molnar1956@yahoo.com

3. Apelul unei funcţii. Revenirea dintr-o funcţie

3.1 Apelul **unei** funcţii care nu returnează o valoare are forma generală:

nume_funcţie (lista parametrilor efectivi);

unde:

- parametru efectiv = parametru actual = parametru real = parametru de apel
- **lista parametrilor efectivi =** poate fi vidă, poate fi o expresie sau mai multe despărţite prin virgulă

Efectul instructiunii de apel este:

- crearea tabelei de parametrii actuali;
- crearea variabilelor locale functiei;
- executarea corpului de instructiuni al functiei;
- desfintarea tabelei de parametrii si a variabilelor locale;
- revenirea la instruciunea urmatoare din program

molnar1956@yahoo.com


O funcţie care returnează o valoare poate fi apelată fie printr-o instrucţiune de apel simplă (cazul funcţiilor care nu returnează valori) şi în plus poate fi apelată ca operand al unei expresii. În cazul în care funcţia se apelază print-o instrucţiune de apel simplă, rezultatul funcţiei se pierde.

Când funcţia se apelează ca operand, valoarea returnată va fi utilizată în expresie. La apelul unei funcţii, valorile parametrilor efectivi se atribuie parametrilor formali corespunzători. În cazul în care unul din tipul unui paramatru efectiv diferă de tipul parametrului formal corespunzător, parametrul efectiv va fi convertit spre parametru formal (dacă este posibil, altfel compilatorul generează eroare).

În momentul în care se întâlneşte un apel de funcţie, controlul execuţiei programul este transferat primei instrucţiuni din funcţie, urmând a se executa secvenţial instrucţiunile funcţiei.

Revenirea dintr-o funcţie se face în unul din următoarele cazuri:

- după execuţia ultimei instrucţiuni din corpul funcţiei
- la întâlnirea unei instrucţiuni **return**

Instrucţiunea return are formatele:

- **return ;**
- **return expresie ;**

molnar1956@yahoo.com

Exemplul 3.1	Exemplul 3.2

# include <iostream> 	
using namespace std;
void f1 ()
{
cout << "abc";
}
 	
void main ()
{
f1();
}
	#include <iostream>
	using namespace std;
	void f1 (int k)
	{
	for (int i=1; i<=k ; i++)
	cout << "abc"<< " ";
	}
	
	int main ()
	{
	f1(5);
	}

Se va afisa: abc	Se va afişa: abc abc abc abc abc
Funcţia nu returnează o valoare Funcţia nu are parametri Apelul funcţiei este o instrucţiune de apel simplă	Funcţia nu returnează o valoare Funcţia are un parametru formal de tip int Apelul funcţiei este o instrucţiune de apel simplă şi se face cu ajutorul unui parametru actual care este de acelaşi tip cu tipul parametrului formal corespunzător

molnar1956@yahoo.com

Exemplul 3.3

```
#include <iostream>
#include <math.h>
using namespace std;
int prim (int x)
{
    int nr_div=0;
    for (int i=2; i<=sqrt(x); i++)
    if (x%i==0)
    nr_div++;
    if (nr_div==0)
                return 1;
           else
               return 0;
}

int main ()
{
    int N;
    cout << "N="; cin >> N;
    if (prim(N))
         cout << "PRIM";
      else
        cout << "NU E PRIM";
}
```

Exemplul 3.4

```
# include <iostream>
# include <math.h>
using namespace std;
int prim (int x)
{
    for (int i=2; i<=sqrt(x); i++)
    if (x%i==0)
    return 0;
    return 1;
}

int main ()
{
    int N;
    cout << "N="; cin >> N;
    if (prim(N))
         cout << "PRIM";
       else
         cout << "NU E PRIM";
}
```

molnar1956@yahoo.com

Funcţia returnează o valoare de tip int ***Funcţia are un parametru formal de tip int*** ***Rezultatul funcţiei este este utilizat în cadrul unei expresii.***	***În cazul în care se întâlneşte un divizor a lui x se execută instrucţiunea return 0. Astfel apelul funcţiei se încheie.*** ***Dacă x este număr prim, instrucţiunea return 0 nu se execută niciodată şi în acest caz, după terminarea execuţiei instrucţiunii for, se execută instrucţiunea return 1 (care determină încheierea execuţiei funcţiei).***

molnar1956@yahoo.com

OBS: În cazul în care **tipul returnat de funcţie lipseşte** din definiţia funcţiei, acesta este implicit: **int** şi nu **void.**

Exemplul 3.5	Exemplul 3.6	Exemplul 3.7

#include <iostream>
using namespace std;
 	
p( )
{
cout << " abcd";
}
 	
int main ()
{
cout << p();
}
	#include <iostream>
	using namespace std;
	
	p( )
	{
	return 25;
	}
	
	int main ()
	{
	cout << p( );
	}
		#include <iostream>
		using namespace std;
		
		void p( )
		{
		cout << "void";
		}
		
		int main ()
		{
		cout << p();
		}

Compilatorul generează eroare deoarece lipseste tipul returnat de functie	Compilatorul generează eroare deoarece lipseste tipul returnat de functie	Compilatorul generează eroare deoarece o functie cu tipul returnat void nu se poate apela in cadrul altei functii, in cazul nostru cout

molnar1956@yahoo.com

	Exemplul 3.6	Exemplul 3.7

	#include <iostream>
	using namespace std;
	
	int p( )
	{
	return 25;
	}
	
	int main ()
	{
	cout << p( );
	}
		#include <iostream>
		using namespace std;
		
		void p( )
		{
		cout << “abcd";
		}
		
		int main ()
		{
		cout << p();
		}

	Se afişează 25	Se afişează abcd