

Module Base de la programmation *C++*

Travaux Pratiques 1

Projets *Visual Studio*, exercices d'illustrations de base *C++*

Objectif: Les objectifs des exercices de cette feuille sont d'illustrer les premières notions du langage *C++* vues en cours. Ce TP sera à réaliser sous *Visual Studio*. Il s'agira dans un premier temps de configurer un IDE puis de créer ses premiers programmes.

Exercice 1.1 Premier Projet *Visual Studio* en mode console

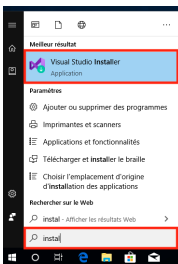
Dans cet exercice, il s'agit de créer un premier projet en mode console à partir de *Visual Studio*. Après avoir fait cet exercice, vous serez capable de :

- Créer un projet *C++ Visual Studio*.
- Lancer votre première application associée au projet.
- Afficher un Hello World en mode console.
- Utiliser des flux d'affichage et afficher différents types.

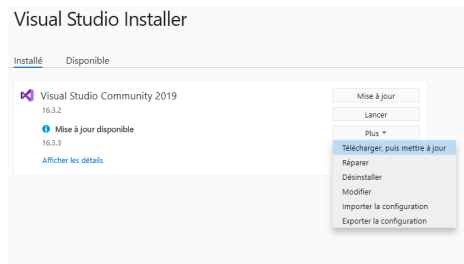
Notions de cours :

- L'opérateur *C++* `std::cout <<` permet d'afficher du texte dans la sortie standard.
 - Le code exécuté au lancement du programme est situé dans la fonction `main()`.
-

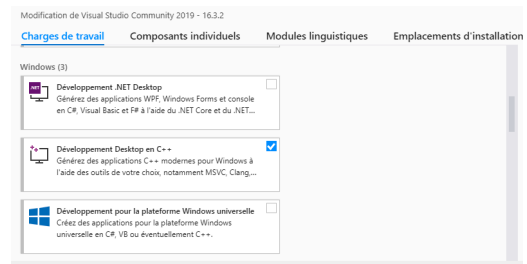
1. Vérifiez que *Visual Studio* a déjà été installé sur votre machine.
2. Pour créer une application minimale en mode console, il faut d'abord vérifier (image (b)) que le composant *Développement Desktop en C++* (image (c)) est bien installé. Profitez en pour mettre à jour la version de *Visual Studio*.



(a)

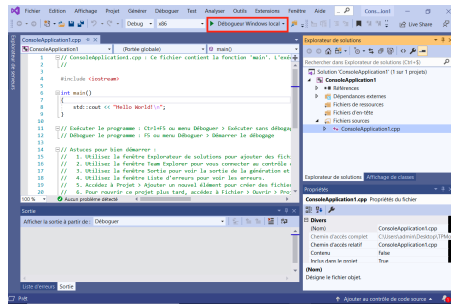


(b)

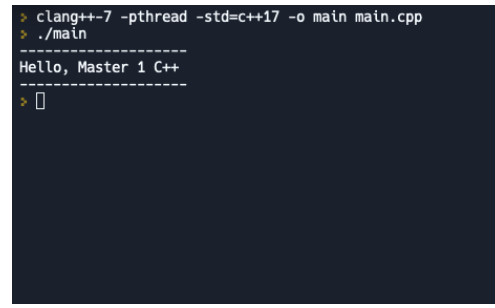


(c)


3. Créez un nouveau projet de type *Application console C++* . Vous devriez avoir une interface qui ressemble à l'image (a) ci-dessous.



(a)



(b)

4. Lancez le programme à partir du bouton  **Débogueur Windows local**, et modifiez le code de façon à faire apparaître le même message de bienvenu comme celui de l'image (b) ci-dessus.
5. Modifiez votre programme de façon à rajouter les variables suivantes :
 - de type `int` initialisée avec un entier.
 - de type `double` initialisée par un nombre décimal.
 - de type `std::string` initialisée avec une chaîne de caractères.
6. Affichez le contenu de chaque variable dans la console.
7. En utilisant la fonction template `std::numeric_limits<>::max()`, affichez les nombres maximaux représentable sur les types `int`, `uint`, et `float`. Pour utiliser cette fonction vous devez inclure le fichier `#include <limits>`.
8. Même question que précédemment mais pour ma valeur minimale avec `min()` puis `lowest()`. Noter la différence entre ces deux fonctions.

Exercice 1.2 Première saisie en mode console et premiers tests

Il s'agit dans cet exercice, de tester la saisie en mode console et d'illustrer les tests conditionnels en `C++` du cours.

Notions de cours :

`std::cin >>` permet de lire la valeur saisie par l'utilisateur.

