

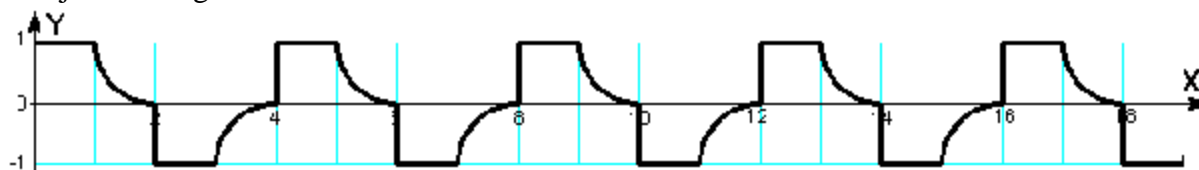
**Tema: Aplicarea instrucțiunilor repetitive imbricate**

**Obiective:**

- să definească noțiunea de instrucțiune imbricată,
- să recunoască contextul utilizării instrucțiunilor ramificare și ciclice imbricate;
- să descrie tipurile de cicluri și particularitățile acestora;
- să descrie metodele de organizare a ramificărilor și salturilor în programe;
- să elaboreze programe ciclice cu instrucțiuni imbricate.

**6.1. Sarcina pentru soluționare**

Pentru funcția  $Y = f(X)$ , graful căreia este prezentat pe Figura 6.1, de afișat la ecran valoarea lui  $Y$  pentru  $X = 0; 0,25; 0,5; \dots; 19,75$ . Cerință suplimentară – de obținut la ecran graficul funcției cu mijloacele regimului textual.



**Figura 6.1.** Graficul funcției  $Y = f(X)$

**6.2. Exemplu de soluționare a sarcinii**

**Metoda generală de elaborare a algoritmului soluționării**

Din graficul funcției prezentat în Figura 6.1 se vede că funcția este formată din cinci perioade. Astfel, programul poate să conțină un ciclu, care de 5 ori va îndeplini unele și aceleași calcule. La fiecare iterație trebuie să se ia în considerație valorile variabilei  $x$  de la 0 până la 3,75 cu pasul 0,25, cu alte cuvinte, avem un ciclu plasat. La fiecare iterație a ciclului plasat se calculează valoarea lui  $y$  pentru  $x$  curent și se extrage rezultatul.

Analiza graficului prezentat arată că fiecare perioadă este formată din patru segmente: pe segmentul  $0 \leq x < 1$  graficul este o dreaptă, pe segmentul  $1 \leq x < 2$  – arc, pe segmentul  $2 \leq x < 3$  – altă dreaptă și segmentul  $3 \leq x < 4$  – este descris iarăși de un arc. Prin urmare, în ciclul plasat trebuie să fie o ramificare, în care se calculează valoarea curentă a valorii  $x$  și se asigură calculul pentru primul, al doilea, al treilea și al patrulea caz.

*Calcularea funcției pe segmentul 0-1*

Pe acest segment, funcția este o dreaptă. Formula dreptei:  $y = ax + b$ . Pentru acest caz,  $a = 0$ ,  $b = 1$ . Astfel, formula finală pentru segmentul  $0 \leq x < 1$  este:  $y = 0 * x + 1$ .

*Calcularea funcției pe segmentul 1-2*

Pe acest segment, funcția este o parte din cerc. Formula cercului:  $(x - x_0)^2 + (y - y_0)^2 = R^2$ , unde  $(x_0, y_0)$  – coordonatele centrului cercului, iar  $R$  – raza. De aici  $y$  poate fi găsit ca:  $y = y_0 + \sqrt{R^2 - (x - x_0)^2}$ . În cazul nostru  $R = 1$ , iar coordonatele centrului –  $(2, 1)$ . La extragerea rădăcinii pătrate se obțin două valori – pozitivă și negativă, în cazul nostru se utilizează numai semicercul de jos, așa că trebuie să luăm numai valoarea negativă. Formula finală pentru segmentul  $1 \leq x < 2$  este:  $y = 2 - \sqrt{1 - (x - 2)^2}$ .

