Funcții predefinite în C/C++ pentru tipuri numerice

Definiție: O funcție predefinită este un bloc de instrucțiuni scris de către autorii limbajului și care face parte intrisecă din limbaj; ea poate avea unul sau mai mulți parametri, ce definesc valorile cu care lucrează în timpul execuției sale. La apel se transmit funcției date pentru parametrii care au fost definiți, iar în urma execuției funcției apelate acesta întoarce o anumită valoare.

Atenție: Pentru ca funcțiile de mai jos să funcționeze, trebuie inclusă biblioteca ***cmath!***

Pentru constantele și variabilele de ***tip real***, există următoarele ***funcții predefinite***:

a. **abs(x)**: returnează modulul numărului x

```
     double abs (double x);
      float abs (float x);
long double abs (long double x);
```

b. **sqrt(x)**: returnează radical din x

```
     double sqrt (double x);
      float sqrt (float x);
long double sqrt (long double x);
```

c. **pow(x,y)**: returnează valoarea lui x la puterea y

```
     double pow (double base     , double exponent);
      float pow (float base      , float exponent);
long double pow (long double base, long double exponent);
     double pow (double base     , int exponent);
long double pow (long double base, int exponent);
```

d. **ceil(x)**: rotunjește pe x la cel mai apropiat întreg mai mare decât x și returnează rezultatul

```
     double ceil (double x);
      float ceil (float x);
long double ceil (long double x);
```

e. **floor(x)**: trunchiază pe x la cel mai apropiat număr întreg mai mic decât el și returnează rezultatul

```
     double floor (double x);
      float floor (float x);
long double floor (long double x);
```

Exemple:

a. Modulul unui număr

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
    float x;
    cout << "x="; cin>>x;
    cout<<abs(x)<<endl;
    return 0;}
```

Output:

```
abs (3.1416) = 3.1416
abs (-10.6) = 10.6
```

b. Radicalul unui număr

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

Output:

```
sqrt(1024.000000) = 32.000000
```

```
    float x;
    cout << "x="; cin>>x;
    cout<<sqrt(x)<<endl;
    return 0;}
```

c. Ridicarea unui număr la puterea y

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
    float x,y;
    cout << "x=";cin>>x;
    cout << "y=";cin>>y;
    cout<<pow(x,y)<<endl;
    return 0;}
```

Output:

```
7 ^ 3 = 343.000000
4.73 ^ 12 = 125410439.217423
32.01 ^ 1.54 = 208.036691
```

d. Rotunjirea unui număr la cel mai apropiat întreg mai mare decât x

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
    float x;
    cout << "x=";cin>>x;
    cout<<ceil(x)<<endl;
    return 0;}
```

Output:

```
ceil of 2.3 is 3.0
ceil of 3.8 is 4.0
ceil of -2.3 is -2.0
ceil of -3.8 is -3.0
```

e. Trunchierea unui număr la cel mai apropiat întreg mai mic decât el

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
    float x,y;
    cout << "x=";cin>>x;
    cout<<floor(x)<<endl;
    return 0;}
```

Output:

```
floor of 2.3 is 2.0
floor of 3.8 is 3.0
floor of -2.3 is -3.0
floor of -3.8 is -4.0
```

Valoarea returnată de către o funcție poate fi:

- afișată:
  - `cout << sqrt(x);` // pentru x=4, se va afișa valoarea 2
- atribuită unei variabile:
  - `y=sqrt(x);` // pentru x=4, y va lua valoarea 2
- folosită într – o expresie
  - `z=(1+sqrt(x))/3;` //pentru x=4, se calculează expresia din dreapta (1+2)/3, rezultă 1, valoare atribuită lui z
- transmisă ca paramentru altei funcții
  - `y=floor(sqrt(x));` // pentru x=2, se calculează radical din 2, rezultă 1.41, iar floor(1.41) returnează valoarea 1

Exerciții:

1. Ce valoare va returna expresia: `floor( -47%5 – sqrt(2))/ceil(19.3)/3`, unde considerăm că radical din 2 este cu aproximație 1.41.
2. Pentru a atribui variabilei x rezultatul expresie:

$$\sqrt{\frac{a}{b}-\frac{b}{a}}$$

unde a și b sunt două variabile reale nenule, se utilizează instrucțiunea de atribuire:

a. `x=sqrt(a*a-b*b)/(a*b)`
b. `x=sqrt(a/b)-(b/a)`
c. `x=sqrt(a*a-b*b)/(a*b)`
d. `x=sqrt((a*a-b*b))/(a*b))`

3. Care dintre următoarele expresii C++ are valoarea 1 dacă variabila x memorează un număr natural pătrat perfect? De exemplu 4 este număr pătrat perfect, deoarece radical din 4 este 2. Alte numere pătrate perfecte sunt: 9,16,25,36,49,61, etc.

a. `sqrt(x)==floor(sqrt(x));`
b. `floor(sqrt(x))!=ceil(sqrt(x));`
c. `sqrt(x)!=floor(sqrt(x));`
d. `x – floor(x)=ceil(x);`

4. Ce afișează instrucțiunea: `cout << floor(27%4) + ceil(27.3) / 4` ?
5. Pentru a atribui variabilei x rezultatul expresie:

$$\sqrt{a^2 - b^2}$$

unde a și b sunt două variabile reale nenule, se utilizează instrucțiunea de atribuire:

a. `x=sqrt(a*a)–sqrt(b*b);`
b. `x=sqrt(a*a – b*b);`
c. `x=sqrt(a+b)*(a – b);`
d. `x=sqrt(a*a)–(b*b);`

6. Dintre expresiile C/C++ de mai jos, cea care are valoarea 1 dacă și numai dacă numărul întreg memorat în variabila întreagă x NU aparține reuniunii de intervale [-5,-2] U [2,5] este:

a. `abs(x)<2 && abs(x)>5`
b. `abs(x)<2 || abs(x)>5`
c. `abs(x-5)<2`
d. `abs(x-5)>abs(x-2)`