# **Departamentul Informatică și Ingineria Sistemelor**

**Disciplina: Programarea calculatoarelor**

**RAPORT**

la lucrarea de laborator nr.**1**

**Tema:** UTILIZAREA INSTRUCȚIUNILOR DE CONTROL ȘI CICLICE ÎN LIMBAJUL C.

A efectuat :

Studentul gr: **CR-251**

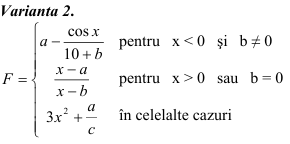
**Romaniuc Xenia**

A verificat :

Asistent universitar Tutunaru Vladina

**Scopul lucrarii**: Studierea tehnicilor şi metodelor de utilizare a instrucţiunilor de control a condiţiei şi a instrucţiunilor ciclice în limbajul C pentru tabularea funcţiei.

**Sarcina (varianta 2)**: Să se calculeze şi să se afişeze la ecran valorile argumentului **x** şi valorile funcţiei **F**, definită prin 3 expresii date, pentru intervalul **х1 ≤ x ≤ х2** şi pasul **px** de incrementare a argumentului **x**. Valorile **x1, x2, px** și parametrii **a, b, c** sunt date de intrare de tip real.



**Pasul 1** (fig. 2.1):

Includerea librăriei standard **<stdio.h>** pentru funcțiile de intrare și ieșire: **printf()**, **scanf()**; **<math.h>** pentru funcții matematice: **pow()**, **cos()**.



*Fig. 2.1. Librării*

**Pasul 2** (fig. 2.2):

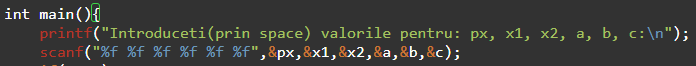
Declararea variabilelor necesare conform sarcinii; variablilele **px, x1, x2, a, b, c, x,** **F** sunt de tipul ”float”, care determină tipul de date posibil de stocat – număr real: variabila **i** de tip ”int” va stoca un număr întreg.



*Fig. 2.2. Declararea variabilelor*

**Pasul 3** (fig. 2.3):

În cursul rulării funcției principale int main(), funcția printf() afișează un mesaj pentru utilizator ”introduceti valorile pentru variabile:”. Valorile introduse prin spațiu sunt citite de funcția scanf(); specificatorul **%f** indică tipul datelor de intrare preconizat este ”float”, iar operatorul **&** determină adresa de memorie pentru fiecare variabilă specificată.



*Fig. 2.3. Citirea valorilor introduse de la tastatură*

**Pasul 4**:

Instrucțiunea condițională(structură de control) **if...else** *(Fig. 2.4)* determină conform condiției(condița: semnul pasului px) dacă x, utilizat în formulele din *Fig1.6,* va fi egal cu extremitatea superioară(x2) sau inferioară(x1) a intervalului [x1,x2].



*Fig. 2.4 Instrucțiunea condițională*

**Pasul 5**:

Instrucțiunea ciclică **while(**condiția**)** va repeta blocul de cod din acolade {…} atât timp cât condiția este adevărată. În program, condiția **x<=x2 && x>=x1** verifică dacă valoarea lui **x** aparține intervalului [x1,x2]; operatorul logic ”și” - **&&** determină că ambele condiții trebuie să fie adevărate. Operația x=x+px determină schimbarea valorii lui x până la momentul cand nu se va mai respecta condiția, astfel, oprinduse rularea buclei(blocului de cod).

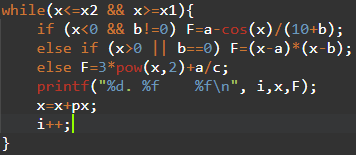
Structura de control **if...else** *(Fig. 2.5)* determină formula utilizată conform condiției, la fiecare iterație:

1. **If()**: dacă condiția x<0 && b!=0 este respectată, if() returnează 1 – adevărat, variabilei F i se va atribui valoarea obținută din ”a-cos(x)/(10+b)”; în caz contrar(if returnează 0 - fals) se trece la instrucțiunea else if().

2. **Else if**: dacă condiția x>0 || b==0 este respectată, variabilei F i se va atribui valoarea conform formulei (x-a)\*(x-b), în caz contrar se trece la intrucțiunea else.