*Calcularea funcției pe segmentul 2-3*

Pentru dreapta de pe acest segment,  $a = 0$ ,  $b = -1$ , formula finală:  $y = 0 * x - 1$ .

### *Calcularea funcției pe segmentul 3-4*

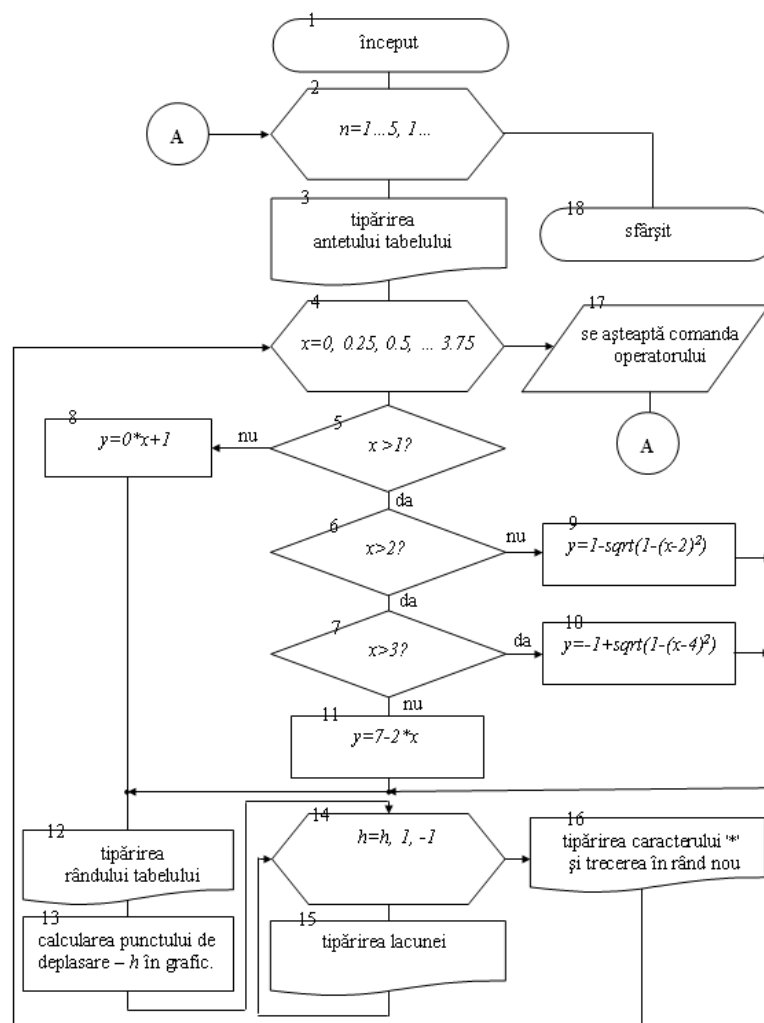
Pe acest segment, funcția este o parte din cerc. La extragerea rădăcinii pătrate, obținem formula finală pentru segmentul  $3 \leq x \leq 4$ :  $y = -1 + \sqrt{1 - (x-4)^2}$ , în cazul nostru se utilizează numai semicercul de sus, astfel luăm numai valoarea pozitivă.

### **Algoritmul de extragere a rezultatelor în forma grafică**

La fiecare iterație a ciclului interior, se obține o coordonată a graficului. Prin urmare, pentru prezentarea rezultatului în forma grafică va fi comod a desfășura graficul la  $90^\circ$  și în fiecare rând al ecranului de reprezentat valorile  $x$ ,  $y$  și simbolul, care reprezintă un punct al graficului. Deplasarea acestui simbol în rând va fi proporțională cu valoarea coordonatei. Pentru reprezentarea acestui simbol la deplasare, coordonata se transformă în număr întreg (cu scara la mărimea rândului), acest număr va fi numărul de „spații albe”, care trebuie introduse în rând înaintea simbolului – „punct”.

La fiecare perioadă, funcția include 16 iterații ale ciclului plasat, astfel, pe ecran vor fi extrase 16 rânduri. Ar fi rațional după afișarea fiecărei perioade (care integral se încadrează pe ecran) de făcut o pauză în program până la comanda utilizatorului de continuare.

