapta cu 1 şi cu 2.

#include <iostream> using namespace std; int main() { int x; x=8>>1; cout<<x<<'\n'; x=12>>2; cout<<x<<'\n'; x=16>>2; cout<<x<<'\n'; return 0;}	$8_{10} = 00001000_2$ $4_{10} = 00000100_2$ $12_{10} = 00001100_2$ $3_{10} = 00000011_2$ $16_{10} = 00010000_2$ 00001000 deplasare dreapta cu 1=> 00000100 afişează 4 00001100 deplasare dreapta cu 2=> 00000011 afişează 3 00010000 deplasare dreapta cu 2=> 00000100 afişează 4

Probleme propuse

1. Scrieţi un program în C++ în care să se afişeze pe două linii: numele şi prenumele vostru şi disciplina preferată.

2. Scrieţi un program în C++ în care să se afişeze cu ajutorul caracterelor speciale, un corp geometric pe patru linii.

3. Scrieţi un program în C++ în care să se afişeze pe prima linie primele cinci numere prime, cu virgulă între ele şi pe a doua linie dublul acestora.

4. Scrieţi un program în C++ în care să se afişeze pe o linie toate numerele pare şi pe o altă linie toate numerele impare care se află între numerele 17 şi 43.

5. * Scrieţi un program în C++ în care să se afişeze pe linii diferite toate numerele naturale a, b, c astfel încât fracţia $\frac{4}{a^2+b^2+c^2}$ să fie supraunitară.

6. * Scrieţi un program în C++ în care să se afişeze numerele naturale a, b, c astfel încât următoarele fracţii să fie echivalente: $\frac{2}{7}, \frac{a}{21}, \frac{b+1}{35}, \frac{48}{2c+4}$

7. Scrieţi un program în C++ în care să se citească două numere şi să afişeze produsul lor.

8. Scrieţi un program în C++ în care se citeşte preţul unei cărţi şi se afişează preţul cu 15% mai mare decât cel citit.

9. Scrieţi un program în C++ în care se citeşte un număr de trei cifre şi se afişează numărul fără cifra zecilor. Dacă se citeşte 289 se afişează 29.

10. Un elev obţine 3 note la matematică şi o notă în teză. Să se afişeze cu ajutorul unui program C++ media obţinută de elev.

11. Fiind dat un număr real x care reprezintă lungimea laturii unui pătrat, să se calculeze perimetrul şi aria pătratului şi să se afişeze cu un program C++ rezultatul.

12. * Se citesc trei cifre x, y, şi z. Să se calculeze şi să se afişeze rezultatul expresiei: $\overline{xyz} + \overline{zyx}$ cu ajutorul unui program C++.

13. Scrieţi un program în C++ în care se citesc două numere întregi. Să se afişeze numărul format cu ultima cifră de la fiecare număr citit.

14. Scrieţi un program în C++ în care se citeşte un număr de 4 cifre şi se afişează suma cifrelor numărului.

15. Scrieţi un program în C++ în care se citeşte un număr de 4 cifre şi se afişează cifrele numărului în sens invers. Dacă numărul este 1243 se afişează 3421.

16. Scrieţi valoarea de adevăr a următoarelor operaţii ştiind că x şi y sunt întregi şi au valorile x=2 şi y=17.

a) (y mod x ≠1) && (y div x =8)
b) not ((y + 2*x +1) mod 5 =0)
c) y div (x+1) > y+1 div x
d) (x ≠ y) || (x + 15 = y) && (y mod 3=(x+1) mod 2)

17. Scrieţi un program în C++ în care să se afişeze rezultatul sumei: 3+6+9+...+390.

Capitolul 2.

Tipuri simple de date

Noțiuni teoretice

Datele cu care lucrăm în cadrul algoritmilor se clasifică după tipul lor în: întregi, reale, logice, şir de caractere. Prin ***tip de date*** înţelegem o mulţime de valori, o mulţime de operaţii pe care le putem utiliza şi suprafaţa pe care o ocupă în memoria calculatorului. Spre deosebire de matematică unde numerele naturale pot avea valori infinite, în C++ un tip de date este limitat de tipul compilatorului şi de resursele calculatorului. De exemplu, noi vom lucra cu compilatorul Code::Blocks pe 32 biţi.

Tipurile de date întregi se clasifică în:

tip de date	**denumirea**	**reprezentarea în memoria calculatorului**	**mulţimea în care poate lua valori**
unsigned char	caracter fără semn	1 B sau 1 octet sau 8 biţi	0 ,… , 255
char	caracter	1 B sau 1 octet sau 8 biţi	-128, ..., 127
unsigned short	întreg scurt fără semn	2 B sau 2 octeţi sau 16 biţi	0, ..., 65535
short	întreg scurt	2 B sau 2 octeţi sau 16 biţi	-32768, ..., 32767
unsigned int	întreg fără semn	4 B sau 32 biţi	0, ..., 4 294 967 295
int	întreg	4 B sau 32 biţi	-2147483648, ..., 2147483647
unsigned long	întreg lung fără semn	4 B sau 32 biţi	0, ..., 4 294 967 295
long	întreg lung	4 B sau 32 biţi	-2147483648, ..., 2147483647

Tipul de date char este o mulţime ordonată şi finită de elemente din caracterele codului ASCII (American Standard Code for Information Interchange).

Exemplu: În exemplul de mai jos se iniţializează variabilele a, b, c, d cu caractere, respectiv A, Z, a, z şi se afişează pe o linie codul ASCII al literelor A şi Z, iar pe linia următoare codul ASCII al literelor a şi z.

```
#include <iostream>
using namespace std;
int main()
{char a, b, c, d;
a='A'; b='Z'; c='a'; d='z';
   cout <<(int)a<<", "<<(int)b<<'\n';
   cout <<(int)c<<", "<<(int)d;
   return 0;}
```

```
65, 90
97, 122
```

Exemplu: În exemplul de mai jos se iniţializează variabilele declarate cu codurile ASCII ale literelor A, Z, a, z afişează pe două linii literele.

```
#include <iostream>
using namespace std;
int main()
{int a, b, c, d, x;
a=65; b=90; c=97; d=122; x=10;
   cout<<(char)a<<", "<<(char)b<<(char)x;
   cout <<(char)c<<", "<<(char)d;
   return 0;}
```

```
A, Z
a, z
```

Observaţie: Pentru a afişa pe linia următoare, am utilizat codul ASCII 10. Mai jos este un tabel cu codurile ASCII cele mai uzuale.

cod ASCII	caracter	cod ASCII	caracter	cod ASCII	caracter
33	!	65	A	97	a
34	“	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	‘	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r

cod ASCII	caracter	cod ASCII	caracter	cod ASCII	caracter
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_	127	del
64	@	96	`		

Tipurile de date reale sunt: float, double şi long double. Numerele reale sunt cele cu zecimale. De exemplu, dacă un măr poate fi declarat întreg, o jumătate de măr este $\frac{1}{2}$, adică 0,5 care este număr real.

În rezolvarea problemelor, de cele mai multe ori, trebuie să efectuăm mai multe operaţii precum calculele matematice pentru a putea ajunge la rezultatul dorit. În acest sens, vom utiliza noţiunea de expresie.

Expresia reprezintă o combinaţie de operatori şi operanzi ce efectuează prelucrări ale datelor respectând regulile limbajului de programare.

Exemplu: Scrieţi un program în C++ care citeşte un număr întreg x. Atribuiţi dublul acestei valori variabilei y şi afişaţi rezultatul.

pseudocod	limbaj de programare C++

întreg x, y;
scrie “x=”;
citeşte x;
y← 2*x;
scrie “y=”;
scrie y;
stop
	#include <iostream>
	using namespace std;
	int main()
	{int x, y;
	cout <<"x="; cin>>x;
	y=2*x;
	cout <<"y="<<y;
	return 0;}

Observaţie: În pseudocod, atribuirea este reprezentată de ← (săgeată spre stânga).

Exemplu: Scrieţi o expresie în C++ pentru $E = \frac{x-y^2+\frac{z}{4}}{9-\frac{\frac{3}{x}-1}{y+2}}$.

