Contenu de la matière:

Chapitre 1. Présentation du langage C++ (1 Semaine)

Historique, Environnement de développement en C++ (création d'objets, compilation, débogage, exécution …).

Chapitre 2. Syntaxe élémentaire en langage C++ (2 Semaines)

Instructions Commentaires, Mots clés et mots réservés– Constantes et variables, Types fondamentaux Opérateurs (unitaires, binaires, priorité,…).

Chapitre 3. Structures conditionnelles et Boucles (2 Semaines)

If/else, Switch/case, Boucle for, Boucle while, Boucle do/while.

Chapitre 4. Entrées/sorties (2 Semaines)

Flux de sortie pour affichage, Flux d'entrée clavier, Cas des chaines de caractères**.**

Chapitre 5. Pointeurs et Tableaux (2 Semaines)

Pointeurs, Références, Tableaux statiques, Tableaux et pointeurs, Tableaux dynamiques, Tableaux multidimensionnels.

Chapitre 6. Fonctions (2 Semaines)

Prototype d'une fonction, Définition d'une fonction, Appel d'une fonction, Passage d'arguments à une fonction, Surcharge d'une fonction, Fichiers.

Chapitre 7. Fichiers (1 Semaine)

Mode texte, Mode binaire, Fichier en C.

Chapitre 8. Programmation orientée objet en C++ (3 Semaines)

Introduction, Concept de classes et objets, Héritage, Méthodes particulières (constructeurs, destructeurs…), Programmation procédurale ou structurée, Programmation par objets.

Chapitre : 01 : Présentation du langage C++

1. Historique :

✓ Le langage C est un langage évolué et structuré, assez proche du langage machine destiné à des applications de contrôle de processus (gestion d'entrées/sorties, applications temps réel, ...).

- ❖ Le C a été inventé au cours de l'année 1972 dans les laboratoires Bell par Dennis Ritchie et Ken Thompson.
- ❖ En 1978, Brian Kernighan, qui aida à populariser le C, publia le livre «The C programming Language».
- ❖ Dans les années 80, Bjarne Stroustrup développa le C++ afin d'améliorer le C, en lui ajoutant des «classes». Le premier nom de ce langage fut d'ailleurs «C with classes».
- ❖ Ce fut en 1998 que le C++ fut normalisé pour la première fois. Une autre norme corrigée fut adoptée en 2003.
- ❖ Une mise à jour importante fut C++11, suivie de C++14, ajoutant de nombreuses fonctionnalités au langage. Toutes ces normes permettent une écriture indépendante du compilateur. Le C++ est le même partout, pourvu qu'on respecte ces normes.

✓ Le langage C++ possède assez peu d'instructions, il fait par contre appel à des bibliothèques, fournies en plus ou moins grand nombre avec le compilateur.

Exemples: math.h : bibliothèque de fonctions mathématiques
stdio.h : bibliothèque d'entrées/sorties standard

✓ **On ne saurait développer un programme en C++ sans se munir de la documentation concernant ces bibliothèques.**

Comment traduire

- Compilation (traduire avant l'exécution)
- Interprétation (traduire pendant l'exécution)

✓ Les compilateurs C sont remplacés petit à petit par des compilateurs C++, Un programme écrit en C est en principe compris par un compilateur C++.

2. Etapes permettant l'édition, la Mise Au Point, l'exécution d'un programme

- ***Edition du programme source***, à l'aide d'un éditeur (exemple : Dev C++). Le nom du fichier contient l'extension .CPP, exemple: EXI_1.CPP (menu « edit »).

- ***Compilation du programme source,*** c'est à dire création des codes machine destinés au microprocesseur utilisé. Le compilateur indique les erreurs de syntaxe mais ignore les fonctions-bibliothèque appelées par le programme.
- Le compilateur génère un fichier binaire, non listable, appelé fichier objet: EXI_1.OBJ (commande « compile »).
- ***Editions de liens***: Le code machine des fonctions-bibliothèque est chargé, création d'un fichier binaire, non listable, appelé fichier exécutable: EXI_1.EXE (commande « build all »).
- ***Exécution du programme*** (commande « flèche jaune »).

✓ Comme dans la plupart des cours de programmation, on commence par un **BONJOUR**: Programme le plus simple qui affiche un message à l'écran.

Un premier exemple :

```
#include<iostream> /* bibliotheque d'entrees-sorties standard */
using namespace std;
int main()
{
cout << "BONJOUR" << endl; /* utilisation d'une fonction-bibliotheque */
cout << "Appuyez sur une touche pour continuer …" << endl;
system("PAUSE"); /* Attente d'une saisie clavier */
return 0;
}
```

❖ **La directive #include**

On place en général au début du programme un certain nombre d'instructions commençant par ***#include***. Cette instruction permet d'inclure dans un programme la définition de certains objets, types ou fonctions. Le nom du fichier peut être soit à l'intérieur des chevrons < et >, soit entre guillemets ".

❖ **using namespace std;**

l'espace de nommage std. Pour simplifier, retenons que, dès que l'on veut utiliser cin ou cout, on doit écrire cette directive.

❖ **Le fichier iostream**

Le fichier ***iostream*** contient un certain nombre de définitions d'objets intervenant dans les entrées/sorties du programme, c'est-à-dire dans l'affichage à l'écran ou dans des fichiers. La définition de cout se trouve dans ce fichier;

❖ **La fonction main()**

Notre programme contient une fonction appelée main : c'est à cet endroit que va commencer l'exécution du programme : exécuter un programme en C++, c'est exécuter la fonction main de ce programme. Tout programme en C++ doit donc comporter une fonction main.

❖ **cout**

Permet d'afficher des messages à l'écran en utilisant l'opérateur <<. Exemple : ***cout<<"BONJOUR";***

Cette instruction affiche BONJOUR à l'écran.

Un autre exemple :

cout<<endl;

Lorsqu'on envoie ***endl*** (End of Line : fin de la ligne) vers l'affichage, on passe à la ligne suivante.

❖ **Retour de la fonction**

La dernière instruction de notre programme est ***return 0;*** Elle indique seulement que la fonction main s'est terminée correctement sans erreur particulière.

❖ **Remarque**

L'ajout de l'instruction ***system("PAUSE");*** sera parfois nécessaire pour que le programme ne s'arrête pas immédiatement après s'être ouvert. Cette instruction doit être ajouté avant l'instruction ***return 0;***.

3. Références

[1] T. Grenier T. Redarce, N. Ducros, O. Bernard, « Langage C/C++ De la syntaxe à la programmation orientée objet », INSA Lyon, 2020.

[2] C. Delannoy, « Apprendre le C++ », ÉDITIONS EYROLLES, 2007.

[3] T. Bastin, « Le langage C/C++ Méthodes numériques de la physique », 2010.

[4] R. Ruelle, « C++ », Master 1 IM, 2017.