Les sous-programmes

I-Introduction

Application

Sous-programme 1

Programme principale (Main)

Appel 1

Appel 2

Sous-programme 2

Lorsqu’un programme est long, il est irréaliste d’écrire son code en un seul tenant. On découpe le programme en plusieurs parties plus petites et donc plus simple a coder qu’on assemble pour former l’application finale. De plus, les sous programmes permettent de répéter des morceaux de codes identiques. Un sous-programme possède un nom, des variables, des instructions, un début et une fin. Mais contrairement à un programme, un sous-programme ne peut pas s’exécuter tout seul.

L’instruction permettant d’exécuter un sous-programme, est nommée appel. On dit qu’un sous programme est demandé par un programme appelant (il peut être soit un sous-programme soit un programme principale).

Remarque :

* Les instructions du sous-programme sont écrites en dehors du programme appelant.
* Si un programme écrit n’est jamais appelé, il ne sera jamais exécuter.
* Lorsque l’exécution d’un sous-programme est terminé, il y a un retour à l’instruction qui suit l’appel.

Exemple :

#include…

//Déclaration d’un sous-prog

Void dessinerLigne(){

Int i ;

For(i=1 ; i<=5 ; i++){

Cout<< »i » ;

}

Cout<<endl ; //saut de ligne

}

// programme principale

Int main(){

Int n ;

For(n=1 ; n<10 ; n++){

dessinerLigne() ; //appel du sous prog

}

System(« pause ») ;

Return 0 ;

}

II-Fonctions et procédures

Il existe 2 types de sous-programme :

1-Les procédures

* Réalisent une suite d’instruction (traits) exécutées a l’appel dans le programme appelant.
* L’appel d’une procédure constitue une instruction en lui-même, c’est-à-dire qu’il occupe une ligne de code.

2-Les fonctions

* Elles réalisent aussi une suite d’instructions (comme les procédures) en plus, elles renvoient une valeur résultat. Cette valeur doit être exploitée dans le programme appelant.
* L’appel d’une fonction est remplacé par la valeur renvoyée. Cette valeur doit etre incluse dans une instruction dans le programme appelant.

Syntaxes

1-Les procédures

Void nomProcédure([<parametre>]){

//traitement

}

2-Les Fonctions

<type\_retour> nomFonction(<param>]){

//traitement

Return <resultat> ;

}

Exemple :

#include…

//déclaration des sous programmes

Double ttc(){\* double resultat ;

Double pht ;

Cout<< »Prix hors taxe : » ;

Cin>>pht ;

Resultat= pht + 1.2 ; //20%

Return resultat ;

}

Int main(){

Double total ;

Int i ;

Total =0 ;

For(i=1 ;i<=5 ;i++){

Total=total+ttc() ;

}

Cout<<  « Le prix total = »<<total<<endl ;

System(« pause ») ;

Return 0 ;

}

Void nombre (…){

III-Variables locales et variables globales

Application

Variables Globales

Variables locales

Sous-programme 1

Variables locales

Sous-programme 2

Variables locales

Main

L’endroit ou est déclarée une variable est très important car il détermine dans quel sous-programme elle peut être utilisée.

Variable locale : est déclarée à l’intérieur d’un sous-programme et elle n’est utilisable que dans le sous-programme ou elle a été déclarée.

Variable globale : est déclaré a l’extérieur du programme principale et des sous programmes. Elle est connue à l’ensemble des sous-programmes et du programme principal.

Exemple :

#include <cstdlib>

#include <iostream>

using namespace std;

//Déclaration de variables

//globales

int nbE; //nombre d'étoiles

void dessinerLigne(){

int i;

for(i=1;i<=nbE;i++){

cout<<"\*";

}

cout<<endl;//saut de ligne

}

int main(...){

int nbL;//nombre de ligne

int i;

cout<<"Entrez le nombre de lignes: ";

cin>>nbL;

cout<<"Le nombre d'étoiles/lignes: ";

cin>>nbE;

for(i=1;i<=nbL;i++){

dessinerLigne();

}

system(" pause ") ;

return 0 ;

}

Variables de mêmes noms :

Le même nom de variable peut désigner plusieurs variables locales différentes : Chaque sous-programme aura sa propre version (case mémoire) de cette variable.

Exemple : Le i déclaré dans le main et le dessignerLigne.

En revanche, la variable globale n’existe qu’en 1 seul exemplaire. Si un sous-programme déclare une variable locale qui porte le même nom qu’une variable globale, l’accès a la variable globale devient impossible car c’est toujours les variables locales qui sont prioritaire par rapport aux variables globales.

Ecrire un programme permettant de convertir un nombre de seconde en heures, minutes et secondes.

IV-Les paramètres

La communication de sous-programme et programme appelant peut se faire par intermédiaire de variables locales particulière qu’on appelle paramètres (ou arguments).

Leurs déclarations se fais dans l’entête du sous-programme.

Syntaxe : Type retour NomSousProgramme (Type nom 1)

(Type nom 2)

/!\ pour chaque paramètres, on déclare son nom et son type (comme une variable)

Chaque paramètre est déclaré individuellement

Lors de la déclaration des paramètres, on définit :

* Le type de chaque paramètre
* Le nombre de paramètre
* L’ordre des paramètres

Exemple :

#include…

void dessinerLigne(int nbe){

int i ;

for(i=1 ;i<=nbe ;i++){

cout<<”\*”;

}

cout<<endl;

}

int main(...){

int nbe,nbl,i;//nombre de ligne

cout<<"Entrez le nombre de lignes: ";

cin>>nbl;

cout<<"Le nombre d'étoiles/lignes: ";

cin>>nbe;

for(i=1;i<=nbl;i++){

dessinerLigne(nbe);