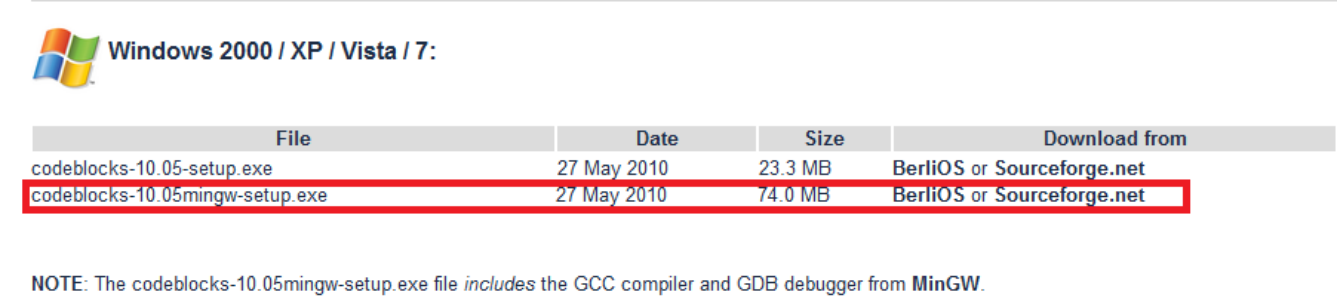
Aplicația CodeBlocks

DESCĂRCAREA ȘI INSTALAREA APLICAȚIEI.

Aplicația se descarcă de pe site-ul: http://www.codeblocks.org/downloads/26

De aici alegem

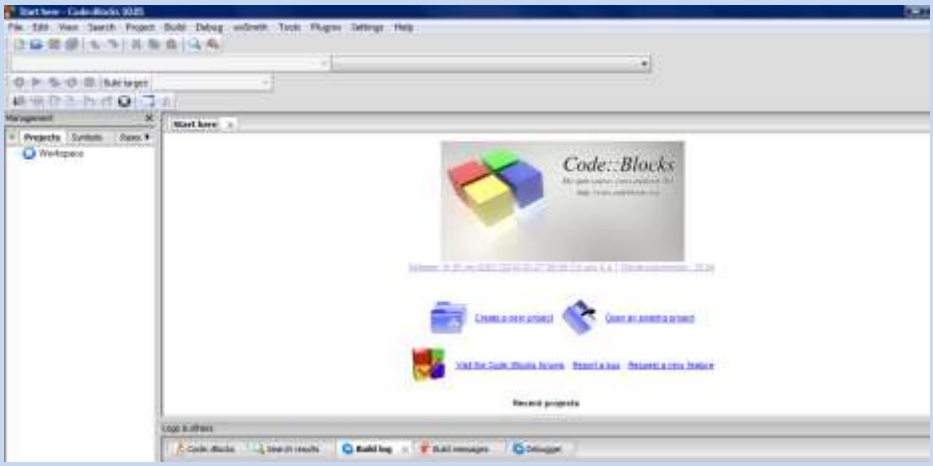


Windows 2000 / XP / Vista / 7:

File	Date	Size	Download from
codeblocks-10.05-setup.exe	27 May 2010	23.3 MB	**BerliOS** or **Sourceforge.net**
codeblocks-10.05mingw-setup.exe	27 May 2010	74.0 MB	**BerliOS** or **Sourceforge.net**

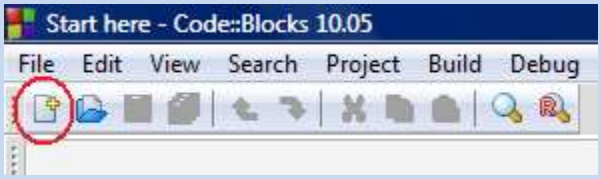
NOTE: The codeblocks-10.05mingw-setup.exe file *includes* the GCC compiler and GDB debugger from **MinGW**.

și *dăm click pe* ***BerliOS*** *sau* ***Sourceforge.net*** și va porni descărcarea.
Instalarea se realizează normal și la prima utilizare, ferestrele mici care apar se închid.