E=(x-y*y+z/4)/(9-(3/x-1)/(y+2))

Observație: În C++ putem deschide oricâte paranteze avem nevoie, important este ca toate să fie rotunde şi orice paranteză care se deschide trebuie să se şi închidă.

Exemple corecte: (()), (((()(()))())

Exemple greşite: ())(, (()(()())

Exemplu: Scrieţi un program în C++ în care se citesc două numere x şi y şi se afişează media geometrică a lor.

pseudocod	limbaj de programare C++

real x, y, z;
scrie “x=”;
citeşte x;
scrie “y=”;
citeşte y;
z← $\sqrt{x * y}$;
scrie “z=”;
scrie z;
stop
	#include <iostream>
	#include<math.h>
	using namespace std;
	int main()
	{float x, y, mg;
	cout <<"x="; cin>>x;
	cout <<"y="; cin>>y;
	z=sqrt(x*y);
	cout <<"z="<<z;
	return 0;}

Observație: Pentru a scrie în C++ radical din x ($\sqrt{x}$), folosim funcţia sqrt(x) şi biblioteca de comenzi <math.h> în care se găsesc funcţii matematice.

Tipul de date logic:

Sunt date care pot avea doar două valori: 1 (adevărat) sau 0 (fals). Operatorii logici pe care îi folosim sunt !, &&, ||.

Exemplu: Scrieţi un program în C++ în care se citesc trei numere reale şi să se afişeze dacă pot forma un triunghi sau nu.

```
#include <iostream>
using namespace std;
int main()
{float x, y, z;
   cout <<"x=";cin>>x;
   cout <<"y=";cin>>y;
   cout <<"z=";cin>>z;
   if (x>0&&y>0&&z>0&&(x+y>z)&&(x+z>y)&&(y+z>x)
         cout<<"Se poate forma un triunghi":"nu se poate forma triunghi";
   return 0;}
```

Exemplu: Ana a pus la bancă o sumă de bani. Ştiind că banca dă dobândă 1% pe lună, câţi bani va avea în cont Ana după prima şi după a doua lună?

pseudocod	limbaj de programare C++
real x, p1, p2; scrie "Suma depusa in banca="; citeşte x;	#include <iostream> using namespace std; int main()

p1←x+x / 100;
p2←p1 / 100 + p1;
scrie "Suma dupa prima luna=";
scrie p1;
scrie "Suma dupa a doua luna="
scrie p2;
stop
	{float x, p1, p2;
	cout<<"Suma depusa in banca=";cin>>x;
	p1=x+x/100;
	p2=p1/100+p1;
	cout<<"Suma dupa prima luna="<<p1<<'\n';
	cout<<"Suma dupa a doua luna="<<p2;
	return 0;}

Exemplu: Se citesc două unghiuri la care cunoaştem gradul, minutul şi secunda. Adunaţi cele două unghiuri şi afişaţi rezultatul.

pseudocod	limbaj de programare C++

întreg g1, g2, g, m1, m2, m, s1, s2, s;
scrie "gradul unghiului 1:"; citeşte g1;
scrie "minutul unghiului 1:"; citeşte m1;
scrie "secunda unghiului 1:"; citeşte s1;
scrie "gradul unghiului 2:"; citeşte g2;
scrie "minutul unghiului 2:"; citeşte m2;
scrie "secunda unghiului 2:"; citeşte s2;
s←s1+s2;
m←m1+m2+s div 60;
s←s mod 60;
g←g1+g2+m div 60;
m←m mod 60;
scrie "suma unghiurilor= ";
scrie g; scrie" grade, ";
scrie m; scrie " minute, ";
scrie s; scrie " secunde ";
stop.
	#include <iostream>
	#include<math.h>
	using namespace std;
	int main()
	{int g1, g2, g, m1, m2, m, s1, s2, s;
	cout<<"gradul unghiului 1:";cin>>g1;
	cout<<"minutul unghiului 1:";cin>>m1;
	cout<<"secunda unghiului 1:";cin>>s1;
	cout<<"gradul unghiului 2:";cin>>g2;
	cout<<"minutul unghiului 2:";cin>>m2;
	cout<<"secunda unghiului 2:";cin>>s2;
	s=s1+s2;
	m=m1+m2+s/60;
	s=s%60;
	g=g1+g2+m/60;
	m=m%60;
	cout<<"suma unghiurilor= "<<g<<" grade,
	"<<m<<" minute, "<<s<<" secunde";
	return 0;}

Probleme propuse

1. Scrieţi un program în C++ în care să se afişeze valorile pentru cel mai mare short, int, long, float, double şi pentru cel mai mic float şi double.
2. Scrieţi un program în C++ în care se citesc două numere întregi. Fără a face suma lor, să se afişeze ultima cifră a sumei dintre cele două numere.

3. Scrieţi un program în C++ în care se citeşte un număr întreg x de patru cifre, calculează produsul cifrelor numărului în variabila y şi afişează y.

4. Scrieţi un program în C++ în care se citesc patru cifre, se construieşte numărul obţinut din cifrele citite şi se afişează rezultatul.

5. Scrieţi un program în C++ în care, fiind date două numere întregi x şi y mai mari decât 100, să se interschimbe valorile între ele şi să se afişeze rezultatul.

6. Scrieţi un program în C++ în care se citeşte un caracter de la tastatură şi se afişează codul ASCII al său.

7. * Scrieţi un program în C++ în care se citeşte de la tastatură o literă mică a alfabetului şi se afişează litera mare corespunzătoare.

8. Scrieţi un program în C++ în care se citesc trei numere întregi x, y şi z, se efectuează atribuirile de mai jos şi se afişează rezultatul.

a) $E1 = \frac{x+\frac{x-y}{2+z}}{z}$

b) $E2 = \frac{2y*\frac{z-\frac{y}{z}}{3}}{\frac{8-\frac{y}{z}}{z}}$

c) $E3 = \frac{x+\frac{x^3-y}{2+\frac{y}{z}}}{\frac{7}{2}-z}$

9. Se citesc trei numere reale ce reprezintă laturile unui triunghi. Să se calculeze şi să se afişeze aria triunghiului într-un program C++.

10. * Ionel trebuie să ajungă din punctul A în punctul B din 4 paşi. Afişaţi distanţa dintre A şi B ştiind că fiecare pas se măreşte cu 30%. Exemplu: Dacă p_1=10, atunci distanţa dintre A şi B este 61.87.

11. * Scrieţi un program în C++ în care ştiind că $\overline{yz}$ este egal cu 4% din numărul $\overline{xyz}$, calculaţi şi afişaţi x+y+z.

12. Ştiind că x=3 şi y=-2, scrieţi care este valoarea de adevăr rezultată în urma operaţiilor de mai jos.

a) (x%2!=y+3)||(x+y>1)
b) (x>y)&&(x-y>0)||(x+y==1)
c) (x/2==1)&&(y<0)||(2*x+y==8)

13. Scrieţi un program în C++ în care se citesc 4 numere întregi de la tastatură. Să se afişeze suma numerelor pare şi produsul numerelor impare citite.

14. Scrieţi un program în C++ în care se citesc 4 numere întregi de la tastatură. Să se afişeze media aritmetică a numerelor pozitive şi produsul numerelor divizibile cu 3.

15. Scrieţi un program în C++ în care se citesc două numere întregi x şi y. Să se afişeze dacă fracţia $\frac{x}{y}$ este subunitară sau supraunitară.

16. Scrieţi un program în C++ în care se citeşte un număr de 4 cifre şi se afişează dacă primele două cifre pot forma un număr mai mare decât ultimele două sau nu.

17. * Scrieţi un program în C++ în care se citesc numerele $\overline{xy}, \overline{yx}, \overline{xx}$. Ştiind că media aritmetică a lor este 11, afişaţi cifrele x şi y.

18. * Robert a primit cadou de ziua lui o sumă de bani x şi a strâns din economiile lui în fiecare lună cu 23% mai mult decât în luna precedentă. Scrieţi un program în C++ în care se afişează câţi bani a strâns Robert după trei luni, dacă în fiecare lună a cheltuit cu 15% mai puţin decât a primit de ziua lui.

Capitolul 3.

Structurile liniară, alternative, repetitive

Noțiuni teoretice

Structura liniară reprezintă modalitatea de a realiza un program prin care executarea liniilor de cod se face în ordinea în care sunt scrise.

