TEST # Exercices C++ pour débutants et intermédiaires

- https://zestedesavoir.com/tutoriels/822/la-programmation-en-c-moderne/
- https://zestedesavoir.com/contenus/beta/822/la-programmation-en-c-moderne/

pedagogie: pas besoin de faire tous les exos. Progressif. Faisable pour la majorite dans wandbox.

3 niveau : refaire les exemples du cours pendant ou juste apres la lecture. Faire des exos similaires apres quelques jours. Faire des problemes plusieurs semaine ou mois.

Résumé, points clés ?

Certains exercices réutilisent le code écrit dans d'autres exercices. C'est indiqué.

- code a completer
- code a corriger (avec message d'erreur, warning ou test fail)

**Questions de cours sur hello world**

level: debutant, question de cours question: Quelle fichier d'en-tête (*header*) faut-il importer (C++20) ou inclure (avant C++20) pour pouvoir utiliser cette fonctionnalité ?

- Qu'est-ce qu'une directive du préprocesseur ? À quoi cela sert ?
- Quelle fonction sert de point d'entrée dans un programme ? (C'est-à-dire la fonction qui sera appelée en première, lors du lancement du programme).
- Quel est le type retourné par la fonction `main` ? À quoi sert la valeur retournée par cette fonction ? Quelle est la syntaxe pour retourner une valeur ? Que signifie la valeur `0` dans ce contexte ?
- Quel est l'espace de noms à utiliser pour les fonctionnalités de la bibliothèque standard ?
- Quelle fonctionnalité de la bibliothéque standard permet d'afficher un message ?
- Quelles sont les **trois** syntaxes possible pour utiliser la fonctionnalité `cout` de l'espace de nom `std` ?
- Quelles sont les **deux** syntaxes pour écrire des commentaires ? Quelle la différence entre ces deux syntaxes ?
- Qu'est-ce qu'un bloc de code ? Quel est la syntaxe pour écrire un bloc de code ?

- Qu'est-ce qui finit habituellement une instruction (*statement*) ? Est-il possible d'avoir plusieurs instructions sur une même ligne ? Est-il possible d'avoir une instruction sur plusieurs lignes ?
- Quelle est la syntaxe pour écrire une chaîne de caractères ?
- Quelles sont les **deux** syntaxes pour faire un retour à la ligne ? Quelle est la différence entre ces deux syntaxes ?
- À quois sert un compilateurs ? (2 choses)
- Quelles sont extensions de fichiers utilisables pour créer des fichiers source et d'en-tête en C++ ? Quelles les extensions qui sont utilisées en pratique ?
- Que signifie l'extension de fichier `.h` ? Et `.cpp` ?
- Quelle est le paradigme de programmation utilisé quand vous faites un *hello world* ? Expliquer brivement à quoi cela consiste.
- etapes de compilation ? Fichier .o ? ligne de commande ? preprocesseur ? edition des liens ? IDE ? etc

### Hello world

tag: debutant

Pour ce chapitre, le seul exercice est de réussir à exécuter un programme "hello world".

Je vous conseille de tester au minimum Wandbox. C'est un outil très utile pour tester des syntaxes simples en C++, pas uniquement pour ces exercices.

### Exercices pour std::cout

tag: debutant, cout

======= escape caracteres, t, n, etc

Certains de ces exercices nécessitent que tu ailles faire des recherches dans la documentation du C++, pour t'habituer à l'utiliser. Tu peux aller consulter les pages suivantes en particulier :

- les parametères de formatage de `std::basic_ostream` : std::ios_base::flags
- les manipulateurs de flux : Input/output manipulators

Sur Windows, il est parfois complexe d'afficher des caractères accentués dans la console. C'est une problématique qui n'est pas spécifique au C++, donc qui n'a pas d'intérêt ici. Si tu as ce problème, écris simplement ton code sans accent.

---

1. Écris le code pour afficher "Bonjour tout le monde".

```
hello world
```

---

2. Écris le code pour afficher :

```
***********
*   C++   *
***********
```

3. Écris le code pour afficher :

```
  ##
 #  #     #      #
#         #      #
#       #####  #####
 #  #     #      #
  ##       #      #
```

4. Corrige le code suivant :

```
include <iostream>

main()
{
    cout < "hello world" < endl
    return 0
}
```

5. Écris le code pour afficher ton nom, ton prénom et ton âge :

```
Nom : Stroustrup
Prenom : Bjarne
Age : 70
```

6. Écris le code pour afficher ton nom, ton prénom et ton âge, aligné à droite. L'aligement devra être respecté, même avec des noms et prénoms différents. Dit autrement, n'utilise pas des espaces pour aligner le texte.

```
Nom :     Stroustrup
Prenom :      Bjarne
Age :             70
```