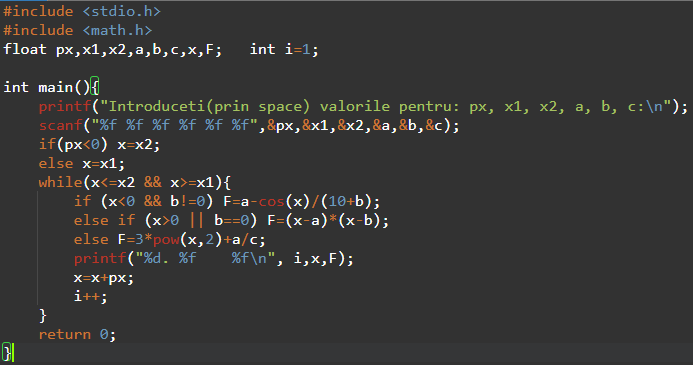
3. **Else**: nu are condiție, așadar, variabilei F îi este atribuită valoarea conform 3\*pow(x,2)+a/c.

În urma atribuirii valorii variabilei F, prin funcția printf() se afișează pe ecran valorile variabilelor *(Fig. 2.6)* x, F; variabila i, valoarea căreia se afișează prima, are rol de numerotare, ordonare *(Fig. 2.6)*.

 *Fig. 2.6. Secvență a codului executat*

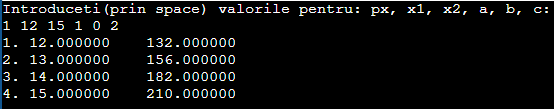
*Fig. 2.5. Instrucțiunea ciclică*

**Programul** (fig. 1.1):

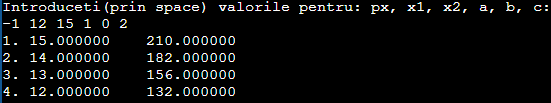


*Fig. 1.1. Programul*

**Cod executat** (fig. 1.3, fig. 1.4):

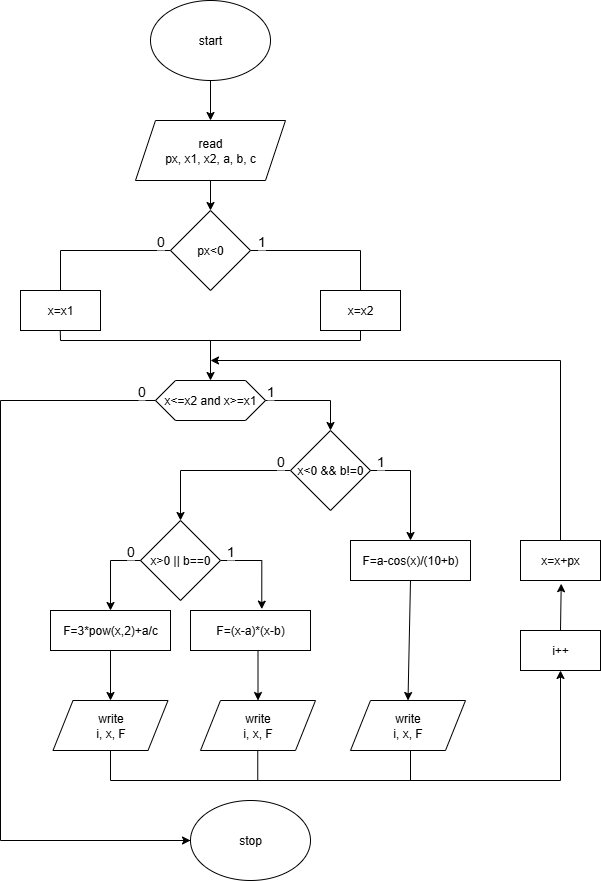
**

*Fig. 1.3. Cod executat*

**

*Fig. 1.4. Cod executat*

**Schema block** (fig. 1.2):



*Fig. 1.2. Schema block*

**Concluzia:**

În urma efectuării lucrării de laborator am studiat modul de funcționare și sintaxa instrucțiunilor de control (if, if...else) și a instrucțiunilor ciclice(for, while, do...while), aplicând în practică cunoștințele acumulate, care sunt esențiale pentru programarea în limbajul C.