Problemă rezolvată

Scrieţi un program în C++ în care se citesc două numere şi se afişează rezultatul expresiilor următoare: E1=2x+3y, E2=x^3-y, E3= $\sqrt{x-\frac{y}{2}}$

pseudocod	limbaj de programare C++

întreg x, y, E1, E2;
real E3;
scrie “x=”;
citeşte x;
scrie “y=”;
citeşte y;
E1← 2*x+3*y;
E2← x*x*x-y;
E3← $\sqrt{x-y/2.0}$;
scrie "E1="; scrie E1; scrie endl;
scrie "E2="; scrie E2; scrie endl;
scrie "E3="; scrie E3;
stop
	#include <iostream>
	#include<math.h>
	using namespace std;
	int main()
	{int x, y, E1, E2;
	float E3;
	cout <<"x="; cin>>x;
	cout <<"y="; cin>>y;
	E1=2*x+3*y;
	E2=x*x*x-y;
	E3=sqrt(x-y/2.0);
	cout<<"E1="<<E1<<'\n';
	cout<<"E2="<<E2<<'\n';
	cout<<"E3="<<E3;
	return 0;}

Noțiuni teoretice

Structurile alternative permit alegerea unei variante din două sau mai multe variante posibile. Ele se clasifică în structură alternativă simplă şi structură alternativă generalizată.

Structura alternativă simplă:
În pseudocod o notăm cu *dacă*, iar în C++ cu *if*.

Forma generală:

dacă (expresie) atunci
  instrucţiune 1;
[altfel
  instrucţiune 2;]
	if (expresie)
	  instrucţiune 1;
	[else
	  instrucţiune 2;]

Paşii care sunt făcuţi în executarea structurii sunt:
P1: Se evaluează expresia
P2: Dacă valoarea de adevăr a expresiei este adevărat (1) se execută instrucţiune 1;
P3: Dacă valoarea de adevăr a expresiei este fals (0) se execută instrucţiune 2.
Observaţie: Este opţională scrierea în cadrul structurii if (dacă) a părţii else (altfel)!

Probleme rezolvate

Problema 1. Scrieţi un program în C++ în care se citeşte un număr, iar dacă este pozitiv se afişează un mesaj.

întreg x;
scrie “x=”; citeşte x;
dacă (x>0) atunci
  scrie “Pozitiv”;
	#include <iostream>
	using namespace std;
	int main()
	{int x;
	cout <<"x="; cin>>x;
	if(x>0)
	  cout<<"Pozitiv";
	return 0;}

În exemplul de mai sus, dacă numărul este negativ nu se va afişa nimic.
Observaţie: În cazul în care în interiorul unei structuri trebuie să executăm mai mult de o instrucţiune, atunci acele instrucţiuni trebuie să le scriem între acolade.

Problema 2. Scrieţi un program în C++ în care se rezolvă ecuaţia de gradul I. Ştim că forma generală a ecuaţiei este ax+b=0.

```
real a, b, x;
scrie “a=”;
citeşte a;
scrie “b=”;
citeşte b;
    dacă (a=0) atunci
      scrie“Ecuaţie imposibilă”;
    altfel
     x←-b/a;
     scrie “x=”;
     scrie x;
stop
```

```
#include <iostream>
using namespace std;
int main()
{float a, b, x;
cout <<"a="; cin>>a;
cout <<"b="; cin>>b;
if(a= =0)
   cout<<"Ecuatie imposibila";
else {x=-b/a;
    cout<<"x="<<x;}
return 0;}
```

Problema 3. Scrieţi un program în C++ în care se citeşte x număr real, se calculează şi se afişează expresia de mai jos:

$$E=\begin{cases} x^2+\sqrt{x}-1, & pentru \quad x<-1 \\ \dfrac{\sqrt{x+7}+9}{12}, & pentru \quad x\geq -1 \end{cases}$$

```
real x, E;
scrie “x=”;
citeşte x;
    dacă (x<-1) atunci
    E← x²+ √x -1
     altfel
    E← (√(x+7)+9)/12
scrie ”E=”; scrie E;
stop
```

```
#include <iostream>
#include<math.h>
using namespace std;
int main()
{float x, E;
   cout << "x=";cin>>x;
   if(x<-1)
      E=x*x+sqrt(x)-1;
      else
         E=(sqrt(x+7)+9)/12;
      cout<<"E="<<E;
   return 0;
}
```

Problema 4. Scrieţi un program în C++ în care se citeşte x număr întreg şi se afişează dacă este sau nu pătrat perfect.

```
real x;
scrie “x=”;
citeşte x;
dacă (intreg( √x )= √x ) atunci
   scrie "Este patrat perfect";
```

```
#include <iostream>
#include<math.h>
using namespace std;
int main()
{int x;
   cout << "x=";cin>>x;
```

altfel
scrie "Nu este patrat perfect";
 	
stop
	if(int(sqrt(x))==sqrt(x))
	cout<<"Este patrat perfect";
	else
	cout<<"Nu este patrat perfect";
	return 0;
	}

Noțiuni teoretice

Structura alternativă generalizată permite alegerea unei variante de acelaşi tip din mai multe posibile.

Forma generală:

switch (selector=caz_c) cazul c1: expresie 1; întrerupe; cazul c2: expresie 2; întrerupe; ..... cazul cn: expresie n; întrerupe; altfel expresie n_1;	switch (expresie) case c1: instrucţiune 1; [break;] case c2: instrucţiune 2; [break;] ... case cn: instrucţiune n; [break;] [else instrucţiune n+1;]

Paşii care sunt făcuţi în executarea structurii sunt:
- Se evaluează expresia dintre paranteze
- Se evaluează pe rând expresiile c1, c2, ..cn
- Dacă valoarea expresiei este egală cu c1, se execută instrucţiune 1;
- Dacă valoarea expresiei este egală cu c2, se execută instrucţiune 2;
...
- Dacă valoarea expresiei este diferită de toate valorile de mai sus, se execută instrucţiune n+1;

Probleme rezolvate

Problema 1. Scrieţi un program în C++ în care se citeşte un număr întreg. Dacă numărul este între 1 şi 4, se afişează anotimpul corespunzător, altfel se afişează un mesaj.

```
#include<iostream>
using namespace std;
int main()
```

```
{int x;
 cout<<"x=";cin>>x;
 switch(x)
 {case 1: cout<<"Primavara"; break;
  case 2: cout<<"Vara"; break;
  case 3: cout<<"Toamna"; break;
  case 4: cout<<"Iarna"; break;
  default: cout<<"nr. nu este intre 1 si 4";
  }
return 0;}
```

Problema 2. Scrieţi un program în C++ în care se citeşte un operator aritmetic şi două numere întregi. În funcţie de operatorul citit, afişaţi rezultatul operaţiei x operator y.

```
#include<iostream>
using namespace std;
int main()
{int x, y, z;
 char opr;
 cout<<"x=";cin>>x;
 cout<<"y=";cin>>y;
 cout<<"dati operatorul aritmetic: +, -, *, /, %: "; cin>>opr;
 switch(opr)
 {case '+': z=x+y; break;
  case '-': z=x-y; break;
  case '*': z=x*y; break;
  case '/': z=x/y; break;
  case '%': z=x%y; break;
  default: cout<<"nu ati dat un operator aritmetic";
  }
  cout<<z;
return 0;}
```

Noțiuni teoretice

Structuri repetitive

De multe ori, pentru a rezolva o problemă, trebuie să repetăm de mai multe ori o instrucţiune sau o secvență de instrucţiuni, de aceea vom învăța despre structuri repetitive. Structurile repetitive se clasifică după cum urmează:

- structură repetitivă cu test iniţial *while (cât timp)*;
- structură repetitivă cu test final *do while (execută cât timp)*;
- structură repetitivă cu număr cunoscut de paşi *for (pentru)*

Structura repetitivă cu test iniţial while (cât timp) are următoarea formă generală:

cât timp (expresie) execută instrucţiune; ■	while (expresie) instrucţiune;

Paşii care sunt făcuţi în executarea structurii sunt:

P1: Se evaluează expresia dintre paranteze. Dacă valoarea de adevăr a expresiei este adevărat, se execută instrucţiunea, după care se reia pasul P1, altfel se execută pasul P2.

