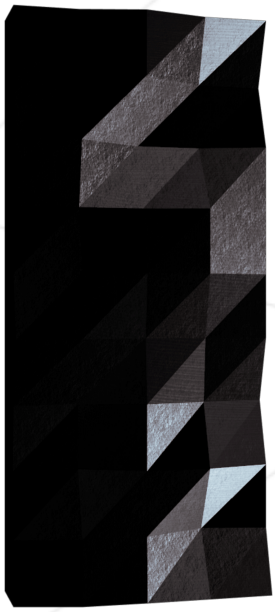

C++ - Module 07

Modèles C++

Résumé:
Ce document contient les exercices du module 07 des modules C++.

Version: 7

Contenu

Chapitre I

Introduction

C++ est un langage de programmation à usage général créé par Bjarne Stroustrup comme une extension du langage de programmation C, ou « C avec classes » (source :Wikipédia).

L'objectif de ces modules est de vous présenter**Programmation orientée objet**. Ce sera le point de départ de votre parcours C++. De nombreux langages sont recommandés pour apprendre la programmation orientée objet. Nous avons décidé de choisir C++ car il est dérivé de votre vieil ami C. Comme il s'agit d'un langage complexe, et afin de garder les choses simples, votre code sera conforme à la norme C++98.

Nous sommes conscients que le C++ moderne est très différent sur de nombreux aspects. Donc si vous voulez devenir un développeur C++ compétent, c'est à vous d'aller plus loin après le 42 Common Core !

Chapitre II

Règles générales

Compilation

- Compilez votre code avecc++et les drapeaux -Mur -Wextra -Werror
- Votre code devrait toujours être compilé si vous ajoutez l'indicateur -std=c++98

Conventions de formatage et de dénomination

- Les répertoires d'exercices seront nommés de cette façon :ex00, ex01, ... , par exemple
- Nommez vos fichiers, classes, fonctions, fonctions membres et attributs comme requis dans les directives.
- Écrivez les noms des classes dans**UpperCamelCase**format. Les fichiers contenant le code de classe seront toujours nommés selon le nom de la classe. Par exemple : NomDeClasse.hpp/NomDeClasse.h, NomDeClasse.cpp,ouNom de classe.tpp.Ensuite, si vous avez un fichier d'en-tête contenant la définition d'une classe "BrickWall" représentant un mur de briques, son nom seraMur de briques.hpp.
- Sauf indication contraire, tous les messages de sortie doivent être terminés par un caractère de nouvelle ligne et affichés sur la sortie standard.
- *Au revoir Nominette !*Aucun style de codage n'est imposé dans les modules C++. Vous pouvez suivre votre style préféré. Mais gardez à l'esprit qu'un code que vos pairs évaluateurs ne peuvent pas comprendre est un code qu'ils ne peuvent pas noter. Faites de votre mieux pour écrire un code propre et lisible.

Autorisé/Interdit

Vous ne codez plus en C. Il est temps de passer au C++ ! Par conséquent :

- Vous êtes autorisé à utiliser presque tout ce qui se trouve dans la bibliothèque standard. Ainsi, au lieu de vous en tenir à ce que vous connaissez déjà, il serait judicieux d'utiliser autant que possible les versions C++ des fonctions C auxquelles vous êtes habitué.
- Cependant, vous ne pouvez pas utiliser d'autres bibliothèques externes. Cela signifie C++11 (et les formes dérivées) etBoosterLes bibliothèques sont interdites. Les fonctions suivantes sont également interdites : *printf(), *alloc()etgratuit().Si vous les utilisez, votre note sera de 0 et c'est tout.

- Veuillez noter que, sauf indication contraire explicite, leen utilisant l'espace de noms <ns_name>et amiles mots-clés sont interdits. Sinon, votre note sera de -42.
- **Vous êtes autorisé à utiliser le STL dans les modules 08 et 09 uniquement.**Cela signifie : non**Conteneurs**(vecteur/liste/carte/et ainsi de suite) et non**Algorithmes**(tout ce qui nécessite d'inclure le <algorithme>(en-tête) jusqu'à ce moment-là. Sinon, votre note sera de -42.

Quelques exigences de conception

- Les fuites de mémoire se produisent également en C++. Lorsque vous allouez de la mémoire (en utilisant le nouveau mot-clé), vous devez éviter**fuites de mémoire**.
- Du module 02 au module 09, vos cours doivent être conçus dans le**Forme canonique orthodoxe, sauf indication explicite contraire**.
- Toute implémentation de fonction placée dans un fichier d'en-tête (à l'exception des modèles de fonction) signifie 0 pour l'exercice.
- Vous devez pouvoir utiliser chacun de vos en-têtes indépendamment des autres. Ainsi, ils doivent inclure toutes les dépendances dont ils ont besoin. Cependant, vous devez éviter le problème de double inclusion en ajoutant**inclure des gardes**. Sinon, votre note sera de 0.

Lis-moi

- Vous pouvez ajouter des fichiers supplémentaires si vous en avez besoin (par exemple pour diviser votre code). Comme ces tâches ne sont pas vérifiées par un programme, n'hésitez pas à le faire à condition de rendre les fichiers obligatoires.
- Parfois, les directives d'un exercice semblent courtes, mais les exemples peuvent montrer des exigences qui ne sont pas explicitement écrites dans les instructions.
- Lisez chaque module dans son intégralité avant de commencer ! Vraiment, faites-le.
- Par Odin, par Thor ! Utilise ton cerveau !!!

