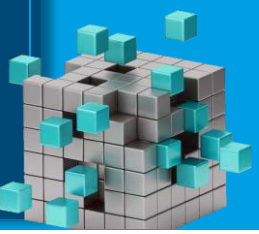
POLYTECH DIJON

TRAVAUX DIRIGÉS

Programmation Orientée Objets / C++

Premiers programmes C++



OBJECTIFS

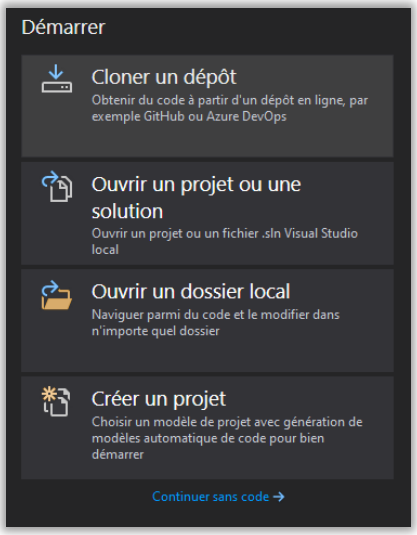
Le but de ce TD est de réaliser une introduction au langage C++ à travers de simples exercices. Les notions suivantes seront abordées :

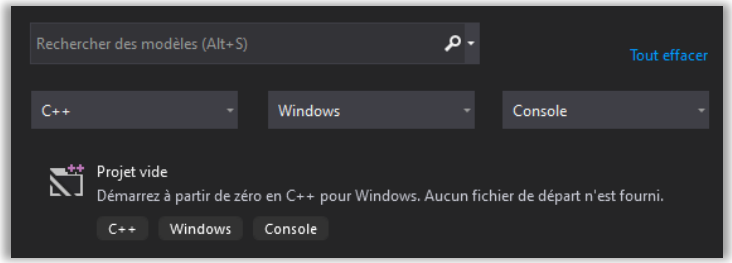
- Déclarations de variables
- Tests et boucles
- Entrée et sortie standard
- Fonctions
- Tableaux
- Classes

Note
Vos enseignants sont là pour répondre à vos questions. Alors posez des questions ! :-)

I. VISUAL STUDIO

- Démarrez Visual Studio 2022
- Cliquez sur « Créer un projet »
- Sélectionnez le langage « C++ », la plateforme « Windows » et le type de projet « Console ».
- Puis double-cliquez sur « Projet vide »

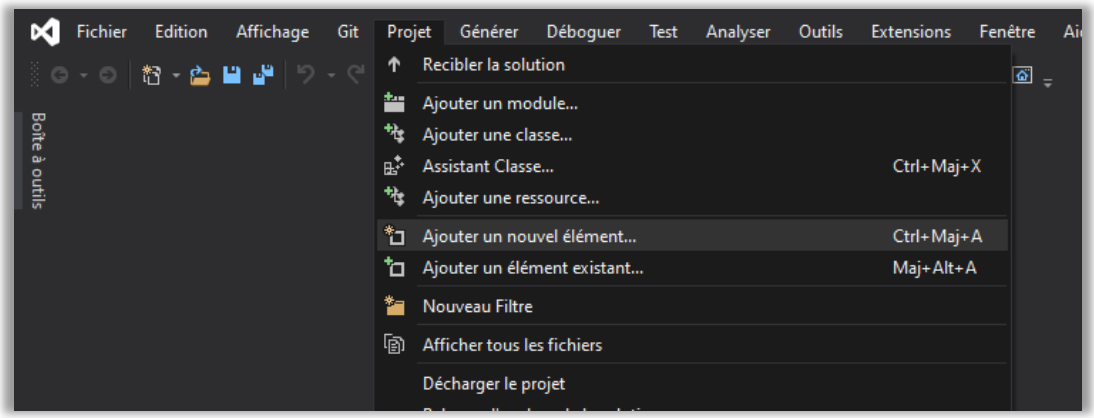




- Saisissez le nom du projet, choisissez le dossier de destination et cliquez sur le bouton « Créer » situé en bas à droite de l'écran.