P2: Structura repetitivă se încheie şi se efectuează instrucţiunea imediat următoare.

Problemă rezolvată

Scrieţi un program în C++ în care se citesc n numere întregi şi se afişează suma lor.

întreg n, x, s, i;
scrie "n=";
citeşte n;
s←0;
i←1;
cât timp (i<=n) execută
scrie "x="; citeşte x;
s←s+x;
i←i+1;
■
scrie s;
stop
	#include<iostream>
	using namespace std;
	int main()
	{int n, x, s, i;
	cout<<"n=";cin>>n;
	s=0;i=1;
	while(i<=n)
	{ cout<<"x=";cin>>x;
	s=s+x;
	i=i+1;}
	cout<<s;
	return 0;}

În programul de mai sus am făcut următoarele notaţii:

- n, numărul de elemente pe care le vom citi;

- x, numărul pe care îl citim. Acest număr poate fi de fiecare dată altul, pentru fiecare buclă a structurii repetitive. În aceeaşi variabilă x putem reţine pe rând mai multe numere. Avem o suprafaţă clar delimitată în memoria calculatorului unde reţinem conţinutul variabilei. Pe acea suprafaţă se poate reţine doar o singură valoare la un moment dat.

- s, suma numerelor citite (suma x-urilor). Suma s trebuie iniţializată cu o valoare neutră faţă de adunare şi acea valoare este zero, de aceea am atribuit s=0.

- i, contorul care ne permite să numărăm de la 1 la n pentru a aduna doar n numere.

Noțiuni teoretice

Structura repetitivă cu test final do while (execută cât timp) are următoarea formă generală:

execută instrucţiune; cât timp (expresie); ■	do instrucţiune; while (expresie);

Paşii care sunt făcuţi în executarea structurii sunt:

P1: Se execută instrucţiunea. Se evaluează expresia dintre paranteze. Dacă valoarea de adevăr a expresiei este adevărat, se reia pasul P1, altfel se execută pasul P2.

P2: Structura repetitivă se încheie şi se efectuează instrucţiunea imediat următoare.

Observație: Spre deosebire de structura while, structura do while se execută cel puţin o dată înainte de a fi evaluată expresia.

Problemă rezolvată

Scrieţi un program în C++ în care se citeşte n număr întreg şi se afişează suma numerelor de la 1 la n.

întreg n, s, i;
scrie “n=”;
citeşte n;
s←0;
i←1;
execută
s←s+i;
i←i+1;
cât timp (i<=n);
■
scrie s;
stop
	#include<iostream>
	using namespace std;
	int main()
	{int n, s, i;
	cout<<"n=";cin>>n;
	s=0;i=1;
	do
	{s=s+i;i=i+1;}
	while(i<=n);
	cout<<s;
	return 0;
	}

Noțiuni teoretice

Structura repetitivă cu număr cunoscut de paşi for (pentru) are următoarea formă generală:

pentru contor ←val_initiala, val_final execută instrucţiune; ■	for(exp_ini; exp_cond; exp_inc) instrucţiune;

Paşii care sunt făcuţi în executarea structurii sunt:

P1: Se iniţializează contorul (exp_ini);

P2: Se evaluează exp_cond. Dacă valoarea de adevăr a exp_cond este adevărat se execută instrucţiune, se execută exp_inc şi se reia pasul P2, altfel se execută pasul P3.

P3: Structura repetitivă se încheie şi se efectuează instrucţiunea imediat următoare.

Observaţie: Spre deosebire de structurile while şi do while, structura for conţine iniţializarea şi incrementarea contorului şi se execută într-un număr cunoscut de paşi.

Probleme rezolvate

Problema 1. Scrieţi un program în C++ în care se citeşte n număr întreg şi se afişează produsul numerelor de la 1 la n.

întreg n, p, i;
scrie "n=";
citeşte n;
p←1;
 	
pentru i←1, n execută
s←s+i;
■
scrie p;
stop
	#include<iostream>
	using namespace std;
	int main()
	{int n, p, i;
	cout<<"n=";cin>>n;
	p=1;
	for(i=1;i<=n;i++)
	p=p*i;
	cout<<p;
	return 0;}

Problema 2. Scrieţi un program în C++ în care se citesc n numere şi se afişează câte sunt pare şi câte sunt divizibile cu 3.

întreg n, x, i, par, d3; i←1; par←0; d3←0;	#include<iostream> using namespace std; int main()

scrie “n=”; citeşte n;
cât timp i<=n execută
scrie “x=”; citeşte x;
dacă x mod 2 = 0 execută
par←par+1;
dacă x mod 3 = 0 execută
d3←d3+1;
i←i+1;
scrie “numere pare= ”; scrie par;
scrie “”numere divizibile cu 3= ”;
scrie d3;
stop
	{int n, x, i=1, par=0, d3=0;
	cout<<"n=";cin>>n;
	while(i<=n)
	{cout<<"x=";cin>>x;
	if(x%2==0)
	par++;
	if(x%3==0)
	d3++;
	i++;}
	cout<<"numere pare="<<par<<'\n';
	cout<<"numere divizibile cu 3="<<d3;
	}

Problema 3. Scrieţi un program în C++ în care se citesc trei numere întregi x, y, z, unde x∈[1,8], y∈[0, 9], z∈[0, 9]. Să se afişeze de x ori numărul format din y şi z după următoarele criterii:

- pe prima linie se afişează numărul format din primele x cifre y urmate de z
- pe a doua linie se afişează numărul format din primele x-1 cifre y urmate de z
- pe ultima linie numărul $\overline{yz}$

De exemplu dacă se citesc numerele 3, 8, 9, se afişează:

889
889
89

```
#include <iostream>
#include<math.h>
using namespace std;
int main()
{int x, y, z, i, j, nr,k;
   do{
   cout << "x=";cin>>x;
   cout << "y=";cin>>y;
   cout << "z=";cin>>z;
   }while(x<1||x>8||y<0||y>9||z<0||z>9);
   k=x;
   for(i=1;i<=x-1;i++)
   {nr=0;
      for(j=1;j<=k;j++)
       nr=nr*10+y;
      nr=nr*10+z;
      cout<<nr<<'\n';
      k=k-1;
   }
   cout<<y*10+z;
   return 0;
}
```

Probleme propuse

1. Scrieţi un program în C++ în care se citesc a, b, c şi x şi se afişează rezultatul expresiilor următoare:

a) $E1 = \sqrt{a+b} - \sqrt{a-b}$;

b) $E2 = \dfrac{a^4 - b}{45 + \sqrt{c}}$;

c) $E3 = ax^2+bx+c$.

2. Fiind dată raza unui cerc, să de afişeze aria şi lungimea cercului. Pentru numărul π se va folosi constanta PI.

3. Fiind date două numere reale pozitive b, h ce reprezintă baza şi înălţimea unui triunghi isoscel, să se calculeze aria triunghiului.

4. Fiind dat un număr real pozitiv x ce reprezintă latura unui triunghi echilateral, să se calculeze şi să se afişeze aria şi înălţimea triunghiului.

5. Fiind date două numere reale pozitive ce reprezintă catetele unui triunghi dreptunghic, să se calculeze şi să se afişeze aria şi înălţimea triunghiului.

6. Se citesc trei numere reale pozitive x, y şi h care reprezintă lăţimea, lungime şi înălţimea unui paralelipiped. Să se afişeze volumul, aria suprafeţei laterale şi diagonala.

7. * Fiind date trei numere reale a, b şi c, să se rezolve ecuaţia de gradul doi $ax^2+bx+c=0$.

8. Scrieţi un program în C++ în care se citeşte un număr real x1. Să se verifice dacă x1 este soluţie a ecuaţiei $5x^2-7x+2=0$.

9. * Fiind date trei numere reale a, b şi c, să se verifice dacă pot fi laturile unui triunghi, iar în caz afirmativ să se afişeze dacă acesta este isoscel, echilateral sau dreptunghic.

