4. Aplicaţii minimale C/C++ (Minimum C/C++ applications)

1. Obiective

- Înţelegerea structurii unui program C/C++
- Înţelegerea noţiunilor: comentarii, directive preprocesor, declaraţii globale, funcţii, definiţia şi prototipul unei funcţii, apelul unei funcţii, parametri formali şi parametri actuali
- Scrierea şi testarea unor programe simple C/C++

1'. Objectives

- Understanding the structure of a C/C++ program
- Understanding the meaning of: comments, preprocessor directives, global declarations, functions, a function's definition and prototype, calling functions, formal and actual parameters
- Writing and testing some simple C/C++ programs

2. Breviar teoretic

Forma generală a unei aplicaţii C/C++ urmăreşte de obicei următoarele etape:

- Comentarii iniţiale, ce prezintă scopul aplicaţiei şi realizatorul ei
- Directive preprocesor de tip *include*
- Directive preprocesor de tip *define*
- Declaraţii globale de variabile sau alte tipuri de date
- Prototipuri de funcţii
- Funcţia *main*
- Definirea celorlalte funcţii din cadrul aplicaţiei

Limbajul C/C++ este un limbaj procedural, la baza lui fiind procedura, numită *funcţie*.

Tipurile de variabile în C sunt recunoscute prin cuvintele cheie:

char,	pentru caractere;
int,	pentru întregi cu semn;
void,	neprecizat;
float,	real simplă precizie;
double,	real dublă precizie, iar modul de implementare poate fi modificat cu ajutorul declaraţiilor suplimentare *signed, unsigned, long, short.*

In C++ avem in plus tipurile de date:
bool – tipul Boolean, pentru stocarea valorilor booleene sau logice. O variabila booleana poate avea valoarea *true* sau *false*
wchar_t – tot un tip de data character, dar care este reprezentat pe 2 octeti
long long int – tip pentru date intregi pe 64 de biti

Preprocesarea permite prelucrarea unui program sursă C sau C++ înainte de a fi supus compilării şi asigură:

- includeri de fişiere cu text sursă;
- definiţii şi apeluri de macro-uri;
- compilare condiţionată.

Constantele sunt valori fixe (numerice, caractere sau șiruri de caractere), care nu pot fi modificate de program.
Pentru a specifica caractere care nu pot fi afișate, secvențele escape sunt folosite cu caracterul „\" („backslash").

3. Exemple

Exemplele următoare, atât din acest capitol cât şi din celelalte, au fost testate în mediul de programare MS Visual C++ 201y.
Specific acestui mediu de programare este utilizarea pentru intrări/ieşiri C++ a combinaţiei:

```
#include <iostream>
using namespace std; // specifica utilizarea spatiului de nume standard
```

Aceasta combinatie include implicit si bibliotecile de baza ale limbajului C.

In alte medii de programare C/C++ doar:

```
#include <iostream.h>
```

se foloseşte pentru operatii de intrare/iesire C++.

Exemplul 1: program pentru citirea şi afişarea unui întreg folosind funcţii.

```
// directive preprocesor

// includerea unor fisiere antet ce contin prototipurile functiilor din
// biblioteca standard folosite in program
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
//#include <conio.h>

// declaratii globale

// prototipuri functii: pentru fiecare functie precizeaza numele, lista cu
// tipurile parametrilor formali si tipul returnat
int cit_int(void);
void afis_int(int);

// functia main(): are aceeasi structura ca orice functie
// antet functie: nume, lista cu numele si tipurile parametrilor, tipul returnat


int main()
// corp functie
{
// declaratii locale
int n;                          // variabila locala in care se va memora numarul citit

    // instructiuni
    n = cit_int(); // apel functie de citire si atribuire rezultat
    afis_int(n); // apel functie de afisare
    //_getch();
    return 0;
}//main

// definiţii functii

// functie pentru citirea unui intreg
// antet functie: nume, lista cu numele si tipul parametrilor, tipul returnat
int cit_int(void)
// corp functie
{
// declaratii locale
int nr;                 // variabila intreaga in care se va memora valoarea citita

    // instructiuni
    // apel functie din biblioteca standard pentru afisarea unui mesaj
    printf("\nIntroduceti un numar intreg : ");
    // apel functie din biblioteca standard pentru citirea unui intreg
    scanf("%d", &nr);

    // returnare rezultat
    return nr;
} //cit_int
```

```
// functie pentru afisarea unui intreg
void afis_int(int nr)
{
	// apel functie din biblioteca standard pentru afisarea unui mesaj
	// si a numarului intreg primit ca paramteru
	printf("\nAti introdus numarul:  %d\n", nr);
} // afis_int
```