Schema algoritmului este prezentată în Figura 6.2:



**Figura 6.2.** Schema bloc

### Determinarea variabilelor programului

Pentru realizarea algoritmului, se vor utiliza următoarele variabile:

$n$  – parametrul ciclului extern, pentru el este de ajuns un număr scurt întreg:

short  $n$ ;

$x$  – parametrul ciclului intern și, totodată – valoarea curentă a abscisei graficului. Cu toate că precizia nu e mare, se anunță ca *double*, în corespundere cu stilul comun de programare în C:

double  $x$ ;

$y$  – valoarea curentă a ordonatei graficului:

double  $y$ ;

$h$  – deplasarea în rând a simbolului, care înseamnă punctul graficului. Întrucât deplasarea este limitată de lățimea ecranului, pentru această variabilă va fi de ajuns un număr scurt întreg:

short  $h$ ;

### Elaborarea textului programului

Elaborarea programului începe cu antetul funcției principale:

```
int main(void)
```

În continuare, se deschide corpul funcției și se include declararea variabilelor.

Partea codată a programului începe în corespundere cu schema algoritmului (blocul 2, Figura 6.2) i – antetul ciclului extern:

```
for (n=0; n<5; n++) {
```

Corpul ciclului include instrucțiunea de tipărire a antetului de tabel (blocul 3) cu instrucțiunile:

```
printf(" |   X   |   Y   | \n");  
printf(" |-----|-----| \n");
```

Apoi se deschide ciclul intern (bloc 4). Parametrul acestui ciclu – valoarea abscisei  $x$  – ia valorile de la 0 până la 3,75 cu pasul 0,25:

```
for (x=0; x<4; x+=0.25) {
```

Blocul 5 din schema algoritmului verifică din ce segment face parte  $x$ , această acțiune se realizează cu ajutorul instrucțiunii condiționale:

```
if (x<1)
```

În cazul îndeplinirii acestei condiții, valoarea variabilei  $y$  se calculează (blocul 8) conform ecuației pentru primul segment:

```
y=0*x+1;
```

În caz contrar, blocul 6 din schema algoritmului se realizează cu instrucțiunea condițională imbricată:

```
else if (x<2)
```

În caz de îndeplinire a condiției, valoarea ordonatei se calculează (blocul 9) după condiția pentru segmentul doi:

```
y=1-sqrt(1-(x-2)*(x-2));
```

Funcția *sqrt()* este descrisă în fișierul *math.h*, care și se include la începutul programului:

```
#include <math.h>
```

Pentru segmentul trei, se execută verificarea variabilei  $x$  conform blocului 7:

```
else if (x<3)
```

În caz de adevăr, se îndeplinește condiția pentru segmentul trei (blocul 11):

```
y=7-2*x;
```

În caz contrar, are loc pătrunderea pe ultimul segment, calcularea ordonatei se realizează conform blocului 10:

```
y=-1+sqrt(1-(x-4)*(x-4));
```

În ciclul intern, se fac calcule pentru o perioadă a funcției, însă ciclul extern se va repeta de 5 ori. Valoarea reală a abscisei se va calcula astfel:  $x+4n$ . Având calculată abscisa și ordonata graficului, datele de ieșire se afișează (blocul 12). Mai întâi de toate, se extrage rândul tabelului:

```
printf(" | %5.21f | %10.71f | ", x+n*4, y);
```

Specificarea formatului – este ales în așa mod, ca valorile  $x$  și  $y$  să fie extrase în coloană. Parametrii numerici specificați *%lf* trebuie să asigure prezentarea valorilor  $x$  și  $y$  cu o exactitate suficientă. Funcția *printf()* este descrisă în fișierul *stdio.h*, de aceea noi includem acest fișier la începutul programului:

```
#include <stdio.h>
```