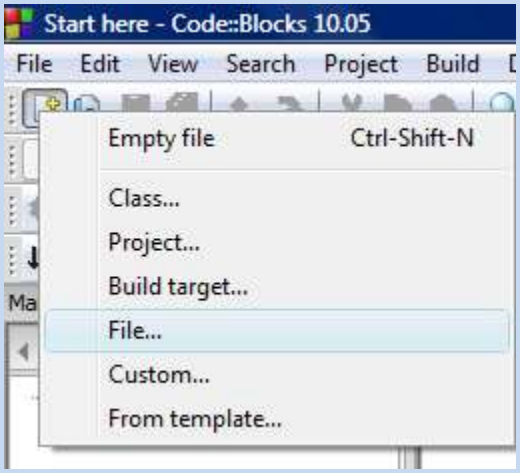


CREAREA PROGRAMELOR

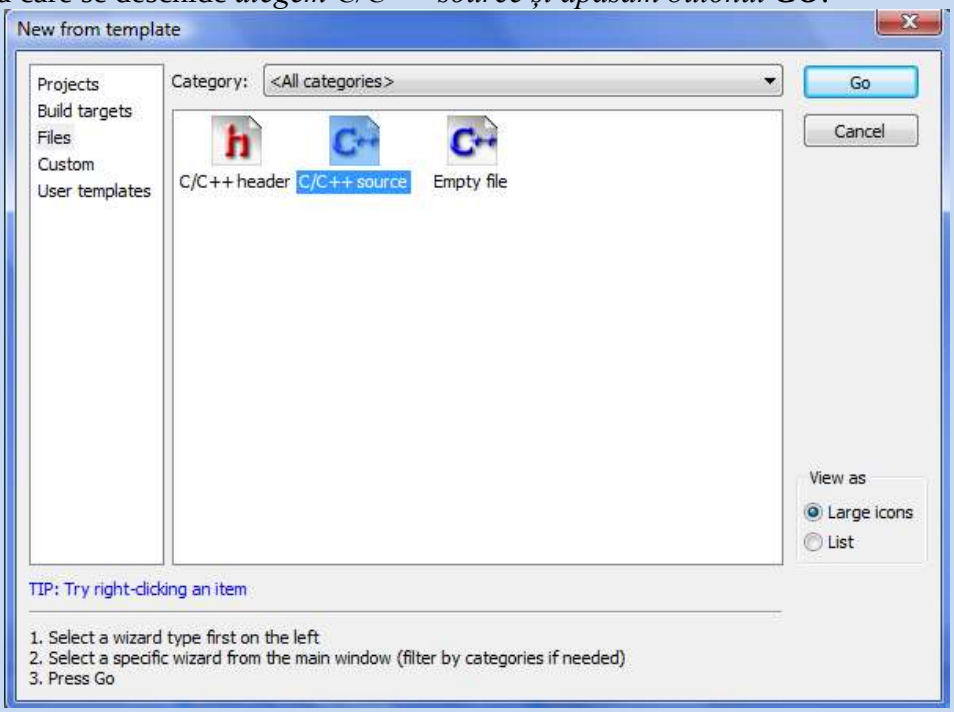
Pentru crearea unui program : ***Meniul File → New → File***
sau