10. Să se calculeze valorile următoarelor expresii:

a) $E1 = \begin{cases} a+b, & pentru \quad a < 0 \\ \dfrac{a-2}{b}, & pentru \quad a \geq 0 \end{cases}$

b)* $E2=\begin{cases}\sqrt{a+b}, & pentru \quad a<2\\ \sqrt{a-2}, & pentru \quad a=2\\ \max(a,b) & pentru \quad a>2\end{cases}$

c)* $E3=\begin{cases}a+\sqrt{b}, & pentru \quad a<3\\ \frac{a}{b}, & pentru \quad 3\le a\le 4\\ a-b, & pentru \quad a>4\end{cases}$

11. Fiind date două numere întregi a şi b şi expresia E=|a-2b|, să se evalueze şi să se afişeze rezultatul.

12. Se citesc patru numere reale a, b, c, d. Să se afişeze dacă intervalele [a, b] şi [c, d] se intersectează sau nu.

13. Scrieţi un program în C++ în care se citeşte un număr întreg. Dacă numărul este între 1 şi 7, se afişează ziua corespunzătoare din săptămână, altfel se afişează un mesaj.

14. Să se calculeze suma numerelor pare până la un număr întreg n.

15. Scrieţi un program în C++ în care se citesc n numere. Să se afişeze câte sunt pare şi câte sunt divizibile cu 3.

16. Scrieţi un program în C++ în care se citesc numere până la întâlnirea lui 0. Să se afişeze dacă sunt ordonate crescător, descrescător sau nu sunt ordonate.

17. Să se afişeze maximul şi minimul dintre n numere întregi citite de la tastatură.
18. Să se calculeze suma numerelor întregi până la citirea numărului 0.

19. Se citesc numere până la întâlnirea numărului -1. Să se calculeze şi să se afişeze media aritmetică a numerelor divizibile cu 5.

20. Să se calculeze suma numerelor pare din intervalul închis [a, b], unde a şi b sunt numere întregi şi a<b.

21. Să se calculeze şi să se afişeze următoarele sume:

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

b) $S = \frac{1}{2} - \frac{2}{3} + \frac{3}{4} ... \pm \frac{n}{n+1}$

c) $S = 1 + 3^2 + 5^2 + ... + (2n-1)^2$

22. Scrieţi un program în C++ în care se citeşte un număr pozitiv şi se afişează dacă este an bisect sau nu.

23. Scrieţi un program în C++ în care se citeşte un număr întreg x. Să se afişeze numărul de cifre al numărului citit.

24. Scrieţi un program în C++ în care se citesc trei numere şi se afişează cel mai mare număr care se poate obţine cu ultima cifră de la fiecare.

25. ** Maria a aflat că numerele naturale care încep cu cifra 1 și au toate cifrele ordonate strict crescător şi consecutive sau încep cu cifra 9 și au toate cifrele ordonate strict descrescător şi consecutive se numesc numere speciale. Interesată să descopere legătura dintre numerele speciale cu același număr de cifre, a observat că poate construi tabelul alăturat.

1	1X8 +1= 9
2	12X8 +2= 98
3	123X8 +3= 987
4	1234X8 +4= 9876
5	12345X8 +5= 98765
6	123456X8 +6= 987654
7	1234567X8 +7= 9876543
8	12345678X8 +8= 98765432
9	123456789X8 +9= 987654321

Cerinţe:
Scrieţi un program care citind patru numere naturale K, N, A şi B determină:
1) cel mai mare număr special situat în tabel pe linia K;
2) numărul special obţinut din numărul N prin ştergerea unei cifre;
3) numărul de numere speciale din mulțimea {A , A +1, A+2, A+3…, B-1, B}.

Date de intrare:
Fişierul de intrare speciale.in conţine pe prima linie un număr natural P. Pentru toate testele de intrare, numărul P poate avea doar valoarea 1, valoarea 2 sau valoarea 3. Pe a doua linie a fișierului speciale.in se găsesc, în această ordine, numerele naturale K, N, A și B, separate prin câte un spațiu.
Date de ieşire:
Dacă valoarea lui P este 1, se va rezolva numai punctul 1) din cerințe. În acest caz, fişierul de ieşire speciale.out va conține pe prima linie un număr natural reprezentând cel mai mare număr special situat în tabel pe linia K.
Dacă valoarea lui P este 2, se va rezolva numai punctul 2) din cerințe. În acest caz, fişierul de ieşire speciale.out va conține pe prima linie un număr natural

reprezentând numărul special obţinut din numărul N prin ştergerea unei cifre sau 0 dacă un astfel de număr nu se poate obţine;
Dacă valoarea lui P este 3, se va rezolva numai punctul 3) din cerințe. În acest caz, fişierul de ieşire speciale.out va conține pe prima linie un număr natural reprezentând numărul de numere speciale din mulțimea {A, A+1, A+2, A+3, …, B-1, B}.

Restricţii:

- $1 \leq K \leq 9$
- $1 \leq N \leq 999999999$
- $1 \leq A \leq B \leq 999999999$
- Pentru rezolvarea corectă a primei cerinţe se acordă 20 de puncte, pentru rezolvarea corectă a celei de a doua cerințe se acordă 40 de puncte, pentru rezolvarea corectă a celei de a treia cerințe se acordă 40 de puncte.

Exemple:

speciale.in	**speciale.out**	**Explicaţii**
1 3 125345 320 888888	987	P = 1, pentru acest test, se rezolva cerința 1). Numerele speciale de pe linia a treia a tabelului sunt 123 și 987, cel mai mare fiind 987.

speciale.in	**speciale.out**	**Explicaţii**
2 3 125345 320 888888	12345	P = 2, pentru acest test, se rezolva cerința 2). Ștergând cifra 5 aflată pe poziția a treia în 125345 se obține numărul special 12345

speciale.in	**speciale.out**	**Explicaţii**
3 3 125345 320 888888	6	P = 3, pentru acest test, se rezolvă cerința 3). Sunt 6 numere speciale în mulțimea {320, 321,..., 888888}și anume 987, 1234, 9876, 12345, 98765, 123456

(OJI 2015, clasa a V-a)

Capitolul 4.

Algoritmi care prelucrează tipuri de date simple

Probleme rezolvate

Problema 1. Suma cifrelor unui număr

Scrieţi un program în C++ în care se citeşte un număr pozitiv. Să se afişeze suma cifrelor lui.

întreg n, s;
scrie “n=”;
citeşte n;
s←0;
cat timp n≠0 execută
  s←s+n mod 10;
  n←n div 10;
■
scrie “Suma cifrelor=”;
scrie s;
stop
	#include<iostream>
	using namespace std;
	int main()
	{ unsigned int n, s;
	  cout<<"n=";cin>>n;
	  s=0;
	  while (n!=0)
	  {s=s+n%10;
	  n=n/10;
	  }
	cout<<"Suma cifrelor="<<s;
	return 0;}

În programul de mai sus am citit un număr întreg pozitiv, după care am iniţializat suma cifrelor cu zero. Într-o structură repetitivă am executat doi paşi, *<la sumă am adunat ultima sumă şi restul împărţirii la 10 a numărului, după care am modificat n în aşa fel încât să conţină câtul împărţirii la 10, adică acelaşi număr fără ultima cifră>* cât timp numărul este diferit de zero, după care am afişat suma cifrelor.

Mai jos vom parcurge paşii care se execută atunci când se citeşte n=264.

-s=0

-se verifică dacă n!=0, adică 264!=0. 264!=0 este Adevărat şi se execută

- s=s+n%10, adică s=0+264%10, adică s=4
- n=n/10, adică n=264/10, adică n=26

-se verifică dacă n!=0, adică 26!=0. 26!=0 este Adevărat şi se execută

- s=s+n%10, adică s=4+6, adică s=10
- n=n/10, adică n=26/10, adică n=2

-se verifică dacă n!=0, adică 2!=0. 2!=0 este Adevărat şi se execută

- s=s+n%10, adică s=10+2, adică s=12

- n=n/10, adică n=2/10, adică n=0

-se verifică dacă n!=0, adică 0!=0. 0!=0 este Fals şi ieşim din bucla repetitivă

- se afişează *Suma cifrelor=12*

Problema 2. Prelucrări ale cifrelor unui număr

Scrieţi un program în C++ în care se citeşte un număr pozitiv. Să se afişeze de câte ori apare fiecare cifră a numărului. De exemplu, dacă citim 155155155 se va afişa:

Cifra 1 apare de 3 ori.