Următorul pas – extragerea în același rând a punctului graficului. Pe spațiul din dreapta urmează să prezentăm valorile  $y$  de la -1 până la +1. Dacă coeficientul scară ea valoarea 10, amplitudinea graficului constă din 20 locuri semn – îndeajuns pentru ilustrarea prezentării. Scalarea valorii ordonatei se îndeplinește cu instrucțiunea (valoarea aleasă pentru coeficientul de scalare  $\div 10$  – poate fi modificată la ajustare).

```
h=(y+1)*10;
```

La realizarea ultimei instrucțiuni, se pierde partea fracționară. Întrucât printre funcțiile bibliotecii limbajului C nu există funcție de rotunjire exactă, noi realizăm o astfel de rotunjire personificată:

```
if ((y+1)*10-h)>0.5) h++;
```

În continuare, în rândul ecranului se extrage (blocul 14-15) numărul necesar de lacune.

Valoarea  $h$  în ciclu funcționează ca un contor de scădere.

```
for (; h>0; h--) printf(" ");
```

se afișează simbolul '\*' și are loc trecerea la rândul nou al ecranului:

```
printf("*\n");
```

Cu aceasta se finisează ciclul intern.

După ieșirea din ciclul intern, mesajul este afișat prin instrucțiunea:

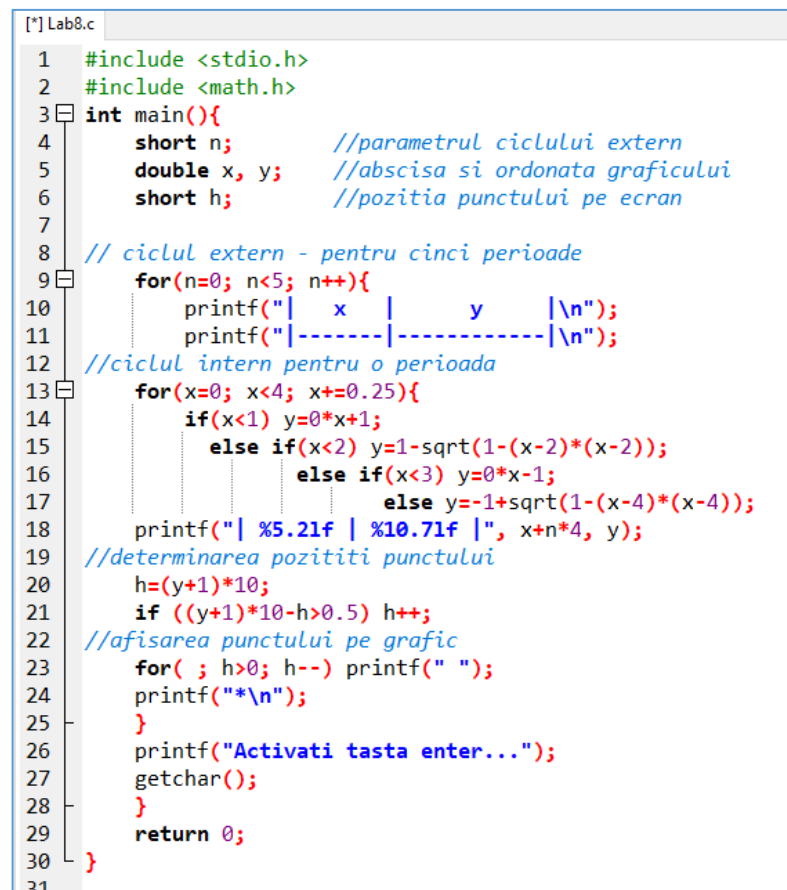
```
printf("Activați tasta Enter...");
```

Se așteaptă apăsarea unei taste:

```
getchar();
```

Cu aceasta se finisează ciclul intern, apoi funcției *main()*.