II. PREMIER PROGRAMME

- Dans le menu « Projet », sélectionnez « Ajoutez un nouvel élément »



- Saisissez « main.cpp » et cliquez sur le bouton « Ajouter »

- Dans ce fichier réalisez un programme qui instancie :
  - Un entier (`int`) initialisé à 42,
  - Un flottant (`float`) initialisé à 12.24,
  - Un booléen (`bool`) initialisé à true,
  - Une chaine de caractères (`std::string`) initialisée à « POLYTECH Dijon ».

- Affichez la valeur de chaque variable dans la console de manière à obtenir le résultat suivant :

```
Integer = 42
Float = 3.14
Boolean = 1
String = POLYTECH Dijon
```

III. PREMIERE FONCTION

- Ajoutez au fichier « main.cpp » une fonction **areEqual** qui prendra en paramètres deux entiers et retournera un booléen indiquant si les deux entiers en paramètre sont égaux.

- Dans la fonction **main**, utilisez la fonction **areEqual** pour comparer la valeur de deux entiers. Le programme affichera le résultat de la comparaison sous la forme suivante :

```
47 == 84 ? Non
```

- Que se passe-t-il si vous déclarez la fonction **areEqual** après la fonction **main** ? Expliquez pourquoi ?

IV. PREMIERE SAISIE

La bibliothèque **iostream** fournit l'objet **cout** pour envoyer de l'information vers la sortie standard (le plus souvent, la console). Elle fournit également l'objet **cin** qui permet de récupérer une saisie clavier (mais vous avez déjà vu cela dans votre cours).

- Modifiez le programme précédent afin que les valeurs des deux entiers comparés soient saisies par l'utilisateur à l'exécution du programme (avec un petit message lui permettant de savoir quoi saisir).

V. PREMIERE BOUCLE FOR

La boucle **for** est utilisée pour exécuter une série d'instructions un certain nombre de fois. Dans l'exemple ci-dessous, la boucle effectuera 10 itérations et affichera les valeurs de 0 à 9 :

```cpp
for (int index = 0; index < 10; ++index)
{
    std::cout << index << std::endl;
}
```

Pour rappel, la boucle for est construite à partir de trois éléments :

```
for ( zone d'initialisation ; condition de continuité ; expression inter-itération )
```

- La zone d'initialisation permet, comme son nom l'indique, d'initialiser des variables avant la première itération de la boucle,
- La boucle effectue ses itérations tant que la condition de continuité est vraie,
- L'expression inter-itération est évaluée à la fin d'une itération, avant que la condition de continuité soit testée.

➢ Modifiez le programme précédent pour procéder à la saisie des entiers et au test d'égalité trois fois.

Bonne pratique

Evitez les nom **`i`**, **`j`**, **`k`** *pour vos indices de boucles* **`for`**. *Privilégiez un nom qui donne davantage d'informations comme* **`iColumn`**, **`iAttempts`**, ...

VI. PREMIERE BOUCLE WHILE

```
while (integer1 == integer2)
{
    std::cout << "Try again !" << std::endl;
}
```

La boucle **`while`** s'exécute tant que la condition de continuité est vérifiée :

```
while ( condition de continuité )
```

➢ Modifiez le programme précédent pour procéder à la saisie des entiers et au test d'égalité tant que les deux entiers saisis sont positifs.

VII. UN CODE UN PEU PLUS PROPRE

La première étape vers un code de qualité est un code bien structuré. Prenez l'habitude de créer de nouveaux fichiers .h et .cpp à chaque fois que vous créez une classe ou un ensemble cohérent de fonctions (bibliothèques ou packages).

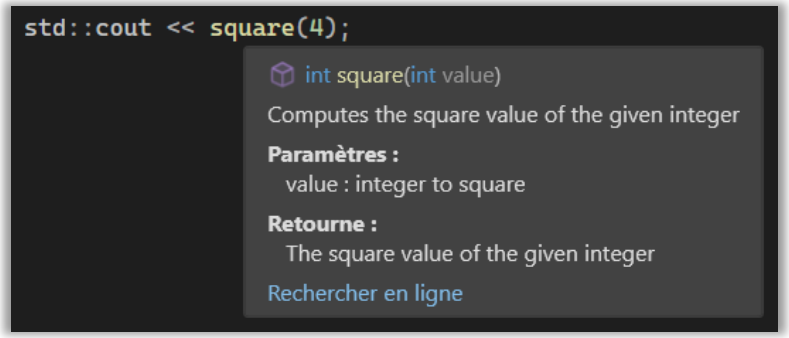
➢ Créer les fichiers « compare.h » et « compare.cpp » qui accueilleront la fonction **`areEqual`**.
➢ Modifiez le fichier « main.h » pour prendre en compte cette modification.