Cifra 5 apare de 6 ori.

În programul de mai sus am verificat toate cifrele numărului începând cu ultima, dacă sunt egale cu cifrele de la zero la nouă. Numărul a fost salvat la fiecare buclă repetitivă şi s-a afişat numărul de apariţii a cifrelor.

```
întreg x, nr, i, aux;
scrie “x=”;
citeşte x;
pentru i←0,9 execută
aux←x;
nr←0;
cât timp aux≠0 execută
  dacă aux mod 10=i atunci
  nr←nr+1;
  aux←aux div 10;
dacă nr≠0
  scrie “cifra”; scrie i;
  scrie “apare de”; scrie nr;
  scrie “ori”
stop
```

```
#include <iostream>
#include<math.h>
using namespace std;
int main()
{long x, nr, i, aux;
 cout<<"x=";cin>>x;
 for(i=0;i<=9;i++)
 {aux=x;
  nr=0;
  while(aux!=0)
    {if(aux%10==i)
     nr++;
    aux=aux/10;}
    if(nr!=0)
       cout<<"cifra "<<i<<" apare de "<<nr<<" ori "<<'\n';
    }
return 0;
}
```

Problema 3. Răsturnatul unui număr

Scrieţi un program în C++ în care se citeşte un număr pozitiv. Să se afişeze numărul răsturnat. Dacă se citeşte n=235 se afişează 532.

```
întreg n, r;
scrie “n=”;
citeşte n;
r←0;
```

```
#include<iostream>
using namespace std;
int main()
{ int n, r;
```

cat timp n≠0 execută
  r←r*10+n mod 10;
  n←n div 10;
 	
scrie “Rasturnatul=”;
scrie r;
stop
	cout<<"n=";cin>>n;
	r=0;
	while (n!=0)
	  {r=r*10+n%10;
	  n=n/10;
	  }
	cout<<"Rasturnatul="<<r;
	return 0;
	}

În programul de mai sus am citit un număr întreg pozitiv, după care am iniţializat răsturnatul cu zero. Într-o structură repetitivă am executat doi paşi, *<la răsturnat am adunat ultimul răsturnat înmulţit cu 10 şi restul împărţirii la 10 a numărului, după care am modificat n în aşa fel încât să conţină câtul împărţirii la 10, adică acelaşi număr fără ultima cifră>* cât timp numărul este diferit de zero, după care am afişat răsturnatul numărului.

Problema 4. Număr prim

Scrieţi un program în C++ în care se citeşte un număr întreg x de la tastatură. Să se afişeze dacă numărul este prim sau nu.

întreg n, prim, i;
scrie “n=”;
citeşte n;
prim←1;
pentru i←2, $\sqrt{n}$ execută
  daca (n mod i=0)
  prim=0;
dacă (prim≠0)
  scrie “nr. prim”;
altfel
  scrie “nr. NU este prim”;
stop
	#include<iostream>
	#include<math.h>
	using namespace std;
	int main()
	{int n, prim, i;
	cout<<"n=";cin>>n;
	prim=1;
	for(i=2;i<=sqrt(n);i++)
	  if(n%i==0)
	  prim=0;
	if(prim)
	  cout<<"nr. prim";
	else
	  cout<<"nr. NU este prim";
	return 0;
	}

Problema 5. Divizori

Scrieţi un program în C++ în care se afişează toţi divizorii unui număr întreg.

întreg x, i;
scrie “x=”; citeşte x;
pentru i←1, x div 2 execută
dacă x mod i=0 atunci
  scrie –i; scrie “,”; scrie i;
	#include <iostream>
	using namespace std;
	int main()
	{int x, i;
	  cout <<"x="; cin>>x;

scrie –x; scrie “, “; scrie x;
stop
	for(i=1;i<=x/2;i++)
	if(x%i==0)
	cout<<-i<<", "<<i<<'\n';
	cout<<-x<<", "<<x;
	return 0;
	}

Problema 6. Cel mai mare divizor comun – algoritmul lui Euclid

Scrieţi un program în C++ în care se citesc două numere întregi şi se afişează cmmdc dintre cele două numere citite. Dacă se citesc numerele 12 şi 15, se va afişa 3.

întreg a,b, x, y;
scrie “a=”; citeşte a;
scrie “b=”; citeşte b;
x←a;
y←b;
cât timp a ≠ b execută
daca (a>b) execută
a←a-b;
altfel
b←b-a;
■
scrie”cmmdc("; scrie x; scrie “,”;
scrie y; scrie “)=”; scrie a;
stop
	#include<iostream>
	using namespace std;
	int main()
	{int a, b, x, y;
	cout<<"a=";cin>>a;
	cout<<"b=";cin>>b;
	x=a; y=b;
	while(a!=b)
	if(a>b)
	a=a-b;
	else
	b=b-a;
	cout<<"cmmdc("<<x<<", "<<y<<")="<<a;
	return 0;}

Problema 7. Cel mai mic multiplu comun

Scrieţi un program în C++ în care se citesc două numere întregi şi se afişează cmmmc dintre cele două numere citite. Dacă se citesc numerele 12 şi 15, se calculează cmmmc(12,15)=2^2*3*5 şi se afişează 60.

întreg a,b, x, y;
scrie “a=”; citeşte a;
scrie “b=”; citeşte b;
x←a;
y←b;
cât timp a ≠ b execută
dacă (a>b) execută
a←a-b;
altfel
b←b-a;
■
scrie”cmmmc("; scrie x; scrie “,”;
scrie y; scrie “)=”; scrie (x*y) div a;
	#include<iostream>
	using namespace std;
	int main()
	{int a, b, x, y;
	cout<<"a=";cin>>a;
	cout<<"b=";cin>>b;
	x=a; y=b;
	while(a!=b)
	if(a>b)
	a=a-b;
	else
	b=b-a;

stop	cout<<"cmmmc("<<x<<", "<<y<<")="<<x*y/a; return 0;}

Problema 8. Transformarea unui număr din baza 10 în baza b, unde 2≤b≤10.

Scrieţi un program în C++ în care se citeşte un număr în baza 10 şi se transformă în baza b.

întreg x, baza, aux, n, p;
scrie "x="; citeste x;
aux←x;
scrie "baza="; citeste baza;
n←0;
p←1;
cât timp (x≠0) execută
  n←n+p*(x mod baza);
  p←p*10;
  x←x div baza;
■
scrie aux;
scrie "in baza";
scrie baza;
scrie "=";
scrie n;
stop
	#include<iostream>
	#include<math.h>
	using namespace std;
	int main()
	{int x, baza, aux, n, p;
	cout<<"x=";cin>>x; aux=x;
	cout<<"baza=";cin>>baza;
	n=0;
	p=1;
	while(x!=0)
	{
	n=n+p*(x%baza);
	p=p*10;
	x=x/baza;
	}
	cout<<aux<<" in baza "<<baza<<" = "<<n;
	return 0;}

Probleme propuse

1. Scrieţi un program în C++ în care se citeşte un număr pozitiv. Să se afişeze produsul cifrelor diferite de zero.

2. Scrieţi un program în C++ în care se citeşte un număr pozitiv. Să se afişeze numărul de cifre al numărului.

3. Scrieţi un program în C++ în care se citeşte un număr pozitiv de maxim nouă cifre. Să se afişeze cifrele numărului pe linii diferite, începând cu cea mai puţin semnificativă.

4. Scrieţi un program în C++ în care se citeşte un număr pozitiv. Să se afişeze cifra cea mai mare a numărului citit. Dacă a=287 se afişează 8.

5. Scrieţi un program în C++ în care se citesc două numere întregi a, b şi se afişează suma şi produsul numerelor din intervalul (a, b).

6. Scrieţi un program în C++ în care se citeşte un număr întreg x şi o cifră. Să se afişeze de câte ori apare cifra respectivă în numărul x citit.

7. Scrieţi un program în C++ în care se citeşte un număr întreg pozitiv n şi se calculează:

a) S=1*4+2*5+3*6+4*7+...+(n-3)*n
b) S=1+1*2+1*2*3+…+1*2*3*…*n
c) S=13+23+33+…+$\overline{n3}$
d) S=31+32+33+34+…+$\overline{3n}$

8. Scrieţi un program în C++ în care se citeşte un număr pozitiv şi se afişează dacă este palindrom sau nu. Un palindrom este un număr care este egal cu răsturnatul lui. De exemplu 121 este palindrom, iar 123 nu este palindrom.