Textul integral al programului este prezentat în Figura 6.3.



```
[*] Lab8.c
1  #include <stdio.h>
2  #include <math.h>
3  int main(){
4      short n;           //parametrul ciclului extern
5      double x, y;       //abscisa si ordonata graficului
6      short h;           //pozitia punctului pe ecran
7
8      // ciclul extern - pentru cinci perioade
9      for(n=0; n<5; n++){
10         printf("    x        y        |\n");
11         printf("|-----|-----|\n");
12         //ciclul intern pentru o perioada
13         for(x=0; x<4; x+=0.25){
14             if(x<1) y=0*x+1;
15             else if(x<2) y=1-sqrt(1-(x-2)*(x-2));
16             else if(x<3) y=0*x-1;
17             else y=-1+sqrt(1-(x-4)*(x-4));
18             printf("| %5.2lf | %10.7lf |", x+n*4, y);
19             //determinarea pozitiei punctului
20             h=(y+1)*10;
21             if ((y+1)*10-h>0.5) h++;
22             //afisarea punctului pe grafic
23             for( ; h>0; h--) printf(" ");
24             printf("*\n");
25         }
26         printf("Activati tasta enter...");
27         getchar();
28     }
29     return 0;
30 }
31
```

Figura 6.3. Textul programului

### Ajustarea programului

Pentru ajustarea programului prezentat, urmează a efectua următoarele acțiuni:

- verificarea organizării corecte a ciclurilor intern și extern;
- verificarea funcționării corecte a instrucțiunilor condiționale;
- verificarea calculării corecte a valorii ordonatei pe cele patru segmente;

- de verificat afișarea corectă a graficului pentru fiecare segment.

### Rezultatele derulării programului

Sunt prezentate numai o parte din rezultatele obținute la ecran după derularea programului (pentru o perioadă a funcției). Celelalte rezultate se repetă raportate față de alte valori ale variabilei  $x$ .

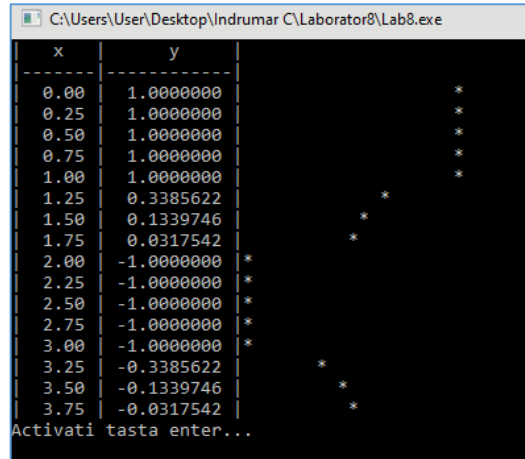


Figura 6.4. Rezultatele furnizate de program

### 6.3. Subiecte de evaluare

#### Exerciții:

1. Să se implementeze programul din exemplul prezentat.
2. Să se tabeleze funcția  $f(x)$  pe intervalul  $[x1, x2]$  cu pasul  $\Delta x$ , dacă:
  - a)  $f(x) = \ln(x+2)$ ,  $x1 = -1$ ,  $x2 = 1$ ,  $\Delta x = 0,2$ ;
  - b)  $f(x) = \begin{cases} 20 + 3x^2, & x \geq 4 \\ 16 - x/3, & x < 4 \end{cases}$ ,  $x1 = -4$ ,  $x2 = 4$ ,  $\Delta x = 0,2$ ;

#### Verificarea cunoștințelor:

1. Descrieți noțiunea de instrucțiune imbricată.
2. Identificați instrucțiunile de întrerupere a instrucțiunilor repetitive. Prezentați exemple de utilizare a acestora.
3. Enumerați cele trei acțiuni care sunt necesare să fie îndeplinite la execuția unui ciclu.
4. Caracterizați avantajul utilizării instrucțiunii *for*.
5. Ce deosebiri sunt între instrucțiunea *while* și instrucțiunea *do-while*?
6. Pornind de la sintaxa instrucțiunii *for*, stabiliți echivalența între aceasta și instrucțiunile *while* și *do-while*.

### 6.4. Bibliografie

1. Kris Jamsa & Lars Klander, *Totul despre C și C++*, Editura Teora, București, 2006.
2. Herbert Schildt, *C++ manual complet*, Editura Teora, București, 1999. Disponibil: <https://drive.google.com/file/d/1BrQtITgykcWk03xxtl3Q0-nvcRRFAsy7/view?usp=sharing>
3. Tutorials C++ Language. Operators. Disponibil: <https://cplusplus.com/doc/tutorial-ro/control/>