VIII. UN CODE ENCORE PLUS PROPRE

Structurer son code ne suffit pas à le rendre maintenable. Il faut aussi le documenter. Il existe des normes (ex : JavaDoc) dans la manière de commenter un programme qui permettent de rapidement identifier les informations importantes et de générer de la documentation automatique :

```
/**
 * Computes the square value of the given integer
 * @param value : integer to square
 * @returns The square value of the given integer
 */
int square(int value)
{
    return value * value;
}
```

Code commenté de la fonction ***square***



Documentation IntelliSense générée automatiquement par Visual Studio

Prenez l'habitude de documenter votre code en même temps que vous codez plutôt que d'attendre que le programme soit terminé.

Et de manière générale, privilégiez toujours l'anglais pour documenter votre code et nommer vos variables, fonctions et classes. Vous pourrez plus facilement partager votre travail avec d'autres développeurs que ce soit dans le cadre d'un projet Open Source ou d'une collaboration avec un prestataire externe.

- Documentez le fichier « compare.h »

IX. PREMIER TABLEAU C++

En C, comme en C++, les tableaux n'existent pas nativement dans le langage. Lorsque l'on utilise la syntaxe ci-dessous, **tab** représente l'adresse du premier élément d'une suite de 10 entiers contigus en mémoire :

```
int tab[10] = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 }
```

De fait, il n'existe pas de méthode qui, associée à **tab**, permettrait de connaître la taille du tableau.

La bibliothèque standard du C++ fournit une classe **array** qui permet de gérer de manière moderne un tableau.

```
std::array<int, 10> tab = { 5, 6, 1, 3, 0, 8, 4, 2, 7, 9 };

std::cout << tab.size();
```

La syntaxe **std::array<int, 10>** déclare un tableau de 10 entiers.

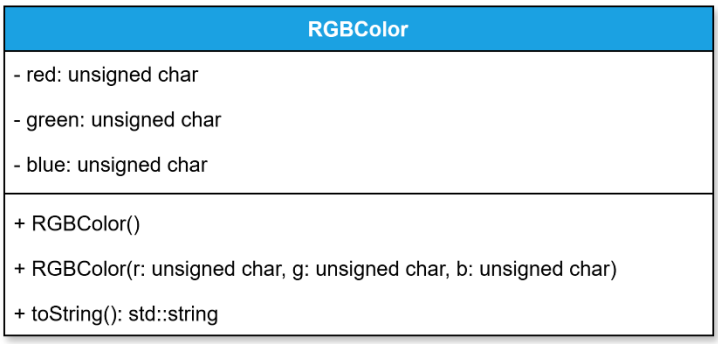
- Importez la bibliothèque « array » dans votre fichier « main.h »
- Déclarez un tableau de 5 éléments initialisé avec des valeurs de votre choix
- Affichez le contenu du tableau à l'aide d'une boucle **for**
- Affichez la moyenne des valeurs contenues dans le tableau

X. LA VIE EN COULEURS

Il existe plusieurs façons de représenter une couleur. Parmi elles, la plus classique en informatique est la représentation RGB (Rouge – Vert – Bleu). La combinaison de trois composantes rouge, vert et bleu permet de recomposer une lumière émise dans la couleur voulue :

Chaque composante est représentée par une valeur entre 0 et 255 stockée sur un octet. Toutes les composantes à 0 représentent le noir, toutes les composantes à 255 représentent le blanc, et au milieu 16 777 214 couleurs sont disponibles.

- Réalisez la classe suivante :



La couleur par défaut est le noir.

La méthode `toString()` renvoie une chaine de caractères représentant la couleur au format hexadécimal (ex : `#0000FF`).

- Testez votre classe en implémentant plusieurs objets représentant des couleurs différentes et en affichant leur valeur au format hexadécimal.