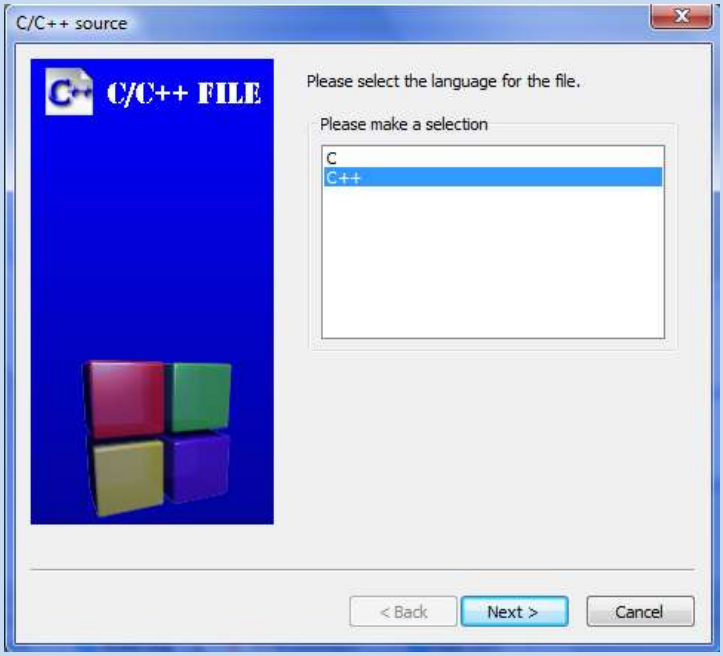
alegem ***File***



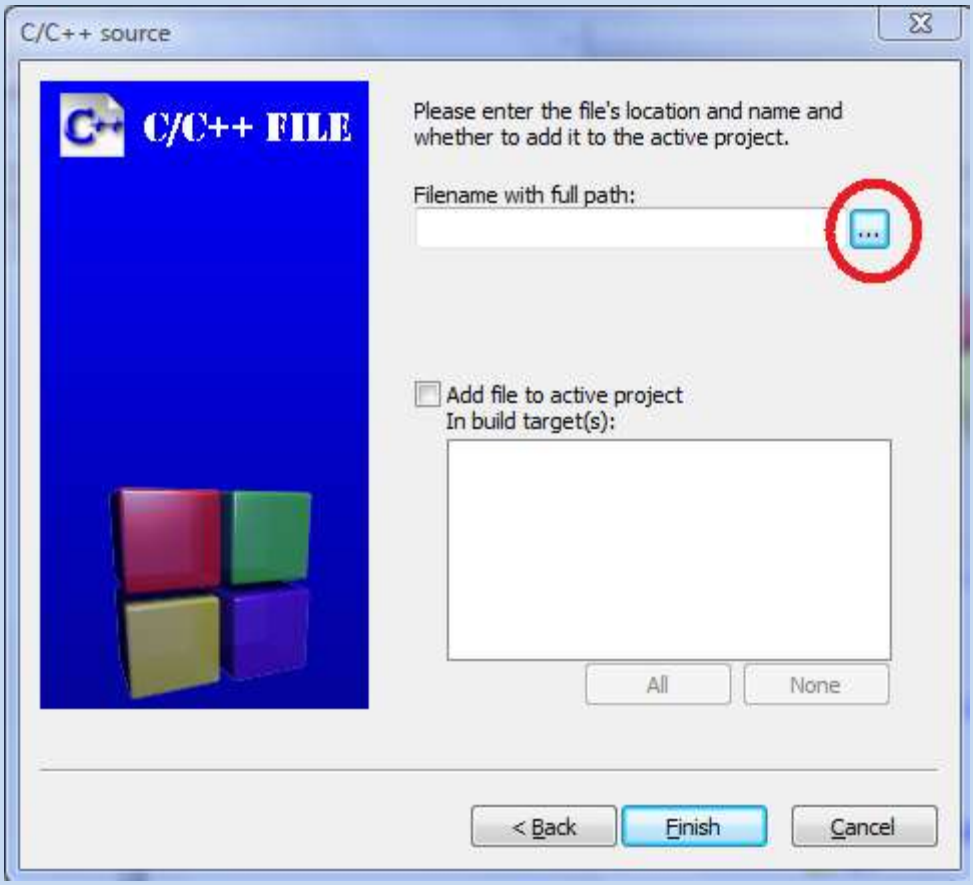
Apoi din fereastra care se deschide *alegem C/C++ source și apăsăm butonul* ***GO***:



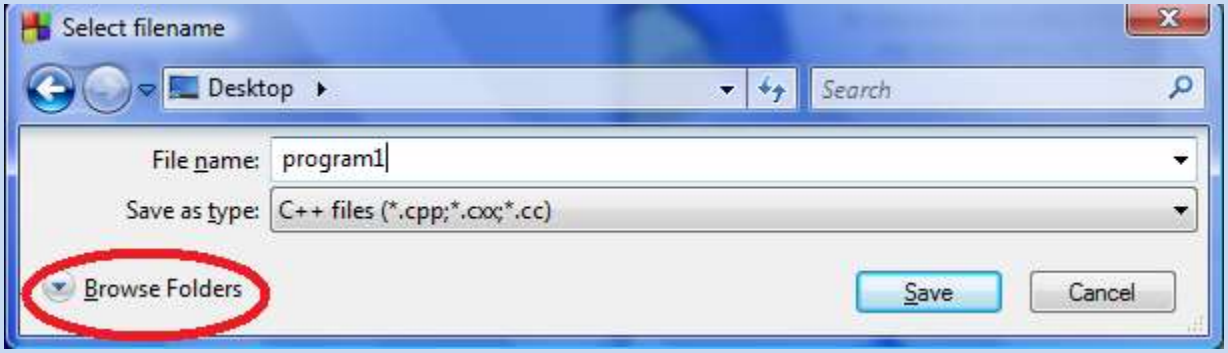
Alegem C++ și dăm ***Next.***



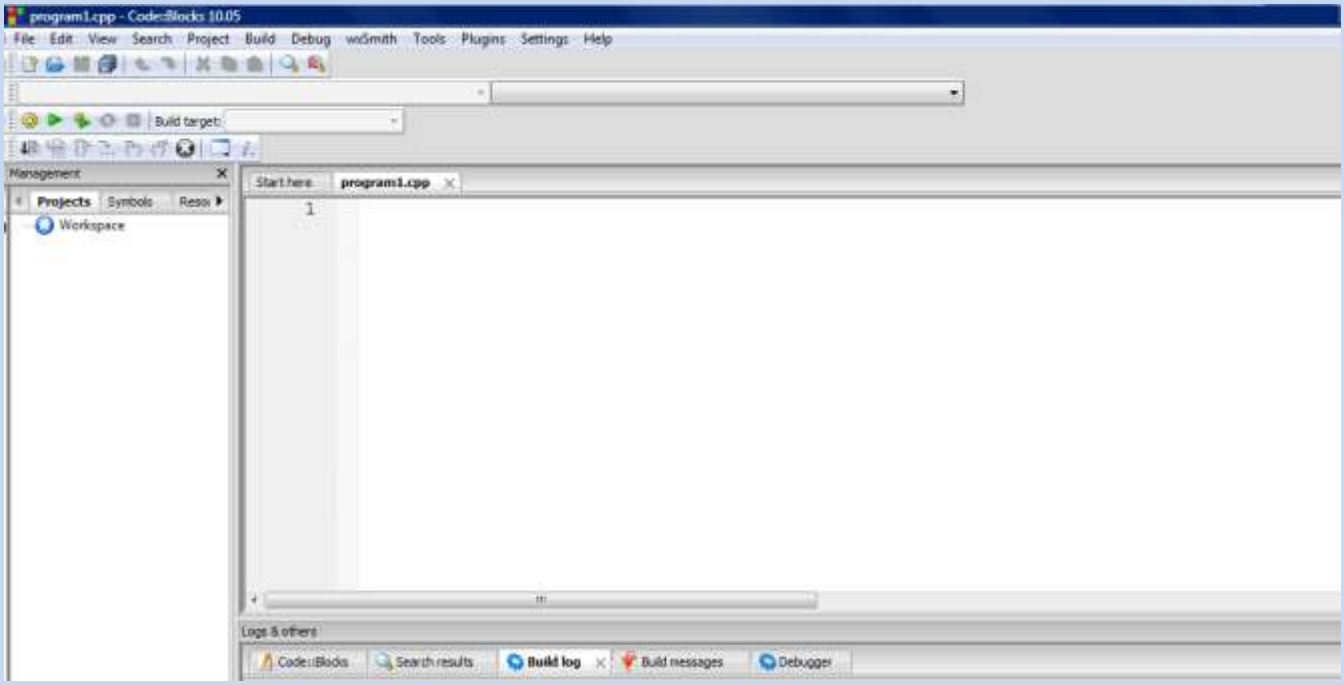
Aici trebuie să căutăm dosarul (folder-ul) în care vom salva programul. *Apăsăm pe butonul cu 3 puncte*.



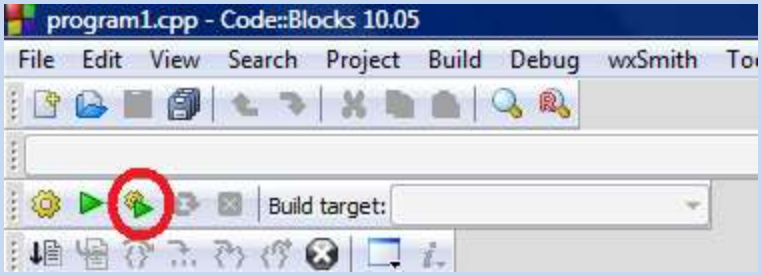
Pentru a căuta dosarul dorit *dăm click pe butorul cu săgeată* ***Browse Folders***. Apoi *scriem numele fișierului C++* și apăsăm butonul ***Save***.



În fereastra deschisă anterior apare numele programul și calea. *Dăm click pe butonul* ***Finish***.



Pentru a executa programul scris dăm click pe 



Elemente de bază ale limbajului C++. Tipuri de date

1.1. Structura programelor C++
1.2. Tipuri de date numerice şi nenumerice (exceptând tipul real)
1.3. Expresii
1.4. Operatori C++
1.5. Citiri. Scrieri

1.1 Structura progamelor C++

Vocabularul limbajului C

Setul de caractere

La scrierea programelor se foloseşte setul de caractere al codului ASCII. Mulţimea caracterelor se împarte în trei grupe:

- caractere negrafice: cod ASCII < 32 şi DEL(cod 127 - excepţie)
  - au diferite funcţii speciale, spre exemplu:
    \n - rând nou

\t - TAB
\b - backspace
\v - TAB vertical
\f - salt de pagină etc.