**Exercices pour les calculs arithmétiques simples**

1. (+) Complète ce code pour calculer l'aire du rectangle :

```
#include <iostream>

int main() {
   constexpr int width { 5 };
   constexpr int height { 7 };
```

```
    constexpr int area { ... }; // écris ton code ici
    std::cout << "L'aire du rectangle est de : " << area << std::endl;
    return EXIT_SUCCESS;
}
```

2. (+) Complète ce code pour calculer l'aire du disque :

```
#include <iostream>

int main() {
    constexpr int radius { 5 };
    constexpr int area { ... }; // écris ton code ici
    std::cout << "L'aire du disque est de : " << area << std::endl;
    return EXIT_SUCCESS;
}
```

2. (++) Complète ce code pour calculer l'année, le mois et le jour et l'heure correspondant au timestamp UNIX simplifié donné. Pour résumer, ce timestamp est un nombre entier qui représente le nombre de jours depuis le 1 janvier 1970.

```
#include <iostream>

int main() {
    constexpr int timestamp { 1'095'379'200 };
    constexpr int year { ... }; // écris ton code ici
    constexpr int month { ... }; // écris ton code ici
    constexpr int day { ... }; // écris ton code ici
    std::cout << "La date est le " << day << " jour du " << month << " mois de l'année " << y
    return EXIT_SUCCESS;
}
```

**Exercices sur les variables** exos de calculs qui melange des float et int, pour le probleme de div entiere vs div reelle exos sur les types et cast

### Exercices sur std::cin

1. Écris le code pour entrer deux nombres entiers, faire leur somme et afficher le résultat.

```
Entre le premier nombre : 4
Entre le second nombre : 3
La somme est : 7
```

### Exercices sur std::string

1. Écris le code pour créer une chaine de caratères qui contient "hello world" et qui l'affiche.

```
hello world
```

2. Écris le code pour créer une chaine qui contient “hello world” et qui l’affiche à l’envers.

```
dlrow olleh
```

**Exercices sur les boucles**

1. Écris le code pour entrer un nombre entier et qui affiche tous les nombres de 0 à ce nombre.

```
Entre le nombre : 5
0
1
2
3
4
5
```

**Exercices sur les conditions**

1. Écris le code pour entrer un nombre entier. Si ce nombre est positif, affiche ce nombre multiplié par 2. Si ce nombre est négatif, affiche le carré de ce nombre.

```
Entre le nombre : 3
6

Entre le nombre : -5
25
```

**Exercices pour les algorithmes de la bibliothèque standard**

2. Écris le code pour créer une chaine qui contient “hello world” puis utilise l’algorithme `std::copy` pour inverser la chaine.

```
dlrow olleh
```

Pareil avec transform ? Autre ?

**Exercices sur les catégories de valeurs** Voici un exercice sur les catégories de valeurs (lvalue, rvalue), la surcharge de fonction avec les lvalue et rvalue references, et le perfect forwarding. En soi, cette serie d’excercices (a faire dans l’ordre) n’est pas complexe si on a compris ces notions (le code a ecrire faire une dizaine de ligne), mais cela permet justement de vérifier si vous avez compris ces notions.

---

1. Ecrire un code qui permet de “voir” les categories de valeur

Modifies le code suivant, de façon a afficher si la fonction f est appélée avec une lvalue ou une rvalue. Il n'est pas nécessaire d'ajouter d'autres include. Il ne faut pas non plus ajouter d'autres fonction que f. (Mais attention, comme dit dans l'introduction, il peut y avoir des surcharges de cette fonction f).

```
#include <iostream>

// declaration de f
void f(...) { ... }

int main() {
    // appel de f avec une lvalue
    f(...);

    // appel de f avec une rvalue
    f(...);
}
```

Et il faut que ce code affiche :

```
f a ete appele avec une lvalue
f a ete appele avec une rvalue
```

2. La transmission

Modifies le code de facon a ce que f soit appellé via une fonction g (sans modifier le code de f, et en remplaçant uniquement f par g dans main)

```
#include <iostream>

// declaration de f
void f(...) { ... }

// declaration de g

void g(...) { f(...); }

int main() {
    // appel de g avec une lvalue
    g(...);

    // appel de g avec une rvalue
    g(...);
}
```

Quel est le problème dans ce qui est affiché ?

3. Perfect forwarding

Corriger le code en utilisant le perfect forwarding de façon a afficher le résultat attendu.

**Enum**

tag: debutant, enum, switch

**Plusieurs énumérations**

tag: debutant, enum

conflit avec plusieurs enums, pour voir les enum class

**Enum et for**

tag: debutant, enum, for