Ministerul Educaţiei și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică şi Microelectronică

**RAPORT**

Lucrare de laborator Nr.2

*la Programarea Calculatoarelor*

Tema: Instrucțiuni repetitive (cicluri)

A efectuat: st. gr. SI-212 Brânză Vlad

A verificat: lect. asist. Mititelu Vitalii

Chişinău 2021

**Lucrare de laborator Nr.2**

**Tema:** Instrucțiuni repetitive (cicluri)

**Scopul:** Utilizarea instrucțiunilor repetitive (for, while și do-while) pentru scrierea programelor în limbajul de programare C.

**Sarcina:** Scrieți un program care citește de la tastatură valorile variabilelor a, b, c, n, i, j, k (în dependență de variantă), efectuează ceea ce este indicat în variantă, apoi afișează pe ecran rezultatul:

**Varianta 4:**

Calculează media aritmetică a cifrelor unui număr n.

**Rezumat:**

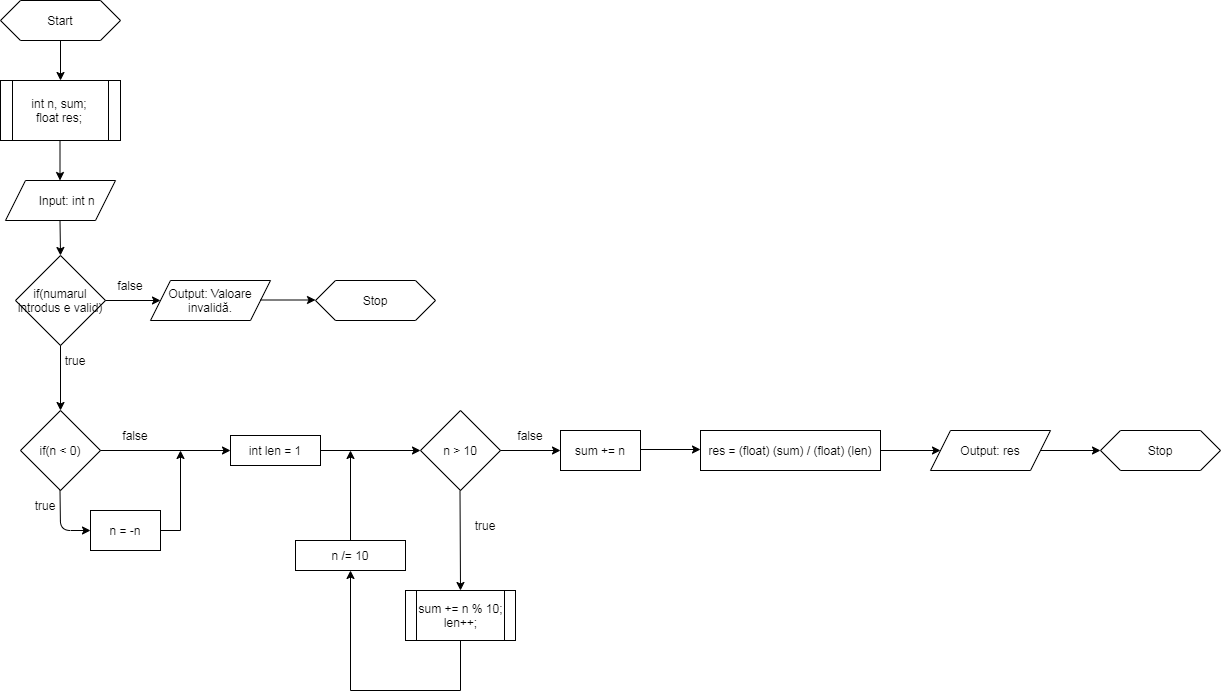
**Noţiuni principale din teorie şi metode folosite:**

Instrucţiunile de ciclare (repetitive, iterative, etc) sunt utilizate în situaţia în care o anumită secvenţă de cod trebuie executată de mai multe ori în mod consecutiv, numărul de repetiţii fiind fix, stabilit iniţial, sau variabil, depinzând de rezultatul acţiunii repetate.

Limbajul C pune la dispoziţia programatorilor trei instrucţiuni de ciclare: while, do-while şi for, toate trei utilizând pentru continuarea ciclării o condiţie de tip “cât timp este adevărat că …”, diferenţa dintre ele fiind dată, în principal, de momentul testării condiţiei de ciclare: în cazul instrucţiunii while testarea are loc înaintea fiecărei execuţii a acţiunii repetate (testare anterioară), în cazul do-while testarea se face la reluare, după fiecare execuţie (testare posterioară), iar în cazul instrucţiunii for testarea se face analog cazului while, instrucţiunea for având însă prevăzută şi evaluarea unei anumite expresii înainte de fiecare reluare, precum şi o secvenţă iniţială, executată înaintea intrării în ciclare.

Instrucţiunile repetitive păstrează controlul fluxului de execuţie al programului până când are loc ieşirea din ciclare. O ieşire din ciclare poate fi normală, atunci când controlul este preluat de următoarea instrucţiune din textul sursă în urma evaluării condiţiei de repetare a ciclării, sau poate fi forţată, prin executarea unei instrucţiuni de salt aflate în corpul instrucţiunii repetitive.

**Schema bloc a algoritmului de soluționare a problemei:**



**Codul deplin al programului:**

#include <stdio.h>

#include <math.h>

int main ()

{

    int n, sum;

    float res;

    printf("\nIntroduceți vă rog numarul n: ");

    if(!scanf("%d", &n)){

        printf("\nValoare invalidă.\n");

        return (1);

    }

    if(n < 0) {

        n = -n;

    }

    int len = 1;

    for(; n > 10; n /= 10) {

        sum += n % 10;

        len++;

    }

    sum += n;

    res = (float) (sum) / (float) (len);

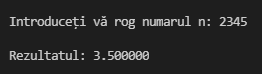
    printf("\nRezultatul: %f\n", res);

    return (0);

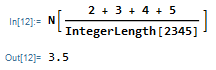
}

**Execuția programului:**

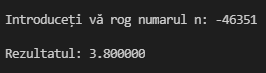
* Set de date nr. 1: 2345



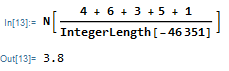
Calcule confirmative în Wolfram:



* Set de date nr. 2: -46351

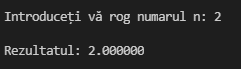


Calcule confirmative în Wolfram:

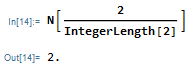


\*Notă: Minusul nu e o cifra, așa că el nu se ia în considerare la sumare.

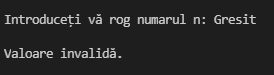
* Set de date nr. 3: 2



Calcule confirmative în Wolfram:



* Set de date nr. 4: „Gresit”



„Gresit” este de tip char \*, ceea ce duce la o eroare.

**Concluzii:**

**Surse:**

* **Instrucţiuni C\C++ (partea a II-a)**

III. Instrucţiuni de ciclare:

https://www.math.uaic.ro/~necula/down\_files/cpp2017/instructiuni\_ii\_2017.pdf