Concernant le Makefile pour les projets C++, les mêmes règles qu'en C s'appliquent (voir le chapitre Norme sur le Makefile).

Vous devrez implémenter de nombreuses classes. Cela peut paraître fastidieux, à moins que vous ne soyez capable de créer un script dans votre éditeur de texte préféré.

Vous disposez d'une certaine liberté pour réaliser les exercices. Cependant, respectez les règles obligatoires et ne soyez pas paresseux. Vous passeriez à côté de beaucoup d'informations utiles ! N'hésitez pas à lire les notions théoriques.

Chapitre III

Exercice 00 : Commencer avec quelques fonctions

	Exercice : 00
Commencez avec quelques fonctions	
Répertoire de remise :*ex*00/	
Dossiers à rendre :Makefile, main.cpp, peu importe.{h, hpp}	
Fonctions interdites :Aucun	

Implémentez les modèles de fonctions suivants :

- échanger:Échange les valeurs de deux arguments donnés. Ne renvoie rien.
- minutes:Compare les deux valeurs passées dans ses arguments et renvoie la plus petite. Si les deux sont égales, la deuxième valeur est renvoyée.
- max:Compare les deux valeurs passées dans ses arguments et renvoie la plus grande. Si les deux sont égales, la deuxième valeur est renvoyée.

Ces fonctions peuvent être appelées avec n'importe quel type d'argument. La seule exigence est que les deux arguments doivent avoir le même type et doivent prendre en charge tous les opérateurs de comparaison.

Les modèles doivent être définis dans les fichiers d'en-tête.

Exécution du code suivant :

```
int         principal(vide) {

        intun =2;
        intb =3;

        ::échanger( a, b );
        std::cout <<"a = "<< un <<", b = "<< b << std::endl; std::cout <<"min( a, b )
        = "<< ::min( a, b ) << std::endl; std::cout <<"max( a, b ) = "<< ::max( a, b )
        << std::endl;

        std::string c ="chaine1";
        std::string d ="chaine2";

        ::échanger(c, d);
        std::cout <<"c = "<< c <<", d = "<< d << std::endl; std::cout <<"min( c, d ) =
        "<< ::min( c, d ) << std::endl; std::cout <<"max( c, d ) = "<< ::max( c, d ) <<
        std::endl;

        retour0;
}
```

Devrait afficher :

```
a = 3, b = 2
min(a, b) = 2
max(a, b) = 3
c = chaîne2, d = chaîne1
min(c, d) = chaîne1
max(c, d) = chaîne2
```

Chapitre IV

Exercice 01 : Itérer

	Exercice : 01
Itérer	
Répertoire de remise :*ex*01/	
Dossiers à rendre :Makefile, main.cpp, iter.{h, hpp}	
Fonctions interdites :Aucun	

Implémenter un modèle de fonctionitérerqui prend 3 paramètres et ne renvoie rien.

- Le premier paramètre est l'adresse d'un tableau.
- Le deuxième est la longueur du tableau.
- La troisième est une fonction qui sera appelée sur chaque élément du tableau.

Tournez dans unmain.cppfichier contenant vos tests. Fournissez suffisamment de code pour générer un exécutable de test.

TonitérerLe modèle de fonction doit fonctionner avec n'importe quel type de tableau. Le troisième paramètre peut être un modèle de fonction instancié.

Chapitre V

Exercice 02 : Tableau

	Exercice : 02
Tableau	
Répertoire de remise :*ex*02/	
Dossiers à rendre :Makefile, main.cpp, Array.{h, hpp} et fichier optionnel : Array.tpp	
Fonctions interdites :Aucun	

Développer un modèle de classe**Tableau**qui contient des éléments de typeTet qui implémente le comportement et les fonctions suivants :

- Construction sans paramètre : Crée un tableau vide.
- Construction avec un int non signénen tant que paramètre : crée un tableau denéléments initialisés par défaut.
  Astuce : essayez de compilerint * a = nouveau int();puis afficher *un.
- Construction par opérateur de copie et d'affectation. Dans les deux cas, la modification du tableau d'origine ou de sa copie après copie ne doit pas affecter l'autre tableau.
- Vous DEVEZ utiliser l'opérateurnouveau[]pour allouer de la mémoire. L'allocation préventive (allocation de mémoire à l'avance) est interdite. Votre programme ne doit jamais accéder à la mémoire non allouée.
- Les éléments sont accessibles via l'opérateur d'indice : [ ].
- Lors de l'accès à un élément avec l'opérateur [ ], si son index est hors limites, un std::exceptionest jeté.
- Une fonction membretaille()qui renvoie le nombre d'éléments dans le tableau. Cette fonction membre ne prend aucun paramètre et ne doit pas modifier l'instance actuelle.

Comme d'habitude, assurez-vous que tout fonctionne comme prévu et remettez unmain.cppfichier qui contient vos tests.

Chapitre VI

Soumission et évaluation par les pairs

Remettez votre devoir dans votreGitcomme d'habitude. Seuls les travaux contenus dans votre dépôt seront évalués lors de la soutenance. N'hésitez pas à vérifier les noms de vos dossiers et fichiers pour vous assurer qu'ils sont corrects.

16D85ACC441674FBA2DF65190663F43A243E8FA5424E49143B520D3DF8AF68036E47
114F20A16827E1B16612137E59ECD492E468BC6CD109F65388DC57A58E8942585C8
D193B96732206