9. * Scrieţi un program în C++ în care se citesc două numere întregi a, b şi se afişează toate numerele palindrom din intervalul (a, b).

10. * Scrieţi un program în C++ în care se citeşte un număr pozitiv şi se afişează dacă este posibil ca numărul să fie format din suma a unor numere consecutive.

11. * Să se calculeze cel mai mare divizor comun dintre trei numere întregi citite de la tastatură.

12. * Să se afişeze suma numerelor naturale prime până la n.

13. * Să se afişeze suma primelor n numere naturale prime.

14. Se citeşte un număr natural pozitiv. Să se afişeze dacă este superprim. Un număr este superprim dacă toate prefixele sale sunt tot numere prime.

15. * Să se afişeze media aritmetică a numerelor prime din intervalul [a, b] unde a şi b sunt numere întregi.

16. Scrieţi un program în C++ în care se citeşte un număr întreg pozitiv şi se afişează suma divizorilor pozitivi.

17. Scrieţi un program în C++ în care se citeşte un număr întreg pozitiv şi se afişează numărul divizorilor pozitivi ai lui.

18. Scrieţi un program în C++ în care se citeşte un număr în baza 10 şi se transformă în baza 2.

Capitolul 5.

Fişiere text

Noțiuni teoretice

Un fişier este o colecţie de date de acelaşi tip ce are **un nume** şi **o extensie**. În C++ programele ce conţin cod C++ sunt salvate ca fişiere ce au extensia .cpp. De exemplu dacă vrem să salvăm un program scris în C++ în care am verificat dacă un număr este prim sau nu, îl putem salva cu numele *prim.cpp*. Avantajul salvării unui program este că îl putem citi şi altă dată pentru a-l îmbunătăţi sau folosi.

În C++ există două tipuri de fişiere: fişiere text şi fişiere binare. Un fişier text poate conţine caractere ale codului ASCII (cifre, litere, caractere speciale), iar un fişier binar poate conţine în plus şi imagini.

Până acum am citit datele de intrare de la tastatură cu cin>> şi le-am afişat pe ecran cu cout<<, dar avem posibilitatea să citim şi să afişăm datele într-un fişier. Un fişier text are extensia .txt. În C++ putem fişierele de unde citim datele le notăm de obicei cu date.in, iar cele unde scriem rezultatele cu date.out.

Pentru a lucra cu fişiere trebuie să adăugăm **biblioteca <fstream.h>.** După ce am scris biblioteca, executăm paşii: deschidere fişier, citire date, prelucrare date, scriere sau adăugare în fişier, închidere fişier.

Citirea din fişier se realizează parcurgând următorii paşi:

- Se creează legătura dintre o variabilă pe care o utilizăm în interiorul programului, de exemplu *f* şi fişierul extern, de exemplu *date.in*

ifstream f(“date.in”);

Dacă fişierul nu se află în acelaşi folder cu programul C++, atunci trebuie să scriem toată calea. De exemplu, dacă fişierul se află în drive C, atunci scriem isfream f(“C:\\date.in”);

- Se citeşte din fişier. Dacă am numit fişierul f, atunci citirea se face cu f>>, de exemplu citim din fişier într-o variabilă x cu *f>>x*.

- Se închide fişierul cu f.close().

Problemă rezolvată

Scrieţi un program în C++ în care se citeşte din fişierul *date.in* un număr întreg şi se afişează pe ecran.

#include <fstream>
#include<iostream>
using namespace std;
int main()
{int x;
ifstream f("C:\\date.in");
f>>x;
cout<<x;
f.close();
return 0;}
	biblioteca fstream unde găsim: ifstream şi close()
	biblioteca unde găsim cout<<
	
	legătura dintre fişierul f şi suportul extern C:\\date.in
	citirea din fişierul f în variabila x
	afişarea pe ecran a conţinutului variabilei x
	închiderea fişierului

Scrierea în fişier se realizează parcurgând următorii paşi:

- Se creează legătura dintre o variabilă pe care o utilizăm în interiorul programului, de exemplu *g* şi fişierul extern, de exemplu *date.out*

ofstream g("date.out");

- Se scrie în fişier. Dacă am numit fişierul g, atunci scrierea se face cu g<<, de exemplu scriem în fişier conţinutul variabilei x cu g<<x.

- Se închide fişierul cu g.close().

Problemă rezolvată

Scrieţi un program în C++ în care se citeşte din fişierul *date.in* un număr întreg şi se afişează în fişierul date.out pătratul lui.

#include <fstream>
using namespace std;
int main()
{int x;
ifstream f("C:\\date.in");
ofstream g("C:\\date.out");
f>>x;
g<<x*x;
f.close();
g.close();
return 0;}
	biblioteca fstream unde găsim: ifstream, ofstream şi close()
	
	legătura dintre fişierul g şi suportul extern C:\\date.out
	
	scrierea în fişierul g a rezultatului x*x

Adăugarea în fişier se realizează parcurgând următorii paşi:

- Se creează legătura dintre o variabilă pe care o utilizăm în interiorul programului, de exemplu *h* şi fişierul extern, de exemplu *date.out*.

ofstream h("date.app");

- Se adaugă după datele deja existente în fişier. Dacă am numit fişierul *h*, atunci adăugarea se face cu h<<, de exemplu adăugăm în fişier conţinutul variabilei x cu h<<x.

- Se închide fişierul cu h.close().

Problemă rezolvată

Scrieţi un program în C++ în care se citeşte din fişierul *date.in* un caracter şi se adaugă în fişierul date.out. Dacă în date.in aveam litera a, în date.out aveam 2, atunci în date.out vom avea 2a.

#include <fstream>
using namespace std;
int main()
{char x;
ifstream f("C:\\date.in", ios::in);
f>>x;
ofstream g("C:\\date.out", ios::app);
g<<x;
f.close();g.close();
return 0;
}
	legătura dintre fişierul f şi suportul extern C:\\date.out, cu permisiunea de a citi din fişier ios::app ne permite să adăugăm date după cele existente

Parcurgerea unui fişier.

Dacă într-un fişier avem mai multe date, dar nu ştim câte şi vrem să le parcurgem pe toate, utilizăm while().

Problemă rezolvată

Scrieţi un program în C++ în care se citesc din fişierul date.in toate numerele întregi şi se afişează în fişierul date.out media aritmetică a lor.

```
#include <fstream>
using namespace std;
int main()
{int x, s, nr;
  ifstream f("C:\\date.in");
  ofstream g("C:\\date.out");
  s=0, nr=0;
  while(f>>x)
    { s=s+x; nr=nr+1;}
  g<<(float)s/nr;
  f.close();g.close();
  return 0;}
```

Probleme propuse

1. Scrieţi un program în C++ în care citim din fişierul *date.in* un număr întreg şi afişăm pe ecran cifrele pare ale numărului.

2. Scrieţi un program în C++ în care citim din fişierul *date.in* un număr întreg, verificăm dacă cifrele numărului sunt egale sau diferite şi afişăm un mesaj în fişierul *date.out*.

3. Scrieţi un program în C++ în care citim din fişierul *date.in* numere întregi şi afişăm în fişierul *date.out* pe linii diferite numerele citite şi în dreptul lor cuvântul par sau impar.

4. Scrieţi un program în C++ în care citim din fişierul *date.in* de la mai multe persoane: un cod şi anul naşterii şi afişăm în fişierul *date.out* pe linii diferite codul corespunzător persoanelor minore, adică al celor mai mici de 18 ani.

5. Scrieţi un program în C++ în care citim din fişierul *date.in* două numere întregi x şi y şi afişăm în fişierul *date.out* x urmat de y zerouri.

6. Scrieţi un program în C++ în care citim din fişierul *date.in* trei numere întregi de trei cifre şi afişăm în fişierul *date.out* pe linii diferite, cele trei numere formate cu unităţile, zecile, respectiv sutele numerelor iniţiale. Dacă se citesc 231, 574, 962 se afişează 142, 376, 259.

