Laborator 1 – Incompatibilităţi/Diferenţe între C şi C++

Obiective:

- prezentarea de concepte care nu fac parte din C si care le vom utiliza mai apoi în programare orientată obiect

Incompatibilităţi între C şi C++

1. Declaraţii şi definiţii de funcţii (diferență istorică)

C++ acceptă numai varianta modernă de definire a unei funcţii în timp ce C acceptă şi varianta Kernighan şi Ritchie.
În C++ este obligatorie declararea prototipului funcţiei sau definirea funcţieiînainte de a fi utilizată.

2. Pointeri void

Pentru tipul (void*) C++ admite conversia implicită în sensul:
Tip pointer →void*

Exemplu:

```
void * vp;
int *ip;
vp = ip; //corect in C si C++
ip = vp; //corect in C, incorect in C++
ip = (int*)vp; //corect in C si C++
```

Completări C++

1. Comentarii

C++ permite comentare unei singure linii folosind //pe lângă comentariului unui bloc de instrucţiuni /**/.

Exemplu:

```
void main(){
    /** Comentariu in stil C*/
    int a=0; /**declarare si initializare a*/
    //Comentariu in stil C++
    int b=0; //declarare si initializare b
}
```

2. Tipul Boolean

C nu oferă printre tipurile de bază tipul boolean. Acesta se poate simula prin folosirea unei enumerări.

```
void main(){
   bool b = true;
   b = false;
}
```

```
typedef enum{FALSE, TRUE} bool;
```

3. Operaţii de intrare/ieşire cu consola

C++ pune la dispoziţie un mod mult mai flexibil de lucru cu consola. Avem două stremuri standard:

- cin –console input (tastatura) ~ stdin
- cout –console output (monitorul) ~ stdout

Transferul informaţei este efectuat de operatorul >> pentru intrare (de la cin) şi operatorul << pentru ieşre (căre cout).

```
cout<< var; //scrie variabila var la cout
cin >> var; //citeste variabila var de la cin

// operatii multiple
cout << var1 << var2 … << varN;
cin >> var1 >> var2 …>>varN;
```

Pentru a putea folosii operaţiile cu consola trebuie inclus fişierul antet *iostream* şi declararea folosirii spaţiului de nume: *using namespace std*.

Exemplu: Să se realizeze un program care citeşte de la tastatură un şir de caractere care reprezintă un nume şi două variabile numerice una de tip int şi una de tip double şi apoi le afişează.

```
#include <iostream>
using namespace std;
int main(){
  char nume[50];
  int i;
  double d;
  cout << "Introduceti un nume: ";
  cin >> nume;
  cout << "Introduceti un numar & o valoare reala:";
  cin >> i >> d;
  cout<< "Hello " << name << "!\n";
  cout << "Numarul intreg este: " << i << endl;
  cout << "Numarul real este: " << d << endl;
  return 0;
}
```

4. Plasarea declaraţiilor

Spre deosebire de C care impune gruparea variabilelor locale la începutul funcţiei, C++ permite declararea de variabile oriunde in corpul funcţiei înainte de a fi utilizate.

Exemplu: Să se calculeze suma primelor *n* numere, *n* se citeşte de la tastatură.

```
#include <iostream>
using namespace std;
int main(){
   int n;
   cout << "Introduceti n: ";
   cin >> n;
   int suma = 0;
   for (int i=0; i<=n; i++)
        suma += i;
   cout << "Suma este: " << suma << endl;
   return 0;
}
```

5. Variabile referinţă

C++ permite declararea unor identificatori ca referinţe de obiecte de un anumit tip. La declararea unei variabile referinţă este obligatorie iniţializarea sa cu adresa unei variabile deja definite.

Exemplu: Declararea unei referinţe la un întreg:

```
int main(){
   int n=6;
   int & ir = n;
   cout<< "n=" << n << " ir=" <<ir << endl;
   ir = 100; // echivalent cu n =100;
   cout<< "n=" << n << " ir=" <<ir << endl;
   return 0;
}
```

Folosirea parametrilor formali referinţă in C++ permite realizarea transferului prin referinţă ca în Pascal, care permite renunţarea la folosirea pointerilor când se trimit variabile.