- spaţiu cod ASCII = 32
- caractere grafice cod ASCII > 32
  - literele mari ale alfabetului englez (A cod ASCII 65, B cod ASCII 66, ...)
  - literele mici ale alfabetului englez (a cod ASCII 97, b cod ASCII 98, ...)
  - cifre 0..9 (0 cod ASCII 48, 1 cod ASCII 49, ...)
  - caractere speciale: !, ", *. + etc.

Identificatori

Identificator = este o succesiune de litere, eventual cifre care începe cu o literă. În calitate de litere se folosesc:
literele mici şi mari ale alfabetului englez şi caracterul subliniere.

ATENŢIE!!! In C++ se face distincţie între literele mici şi cele mari !!
Max ≠ max, ordonat ≠ ORDONAT
Exemplu: Max, _min, distincte, frecv, Max2 - reprezintă identificatori corecţi
2min, 67prim, 7_org, A+B, a&, - nu reprezintă identificatori

Separatori

Pot fi:

- **space, TAB**
- **;** pentru a separa instrucțiunile
- **,** separă elementele unei liste

Cuvinte rezervate

Cuvintele rezervate = sunt identificatori cu semnificaţie fixată, care nu pot fi folosiţi în alt context decât cel precizat
în definirea limbajului.
Exemplu: if, while, do, printf, int, main, void etc.

Comentarii

Comentariile = note explicative (comentarii) ataşate unor secvenţe de operaţii, care nu au rol activ în derularea
programului.
Comentariile

- pot fi scrise pe un singur rând şi sunt precedate de caracterele //
- pot fi scrise pe mai multe rânduri şi sunt cuprinse intre caracterele /* */

Exemplu: // acesta este un comentariu pe un rând

```
/* acesta este
   un comentariu
   pe mai multe randuri  */
```

Structura unui program C++

mediu de programare = un program care permite asistarea programatorului în toate fazele de elaborare a unui program, scris intr-un limbaj de programare (editare, depanare, compilare, execuţie). Un astfel de mediu de programare este **CodeBlocks.**

program = o succesiune de comenzi (instrucţiuni) de prelucrare a datelor, scrise intr-un limbaj de programare.

Programul este memorat într-o entitate numită fişier sursă (este un fişier text cu extensia **.cpp**).

Prelucrările dintr-un program C++ sunt grupate în funcţii. Rezolvarea unei probleme se face prin utilizarea unor *funcţii definite în limbaj* şi/sau a unor *funcţii scrise de programator*, atunci când funcţiile deja existente nu sunt suficiente.

Funcţiile pe care limbajul le pune la dispoziţia utilizatorului sunt grupate, după tipul de prelucrare oferit, în mai multe fişiere numite "biblioteci" (fişiere header). Pentru a putea utiliza în program o funcţie trebuie să se specifice la începutul programului numele bibliotecii care conţine funcţia respectivă.

Orice program C++ trebuie să conţină o funcţie numită "main" (un fel de "program principal"). Instrucţiunile conţinute de funcţia main fiind cele prelucrate atunci când programul este lansat în execuţie.

ATENŢIE!!! După includerea bibliotecilor (#include<iostream> şi altele) OBLIGATORIU TREBUIE INTRODUSĂ LINIA

using namespace std; !!

Cel mai simplu program C++:

```
#include<iostream>
using namespace std;
int main()
{
  cout<<"astazi este sambata";  //afisarea unui mesaj pe ecran
  return 0;
}
```

1.2 Tipuri de date numerice întregi in C++

Dată = orice entitate asupra căreia calculatorul poate opera.
Orice dată care apare într-un program C *trebuie declarată*, pentru a fi cunoscută la compilare.

Avem:

- Date variabile = îşi modifică valoarea în timpul execuţiei programului

Exemplu:

```
int a,b;   // se declara doua variabile de tip intreg de maxim 5 cifre
long v1,x,a1; // se declara trei variabile de tip intreg de maxim 9 cifre
int x=12;          /*declararea unei variabile x de tip intreg - este o dată
                     asupra căreia se operează care are valoarea initiala 12*/
```

- Date constante = nu își modifică valoarea în timpul execuției programului

Exemplu:

```
const int a=5, b=12; // declar doua constante intregi de maxim 5 cifre care au valorile 5
                     //  respectiv 12
const long v1=34232321; //declar o constanta de tip intreg care are valoarea 34232321
```

Tipul de dată precizează

- *mulţimea valorilor* pe care le poate lua o dată,
- *operaţiile* care se pot efectua asupra ei şi
- *lungimea de reprezentare internă*.

