**MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Informatică şi Ingineria Sistemelor**

**RAPORT**

Lucrare de laborator nr. 1

la cursul „***Programarea Calculatoarelor***”

**Tema: Calcularea expresiilor matematice. Ordinea operanzilor.**

A efectuat : **St. gr. CR-221FR Serba Cristina**

A verificat: **Asis.univ. Toma Olga**

**Chișinău 2022**

**Cuprins**

[INTRODUCERE 2](#_Toc123296517)

[REALIZAREA PRACTICĂ A SARCINII DE LUCRU 3](#_Toc123296518)

[CONCLUZII 5](#_Toc123296519)

[BIBLIOGRAFIE 6](#_Toc123296520)

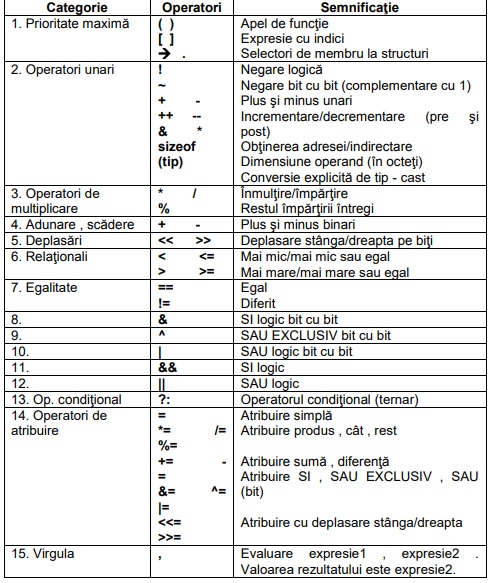
# 

# INTRODUCERE

In limbajul C/C++, expresiile sunt secvenţe de operatori şi operanzi. Orice expresie este de fapt o compunere de funcţii matematice scrisă în stil operatorial, operatorii fiind funcţii iar operanzii argumentele acestor funcţii. Operatorii pot fi unari, binari sau ternari.

Spre deosebire de ordinea evaluării operanzilor, care în anumite situaţii nu poate fi decisă, programatorul are posibilitatea să precizeze totdeauna ordinea evaluării operatorilor, prin folosirea parantezelor rotunde. Pentru a reduce numărul parantezelor necesare precizării acestei ordini, limbajul C defineşte pentru fiecare operator câte un nivel de prioritate şi câte o ordine de grupare (sau de asociere) a operatorilor cu acelaşi nivel de prioritate.

Utilizarea nivelelor de prioritate simplifică mult scrierea expresiilor şi creşte lizibilitatea lor. Totuşi, într-o expresie în care apar operatori mai rar folosiţi, când nu suntem siguri asupra nivelelor de prioritate, este recomandată utilizarea parantezelor rotunde pentru clarificarea evaluării.



**Figura 1 *Prioritatea operatorilor***

În figura 1 este reprezentată prioritatea operatorilor în cadrul unei expresii

# REALIZAREA PRACTICĂ A SARCINII DE LUCRU

**Sarcina lucrării de laborator (varianta 8):**

**Să se scrie un program care va calcula valorile expresiilor date:**

**W = ((a+b) / sqrt((pow(a, 2)+13) / (13+pow(b, 2)))) - x - y;**

**Y = (pow(x, 2-cos(y) \* sqrt(pow(a, 2) + pow(y, 2)))) / (sin(pow(x, 2) - pow(cos(y), 2)));**

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

//varianta 8

int main()

{

    float W, Y;

    int a, b, x, y;

    printf("dati variabilele a, b, x, y\n");

    scanf("%d%d%d%d", &a, &b, &x, &y);

    W = ((a+b) / sqrt((pow(a, 2)+13) / (13+pow(b, 2)))) - x - y;

    Y = (pow(x, 2-cos(y) \* sqrt(pow(a, 2) + pow(y, 2)))) / (sin(pow(x, 2) - pow(cos(y), 2)));