7. Scrieţi un program în C++ în care citim din fişierul *date.in* un număr întreg x şi afişăm în fişierul *date.out* de câte ori apare fiecare cifră a lui.

8. * Scrieţi un program în C++ în care citim din fişierul *date.in* un număr întreg x şi afişăm în fişierul *date.out* câte cifre distincte conţine.

9. * Scrieţi un program în C++ în care citim din fişierul *date.in* numitorul şi numărătorul unei fracţii şi afişăm în fişierul *date.out* fracţia simplificată sau un mesaj în cazul în care nu se simplifică. Dacă se citeşte 6, 12 se afişează ½.

10. Scrieţi un program în C++ în care citim din fişierul *date.in* un număr întreg x şi se afişăm în fişierul *date.out* în următoarele moduri. Dacă x =4 se va afişa ca mai jos:

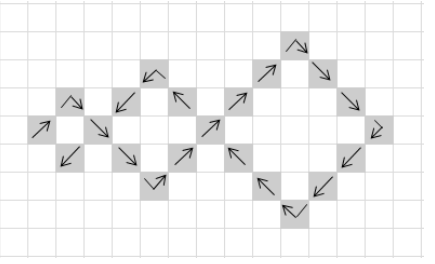
a)	b)	c)	d)
* * * *	a	1 2 3 4	1
* * * *	b b	1 2 3	2 2
* * * *	c c c	1 2	3 3 3
* * * *	d d d d	1	4 4 4 4

11. Scrieţi un program în C++ în care citim din fişierul *date.in* de pe prima linie două numere reale x şi y ce reprezintă capetele unui interval deschis şi de pe a doua linie mai multe alte numere. Să se afişeze în fişierul *date.out* câte din aceste numere se află în intervalul (x, y).

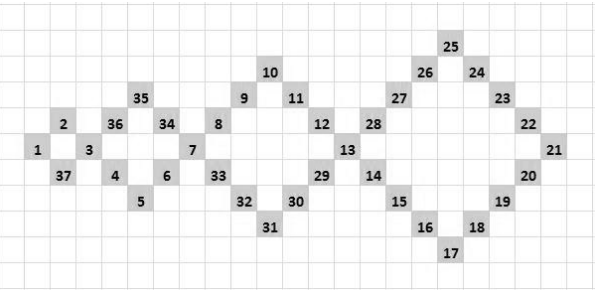
12. Scrieţi un program în C++ în care se citesc n numere întregi din fişierul *date.in* şi se afişează în fişierul *date.out* câte din aceste numere au cifrele consecutive. De exemplu, dacă se citeşte n=3 şi numerele 345, 418, 6789 se afişează 2.

13. Scrieţi un program în C++ în care se citesc 2 numere întregi din fişierul *date.in* şi se afişează în fişierul *date.out* toate numerele prime din acel interval.

14. ** *Problema covor:*



Bunica Marei țese un covor. Mara urmărește cu mare atenție modelul și încearcă să-l reconstituie pe caietul de matematică. Modelul este format din romburi. Primul romb, de indice 1, are latura formată din două pătrățele, al doilea romb, de indice 2, are latura formată din trei pătrățele etc. Un romb de indice i are latura formată din i+1 pătrățele.
Romburile sunt unite, consecutiv, ca în exemplul din imaginea alăturată. Săgețile indică sensul în care bunica țese covorul.



Ca să nu uite modelul, Mara scrie pe caiet, începând cu 1, numere consecutive care să indice modul în care țese bunica covorul.
În exemplul următor este reprezentat modul în care se țese un model format din patru romburi.

Cerinţe:
Cunoscându-se numerele n și k să se determine:
1. numărul maxim de romburi complete care pot forma modelul unui covor, descris cu ajutorul unui șir format din maximum n numere naturale consecutive (primul număr din șir fiind 1);
2. cel mai mic indice al unui romb ce conține numărul k.
Date de intrare: Fișierul de intrare covor.in conține pe prima linie, separate prin spațiu, două numere naturale: n (reprezentând numărul maxim de numere consecutive utilizate la descrierea unui model) și k (reprezentând un număr din șirul celor n numere consecutive). Linia a doua conține una dintre valorile 1 sau 2

reprezentând cerinţa 1, dacă se cere determinarea numărului maxim de romburi complete care pot forma modelul unui covor descris cu ajutorul unui șir format din maximum n numere, respectiv cerinţa 2, dacă se cere determinarea celui mai mic indice al unui romb ce conține numărul k.

Date de ieşire: Fişierul de ieşire covor.out conţine pe prima linie o valoarea naturală reprezentând numărul maxim de romburi complete care pot forma modelul unui covor, descris cu ajutorul unui șir format din maximum n numere, dacă cerinţa a fost 1, respectiv un număr natural reprezentând cel mai mic indice al unui romb ce conține numărul k, dacă cerinţa a fost 2.

Restricţii și precizări:

- $4 \leq n,k \leq 999999999$; $1 \leq k \leq n$
- Dacă numărul k nu se află pe niciunul dintre romburile complete ce pot fi construite folosind maximum n numere, atunci răspunsul de la cerința 2 este 0.
- Pentru rezolvarea corectă a cerinţei 1 se acordă 30% din punctaj, iar pentru rezolvarea corectă a cerinţei 2 se acordă 70% din punctaj.

Exemple:

covor.in	**covor.out**	**Explicaţii**
40 32 1	4	Cel mai mare număr de romburi ce pot forma un model descris cu maximum 40 de numere este 4.
40 32 2	3	Numărul 32 se află pe cel de-al treilea romb.
37 7 2	2	Numărul 7 se află pe cel de-al doilea și pe cel de-al treilea romb. Cel mai mic indice al unui romb ce conține numărul 7 este 2.
14 12 2	0	Numărul 12 nu se află pe niciunul dintre cele două romburi ce pot forma un model descris cu maximum 14 de numere.

(OJI 2015, clasa a VI-a)

BIBLIOGRAFIE

1. Clara Ionescu, Gabriela Bălan, Mihaela Giurgea, Claudiu Soroiu – *Informatică pentru grupele de performanță*, Editura Dacia Educațional, Cluj-Napoca, 2004
2. Dan Pracsiu – *Culegere de probleme semnificative de informatică*, Editura Media Sind, Vaslui, martie 2015
3. Dana Lica, Mircea Pașoi – *Informatică. Fundamentele programării*, Editura L&S Soft, 2005
4. Donald E. Knuth – *Arta programării calculatoarelor, vol. 4, fascicola 2, Generarea tuturor tuplurilor și permutărilor*, Editura Teora SRL, București, 2005
5. Emanuela Cherchez, Marinel Șerban – *Programarea în limbajul C/C++ pentru liceu*, Editura Polirom, Iași, 2005
6. Nicolae Cecilian Istrate, Dumitru Fanache, Marius Duță – *Informatica. Manual pentru clasa a X-a*, Editura Gimnasium, Târgoviște, 2000
7. Nicolae Cecilian Istrate, Luminița Duță, Adriana Alexandru, Gabriel Gorghiu – *Programarea calculatoarelor în limbajul C++,* Editura Cetatea de Scaun, Târgoviște, 2008
8. Nicolae Cecilian Istrate și colaboratori, Coordonatori Giorgie Vlad, Ovidiu Marcu – *Informatică. Modele de rezolvare. Bacalaureat 2008*, Editura Gil, Zalău, 2008
9. Răzvan Andonie, Ilie Gârbacea – *Algoritmi fundamentali. O perspectivă C++*, Editura Libris, Cluj, 1995
10. Thomas H. Cormen, Charles E. Leiserson, Ronald R. Rivest – *Introducere în algoritmi*, Editura Computer Libris Agora, Cluj Napoca, 2000
11. Tudor Sorin – *Tehnici de programare, manual pentru clasa a X-a*, Editura L&S Infomat, 1996
12. Site Lista lui Frâncu, http://probleme.francu.com/rom/home.html , accesat la 22 iunie 2015
13. Site Campion-pregătire olimpici, http://campion.edu.ro/arhiva, accesat la 22 iunie 2015
14. Site Infoarena-pregătire olimpici, http://infoarena.ro, accesat la 22 iunie 2015
15. Site Gazeta de informatică, http://gazeta.info.ro/ , accesat la 22 iunie 2015