Clasificare: În limbajul C distingem două categorii de date:

- tipuri predefinite de date (standard sau fundamentale)
  - numerice (întregi şi reale)
- tipuri derivate
  - tipuri structurate
    - tablouri (şiruri de caractere)
    - înregistrare
  - pointer
  - referinţă

Pentru început ne vom ocupa numai de **tipurile intregi simple de date** (predefinite sau standard).

Tip de date	Reprezentarea internă se face pe...	Domeniu de valori
char	8 biti	-128...127
unsigned char	8 biti	0...255
int	16 biti	-32 768...32 767
unsigned int	16 biti	0...65 535
long	32 biti	-2 147 483 648...2 147 483 647
unsigned long	32 biti	0...4 294 967 205

Observaţie: Tipul char şi tipul unsigned char se folosesc pentru a defini date de tip caracter

1.3 **Expresii**

operator = este un simbol care arata ce operatii se executa asupra unor operanzi (termeni).
operand = este o constanta, o variabila, un nume de functie sau o subexpresie a carei valoare este prelucrata direct de operator sau suporta în prealabil o conversie de tip.

În C exista 45 de operatori diferiţi dispuşi pe 15 niveluri de prioritate.
În funcţie de tipul operanzilor asupra cărora se aplică, operatorii pot fi:

- aritmetici,
- relaţionali,
- binari,
- logici, etc.

Operatorii sunt împărţiţi în clase de precedenţă (sau de prioritate) care arată ordinea în care se efectuează operaţiile. În fiecare clasă de precedenţă este stabilită o regulă de asociativitate, care indică ordinea de aplicare a operatorilor din clasa respectivă: de la stânga la dreapta sau de la dreapta la stânga.

expresie = este o combinaţie de operanzi, separaţi între ei prin operatori.
Prin evaluarea unei expresii se obţine o valoare rezultat. Tipul valorii rezultat depinde de tipul operanzilor şi a operatorilor folositi.

1.4 **Operatori C++**

Operatorii aritmetici

Operatorii aritmetici binari sunt: **+, -, *, / şi % (modul = restul împărţirii întregi)**
Prioritatea operatorilor aritmetici este:

1. +, - unari (pentru semnul unei variabile sau a unei constante)
2. *, /, % binari
3. +, - binari

Regula de asociativitate este *de la stânga la dreapta* (la priorităţi egale operatorii sunt evaluaţi de la stânga la dreapta).

Operatorul de atribuire

În limbajul C se defineşte operatorul de atribuire =.
Acest operator este binar iar **sintaxa** lui este:

variabila=expresie

Efectul acestui operator este: se evaluează expresia şi variabila primeşte acea valoare.

Operatorii relaţionali

Sunt operatori binari şi expresiile în care apar sunt 0, daca relaţia e falsă, 1 daca e adevarată.
C-ul nu defineste tipul logic, asa încât: 0 este interpretat ca valoarea logică *fals*, iar orice valoare diferita de 0 este interpretată ca adevarat.
Operatorii relationali sunt:

< <= > >= (mai mic, mai mic sau egal, mai mare, mai mare sau egal)
== != (egal, respectiv diferit)

Observatie!! *A nu se confunda* condiţia a==b cu expresia a=b, ultima având valoarea adevarat pentru orice valoare a lui b diferita de 0.

Operatorii logici

Condiţiile mai complexe se obţin aplicând condiţiilor simple operatorii logici:

! negaţie logică
&& şi logic
|| sau logic

Operanzii sunt intregi, interpretati ca valori logice. Prioritatea operatorilor && şi || este mai scazută decât a celor condiţionali, a celui de negare, fiind unar, este cea mai ridică.
Operatorii tratează operanzii ca valori logice, deci orice valoarea diferită de 0 este interpretată ca adevărat, iar 0 ca fals. Aplicând unui întreg operatorul de negaţie logică, se obţine 1 dacă operandul e fals, respectiv 0 dacă operandul este adevarat.

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

1.5 **Citiri. Scrieri**

Datele pot avea următorul traseu:

sursa: tastatură → destinaţie: monitor

Fişierul <iostream.h> conţine definiţia următoarelor obiecte:

- **cin** - face parte din clasa istream ce defineşte un stream de intrare
- **cout** - face parte din clasa ostream ce defineşte un stream de ieşire

OBSERVATIE: Citirile şi scrierile se pot face folosind obiectele cin şi cout.

Sintaxa pentru citirea datelor:

cin>>nume_variabila1>>nume_variabila2>>...>>nume_variabilan;

Efectul: citirea valorilor pentru variabilele nume_variabila1, nume_variabila2, nume_variabilan.