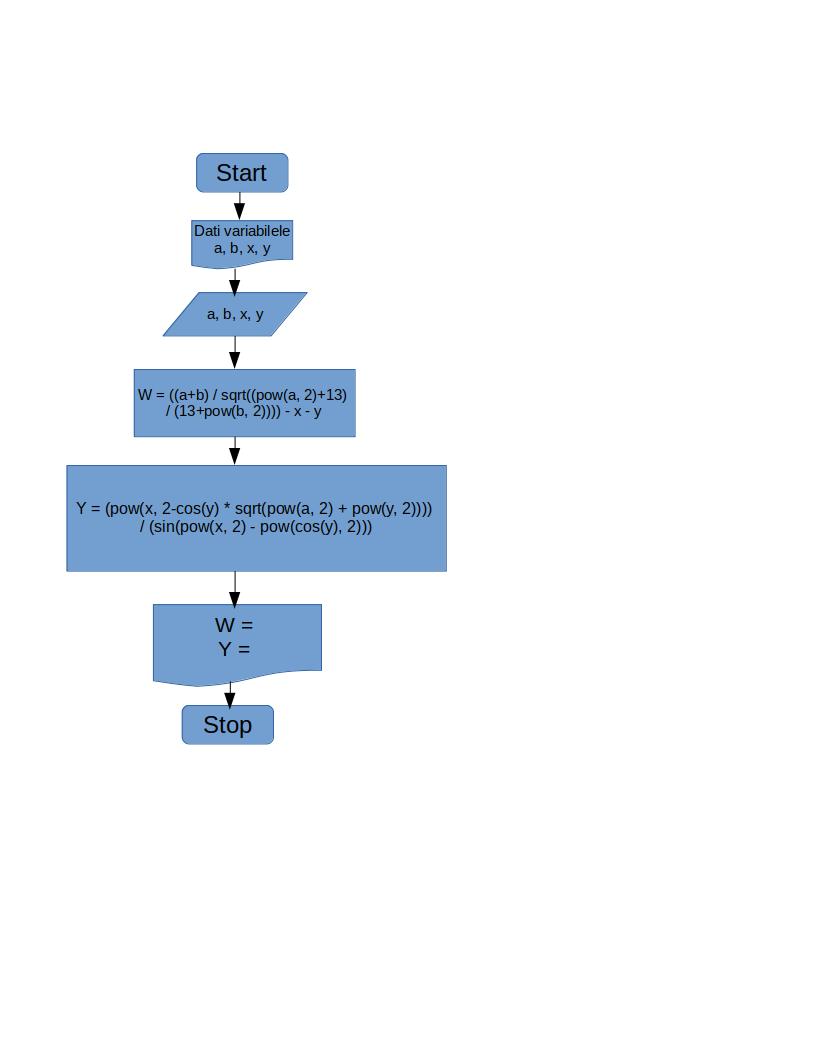
    printf("W = %f Y = %f\n", W, Y);

    return 0;

}

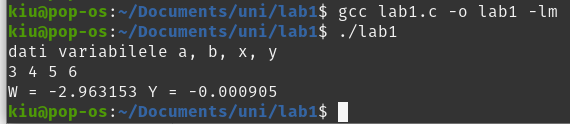
Programul calculează două valori, W și Y, pe baza valorilor variabilelor a, b, x și y. Valorile acestor variabile sunt citite de la tastatură.

În cele din urmă, acestea sunt imprimate pe consolă folosind funcția printf.



**Figura 2 *Schema bloc a programului***

Figura 2 reprezintă schema bloc a programului



# CONCLUZII

Programul dat urmărește utilizarea instrucțiunilor ciclice și de control. Prin afișarea meniului cu 3 opțiuni: if, if-else și if-else-if, utilizatorul are libertatea de a alege metoda preferată fără a se abate de la rezultat.

Deoarece cazul 2 presupune controlul a 3 intervale cu doar 2 condiții, a fost mai greu de realizat, dar posibil, datorită creării unei condiții care verifică 2 intervale în același timp. După realizarea sarcinii, am observat că o singură funcție poate fi rezolvată în mai multe moduri dacă se folosesc combinații de condiții.

Lucrarea dată, in efect, m-a ajutat să abordez subiectul utilizării funcțiilor de control dintr-o altă perspectivă.

# BIBLIOGRAFIE

[***https://www.math.uaic.ro/~necula/down\_files/cpp2017/expresii\_i\_2017.pdf***](https://www.math.uaic.ro/~necula/down_files/cpp2017/expresii_i_2017.pdf)

[***https://tuneam.files.wordpress.com/2015/02/operatori\_c-1.pdf***](https://tuneam.files.wordpress.com/2015/02/operatori_c-1.pdf)