Exemplul 2: program pentru calculul mediei aritmetice a două numere întregi.

```
// directive preprocesor
// includerea unor fisiere antet ce contin prototipurile functiilor din
// biblioteca standard folosite in program
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
//#include <conio.h>

// declaratii globale

// prototipuri functii
int cit_int(void);
float medie_a(int, int);


int main()
{
// declaratii locale
float ma;		// variabila flotanta pentru media aritmetica
int n1, n2;		// variabile intregi pentru numerele intregi ce vor fi citite

	// apel functie de citire a unui intreg
	n1 = cit_int();
	n2 = cit_int();

	// apel functie de calcul a mediei aritmetice
	ma = medie_a(n1, n2);

	// afisare rezultat
	printf("\n\t Media aritmetica : %f\n", ma);
	//_getch();
	return 0;
}

// definitii functii

// functie de calcul a mediei aritmetice pentru doua numere intregi
float medie_a(int n1, int n2)
{
// declaratii locale
float medie;			// variabila flotanta pentru stocarea mediei aritmetice

	medie = (n1 + n2)/2.0f;
	return medie;
}

// functie pentru citirea unui intreg
int cit_int(void)
{
// declaratii locale
```

```
int nr;              // variabila întreaga in care se va memora valoarea citita

    printf("\nIntroduceti un numar intreg : ");
    scanf("%d", &nr);
    return nr;
}
```

Exemplul 3: program pentru exemplificare caractere, siruri de carcatere, simple si wide.

```
// characters, wide characters, and character strings
#include<iostream>
using namespace std;

int main( ) {
        char character = 'a';//0x61=97
        wchar_t wide_character = L'a';
        cout << "The character is: " << character << endl;
        cout << "The character size: " << sizeof(character) << endl;//1
        wcout << "The wide character is: " << wide_character << endl;
        cout << "Wide character size: " << sizeof(wide_character);//2
        const char str1[] = "\nThis is character array";
        cout << str1 << "\nThe character array size: " << sizeof(str1) << endl;
        const wchar_t str2[] = L"This is wide character array";
        wcout << str2 << "\nThe wide character array size: " << sizeof(str2)<< endl;
        return 0;
}
```

4. Întrebări:

- Care este structura unui program C/C++ ?
- Ce sunt comentariile ? La ce se folosesc ?
- Care este deosebirea între prototipul şi definiţia unei funcţii ?
- Cum se face apelul unei funcţii ?
- Cum se face revenirea din funcţii ?

5. Teme:

1. Să se scrie un program pentru determinarea mediei aritmetice a trei numere neîntregi.
2. Să se scrie un program pentru pentru determinarea mediei geometrice a două numere întregi.
3. Să se scrie un program C/C++ care defineşte o variabilă întreagă care va fi iniţializată cu valori constante. Afişaţi rezultatul cu ajutorul supraîncărcării operatorului << şi a lui *cout*.
4. Definiţi un şir de caractere care va fi afişat cu *cout*. Definiţi alte şiruri de caractere folosind secvenţe escape. Verificaţi utilizarea spaţiilor albe.
5. Să se scrie un program în care se dau 3 numere întregi și se cere să se calculeze suma lor ponderată, ponderile fiind numere subunitare a căror suma este 1.
6. Definiți într-un program constante simbolice de tipuri diferite (întregi, reale, șiruri de caractere). Afișati valorile acestor constante utilizând operatorul << și fluxul *cout*.
7. Definiți 3 numere reale *a*, *b*, și *c*. Afișați rezultatul operației *1/a+1/b+1/c*. Efectuati aceeasi operatie considerand ca si intrare numere intregi.

4'. Questions

- What is the structure of a C/C++ program?
- What are comments? What are they used for?
- What is the difference between a function's prototype and definition?
- How can a function be called?
- How is a function's returning done?

5. Homework

1. Write a program that determines the average value of 3 non-integer numbers.

2. Write a program that determines the geometric average of 2 integer numbers.
3. Write a C/C++ application that defines an integer variable, initialized with several constant values. Display its value by overloading the << operator and by using the *cout* object.
4. Define an array of characters that will be displayed using *cout*. Display other character arrays and use escape sequences. Verify the usage of the whitespaces.
5. Write a program that defines 3 integer values. Calculate and display their weighted sum, the weights being represented as positive values smaller than 1 that add up to 1.
6. Define several symbolic constants of different types (integer numbers, real numbers, arrays of characters). Display their values using *cout* and the << operator.
7. Define 3 real numbers named *a*, *b* and *c*. Display the value of *1/a + 1/b + 1/c*. Display the same result considering as input integer numbers.