1. Testez la commande `std::cin` pour recevoir une chaîne de caractères de type `std::string`.
2. Ecrivez des instructions de façon à obtenir un affichage personnalisé et qui permette d'interagir de manière simple avec l'utilisateur en l'invitant à répondre à une question et en lui répondant. Par exemple vous devez obtenir ce type d'interactions :

```
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
-----
Hello, Master 1 C++
-----
---Exercice 2---
-----
Entrer vore nom: Jean
```

```
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
Hello, Master 1 C++
-----
---Exercice 2---
-----
Entrer vore nom: Jean
Bonjour Jean!! comment ca va ?
```

```
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
Hello, Master 1 C++
-----
---Exercice 2---
-----
Entrer vore nom: Jean
Bonjour Jean!! comment ca va ? bien
Super allons faire du C++!!
```

```
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
Hello, Master 1 C++
-----
---Exercice 2---
-----
Entrer vore nom: Jean
Bonjour Jean!! comment ca va ? bof
Allez, courage...>
```

Exercice 1.3 *Jeu du nombre mystère*

L'objectif de cet exercice est de réaliser le jeu du nombre mystère qui consiste à faire deviner un nombre à l'utilisateur.

Notions de cours :

Pour générer un nombre mystère, vous pouvez utiliser les instructions suivantes :

```
#include <time.h>
#include <stdlib.h>
...
// Initialiser la graine aléatoire
srand ( time (NULL) ) ;
// Générer un nombre aléatoire entre 0 et RAND_MAX
int r = rand()
```

1. Ajoutez des instructions de façon à générer le nombre mystère compris entre 1 et 100 et enregistrez le dans une variable avant d'inviter le joueur à jouer.
2. Testez le nombre joué et affichez les indications s'il a gagné, ou si son chiffre est trop grand ou trop petit.
3. A l'aide d'une boucle **while**, faire en sorte que le joueur puisse jouer jusqu'à ce qu'il gagne.
4. Affichez des statistiques sur la performance du joueur (entre 1 et 3 : exceptionnel, 3-10 : bonne stratégie, > 10 : peut mieux faire).
5. Améliorez le programme de façon à laisser que 10 tentatives au joueur et en indiquant au joueur le nombre de tentatives restante.
6. Améliorez le programme de façon à éviter à empêcher qu'un joueur saisissent deux fois le même chiffre.
7. Au début du jeu, rajoutez une option permettant au joueur de choisir le mode difficile ou normal.
8. En fonction du choix effectué, implémentez le mode de jeu difficile où il s'agira de donner une fois sur 3 une mauvaise réponse.

Exercice 1.4 *Nombres premiers...*

Dans cet exercice il s'agit de manipuler les fonctions et la boucles à travers la recherche des nombres premiers.

1. Ecrire une fonction **estNombrePremier** prenant en paramètre un entier et qui renvoie un booléen si le nombre passé en paramètre est premier. Pour rappel, un nombre premier est un entier qui a exactement deux diviseurs. Il s'agira de tester naïvement la divisibilité de tous les entiers entre 2 et n-1.
2. Testez votre fonction en demandant à l'utilisateur de saisir un nombre et faire en sorte à ce qu'il affiche si le nombre est premier ou non.
3. Rajoutez des instructions de façon à lister tous les nombres premiers compris entre 1 et 100.

Exercice 1.5 Type String et tests conditionnelles

L'idée de cet exercice est de réviser l'accès à un élément d'une structure de données comparable à un tableau dont les éléments sont accessibles à partir d'un indice de position. Après avoir fait cet exercice vous aurez acquis les points suivants :

- Accès à éléments d'un tableau.
- Itérer sur une structure contenant des éléments.

Notions de cours :

- L'opérateur `C++ []` permet d'accéder à un élément d'un tableau ou d'une chaîne de caractères.

Par exemple :

```
std::string s = "bonjour";  
std::cout << s[2];
```

affichera "n" en sortie standard.

- Une boucle `for` peut s'écrire de la façon suivante :

```
for (int i = 0; i < 10; i++){  
    std::cout << i << ", ";  
}
```

affichera en sortie standard : "1,2,3,4,5,6,7,8,9 ".

1. On considère le programme Blockly suivant :



Traduisez ce programme en C++.

2. Donnez une suite d'instruction qui demande à l'utilisateur d'entrer son nom et qui affiche tous les caractères avec deux espaces.
3. Reprendre la question précédente mais en utilisant des espaces variables de plus en plus grand.
4. Depuis la nouvelle norme C++ 11 il est aussi possible d'utiliser la boucle `for` en itérant directement sur une collection d'objets. La syntaxe est la suivante : `for (variable : collection) corps de boucle;`. Utilisez cette syntaxe pour afficher le nom de l'utilisateur en diagonale.