Exemplu: Scrieţi o metodă care interschimbă două numere.

C	**C++**

`void swap(int *a, int*b);`
`void main(){`
`  int n, m;`
`  cout << "n="; cin >> n;`
`  cout << "m="; cin >> m;`
`  swap(&n, &m);`
`  cout << "n=" << n << endl;`
`  cout << "m=" << m << endl;`
`}`
`void swap(int *a, int*b){`
`  int temp;`
`  temp = *a;`
`  *a = *b;`
`  *b = temp;`
`}`
	`void swap1(int &a, int&b);`
	`void main(){`
	`  int n, m;`
	`  cout << "n="; cin >> n;`
	`  cout << "m="; cin >> m;`
	`  swap1(n, m);`
	`  cout << "n=" << n << endl;`
	`  cout << "m=" << m << endl;`
	`}`
	`void swap1(int &a, int&b){`
	`  int temp;`
	`  temp = a;`
	`  a = b;`
	`  b = temp;`
	`}`

6. Parametricu valori implicite

C++ oferă posibilitatea declarării de funcţii cu valori implicite ale parametrilor. Acest lucru permite ca la apelarea funcţiilor să se poată omite parametrii cu valori implicite, aceşti parametri se vor completa cu valoare implicită. Dacă la apelul unei funcţii lipseşte un parametru căruia i-a fost precizata o valoare implicita, compilatorul va atribui aceasta valoare în momentul apelului.

OBSERVAŢII:
1. Valorile implicite se specifică o singură dată în prototipul sau definiţia funcţiei.
2. Argumentele cu valori implicite trebuie plasate la sfârşitul listei de argumente a funcţiei.

```
#include <iostream>
using namespace std;
//double fct1(char 'A', int, float = 12);//eroare de compilare
void fct3(int, long=99, float=10.6);
void main(){
   float f = 1.4;
   fct3(0);//apel corect
   fct3(1,f); //apel incorect
```

```
   fct3(2, 2.6); //apel incorect
}
void fct3(int i, long l, float f){
   cout << "i=" << i << " l=" << l << " f=" << f << endl;
}
```

7. Compilare în linia de comandă

Medii de compilare execuţie:

- Folosirea unui IDE ca CodeBocks
- Compilarea şi execuţia în linia de comandă
  - Compilatoare g++ sau clang++
  - Make file este un fişier cu numele makefile

```
executabilCreion:
# inlocuim compilarorul de C cu cel de CC
CC=g++
# optiuni ale compilatorului
CFLAGS=-c -Wall
all: executabilCreion
        executabilExCreion: creion.o main.o
${CC} main.o creion.o -o executabilCreion
        main.o:
$(CC) $(CFLAGS) main.cpp
        creion.o:
$(CC) $(CFLAGS) creion.cpp
clean:
        - rm -rf *o executabilCreion
```

Makefile-ul de mai sus este utilizat pentru un program C++ care conţine 3 fişiere: main.cpp, creion.h, creion.cpp şi prezintă modul în care se pot reţine paşi de compilare şi execuţie pentru reutilizare într-un fişier. Pentru a executa fişierul makefile sub Linux se scrie targetul principal ./executabilCreion

Probleme suplimentare

1. Realizati o prezentare despre scriere la tastatură formatată și rolul folosiri funcției cin.ignore().
2. Realizati un program care permite simularea unui calculator simplu
   a) Programul va permite sa introduca date in urmatoarele formate
      i. Numar operator numar (citirea separata a datelor ca numar caracter numar)
      ii. Citirea unui singur șir de caractere și parsarea lui astfel încât să se obțină operatori și operanzi
   b) Fiecare operatie va fi realizată intr-o funcție (folosiți pointeri la funcții pentru a decide care funcție va fi apelata în functie de operația introdusă)