Exemplu:

```
...
int a,b,c;
...
cin>>a; //citeste o valoare pentru variabila a
cin>>a>>b; //citeste cate o valoare pentru variabilele a,b
```

Sintaxa pentru scrierea datelor

cout<< nume_variabila1<< nume_variabila2<<...<<nume_variabilan;

Efectul: afisarea valorilor pentru variabilele nume_variabila1, nume_variabila2, nume_variabilan.

Exemplu:

```
...
int a,b,c;
...
cout<<a; //afiseaza valoarea variabilei a
cout<<a<<b; //afiseaza valoarea variabilelor a si b
```

Afisarea unor mesaje

```
cout<<”mesaj”;
```

Exemplu:

```
cout<<”Acesta este un mesaj”;
```

Construcţia endl face posibilă trecerea la o linie nouă (**end l**ine).

Exemple:

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

```
{
  char a,b,c;
  cin>>a;                    //daca tastăm 'a' se tip 'a'
  cout<<a;       //daca tastăm  '    a' se tipărește 'a'
  cin>>a>>b>>c;          //se citesc valorile pentru a, apoi b, apoi c
  cout<<a<<b<<endl<<c; //se afişează variabilele a şi b, iar pe următoarea linie c
}
```

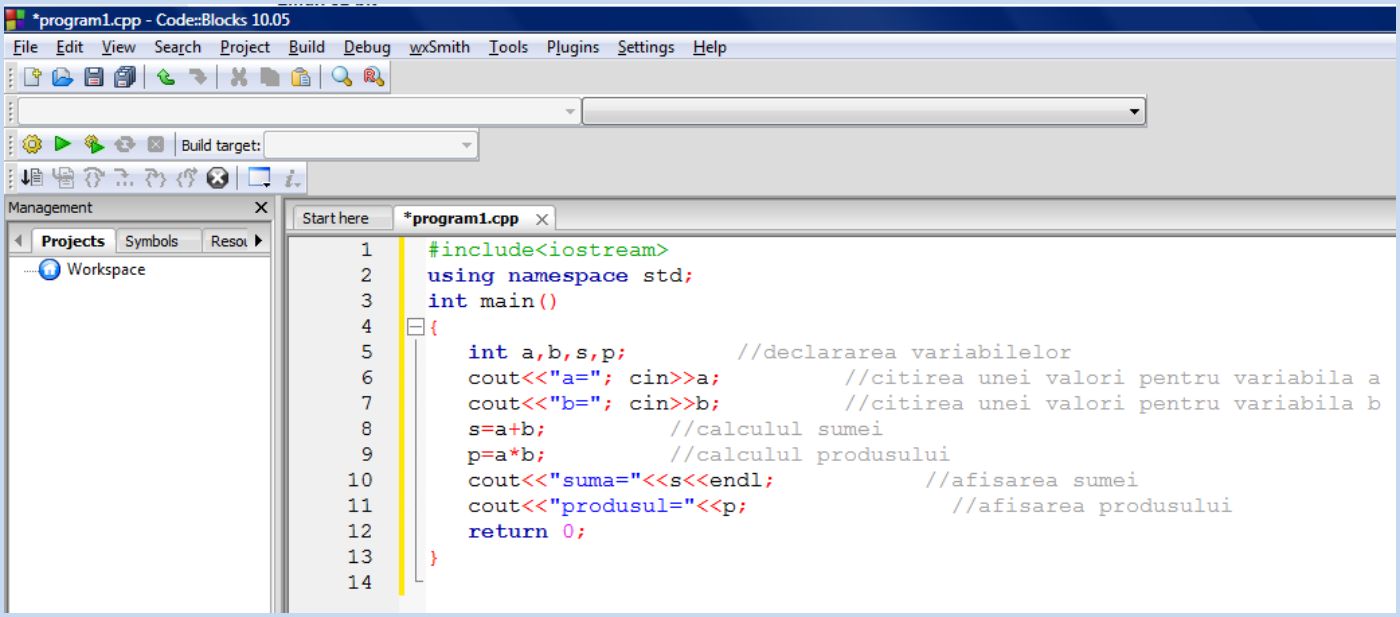
Aplicaţii rezolvate:

1. Se citesc două numere naturale a,b. Să se calculeze suma şi produsul lor.

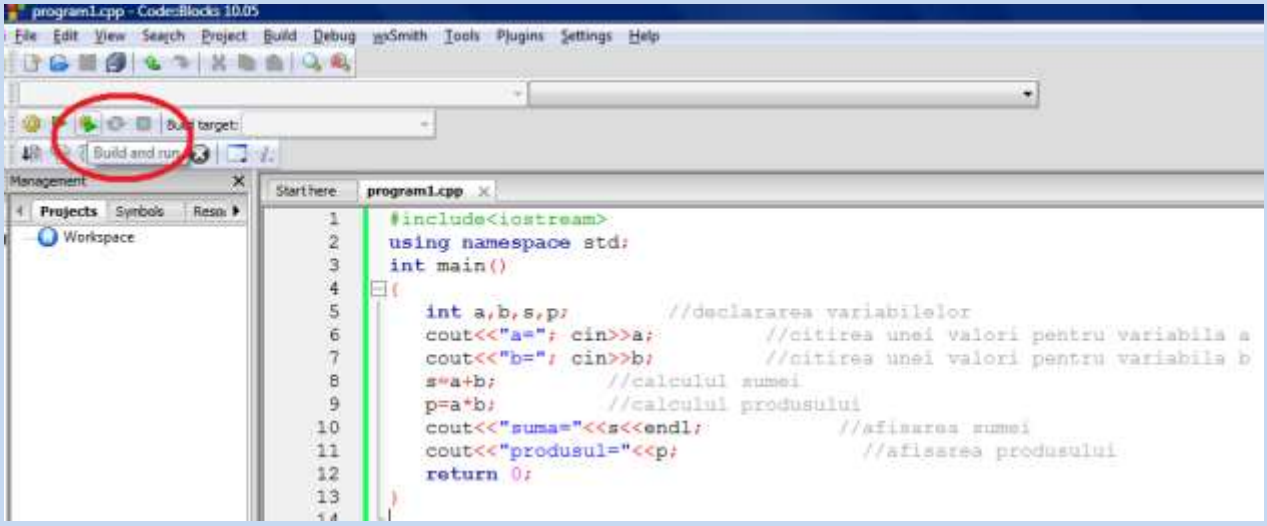
Rezolvare: programul este prezentat mai jos:

```
#include<iostream>
using namespace std;
int main()
{
  int a,b,s,p;       //declararea variabilelor
  cout<<"a="; cin>>a;        //citirea unei valori pentru variabila a
  cout<<"b="; cin>>b;        //citirea unei valori pentru variabila b
  s=a+b;        //calculul sumei
  p=a*b;        //calculul produsului
  cout<<"suma="<<s<<endl;          //afisarea sumei
  cout<<"produsul="<<p;            //afisarea produsului
  return 0;
}
```

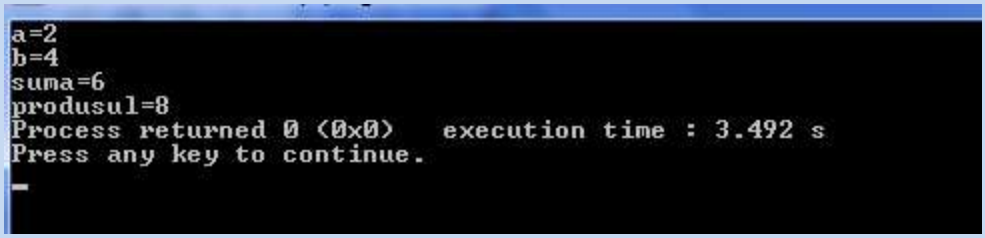
In fereastra CodeBlocks programul se va scrie ca mai jos:



Apoi rulăm programul:



Rezultatul rulării pentru a=2 şi b=4 este suma=5 si produsul=8



Observăm că după ce se afișează răspunsul apare și timpul de execuție al programului.

2. Se citeşte latura l a unui pătrat. Să se calculeze aria pătratului (A) şi perimetrul său (P).

Rezolvare: programul este prezentat mai jos:

```
#include<iostream>
using namespace std;
int main()
{
  int l,A,P;
  cout<<"l="; cin>>l;
  A=l*l;
  P=4*l;
  cout<<"aria="<<A<<endl;
  cout<<"perimetrul="<<P;
  return 0;
}
```

Exemplu de rulare, pentru latura l=4

```
l=4
aria=16
perimetrul=16
Process returned 0 (0x0)   execution time : 1.891 s
Press any key to continue.
```

Aplicaţii temă:

1. **Se citesc trei numere naturale a,b,c. Să se calculeze suma si produsul lor.**
2. **Variabilele a,b,c,d,x,y iau valori numere naturale. Să se calculeze valoarea expresiei:**
   E=a*x+b+c*y+d
3. **Se citesc cinci caractere care formează un cuvânt. Să se afişeze cuvântul precum şi cuvântul inversat (cu caracterele în ordine inversă).**
4. **Să se afişeze diverse figuri geometrice folosind caracterul *.**
   Exemplu :