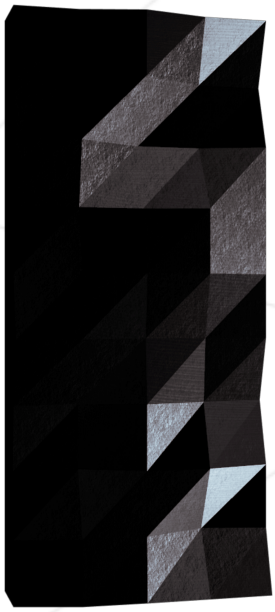

C++ - Module 00

Namespace, class, member functions, stdio stream, initialization lists, static, const, et plein d'autres trucs basiques

Résumé: Ce document contient le sujet du module 00 de la piscine C++ de 42

Chapitre I

Règles Générales

- Toute fonction déclarée dans une header (sauf pour les templates) ou tout header non-protégé, signifie 0 à l'exercice.
- Tout output doit être affiché sur stdout et terminé par une newline, sauf autre chose est précisé.
- Les noms de fichiers imposés doivent être suivis à la lettre, tout comme les noms de classe, les noms de fonction, et les noms de méthodes.
- Rappel : vous codez maintenant en `C++`, et plus en `C`. C'est pourquoi :
  - Les fonctions suivantes sont **INTERDITES**, et leur usage se soldera par un 0 : `*alloc`, `*printf` et `free`
  - Vous avez l'autorisation d'utiliser à peu près toute la librairie standard. CEPENDANT, il serait intelligent d'essayer d'utiliser la version C++ de ce à quoi vous êtes habitués en C, plutôt que de vous reposer sur vos acquis. Et vous n'êtes pas autorisés à utiliser la STL jusqu'au moment où vous commencez à travailler dessus (module 08). Ca signifie pas de Vector/List/Map/etc... ou quoi que ce soit qui requiert une include <algorithm> jusque là.
- L'utilisation d'une fonction ou mécanique explicitement interdite sera sanctionnée par un `0`
- Notez également que sauf si la consigne l'autorise, les mot-clés `using namespace` et `friend` sont interdits. Leur utilisation sera punie d'un `0`.
- Les fichiers associés à une classe seront toujours nommés `ClassName.cpp` et `ClassName.hpp`, sauf si la consigne demande autre chose.
- Vous devez lire les exemples minutieusement. Ils peuvent contenir des prérequis qui ne sont pas précisés dans les consignes.
- Vous n'êtes pas autorisés à utiliser des librairies externes, incluant `C++11`, `Boost`, et tous les autres outils que votre ami super fort vous a recommandé.
- Vous allez surement devoir rendre beaucoup de fichiers de classe, ce qui peut paraitre répétitif jusqu'à ce que vous appreniez a scripter ca dans votre éditeur de code préferé.

- Lisez complètement chaque exercice avant de le commencer.
- Le compilateur est `clang++`
- Votre code sera compilé avec les flags `-Wall -Wextra -Werror -std=c++98`
- Chaque include doit pouvoir être incluse indépendamment des autres includes. Un include doit donc inclure toutes ses dépendances.
- Il n'y a pas de norme à respecter en `C++`. Vous pouvez utiliser le style que vous préferez. Cependant, un code illisible est un code que l'on ne peut pas noter.
- Important : vous ne serez pas noté par un programme (sauf si précisé dans le sujet). Cela signifie que vous avez un degré de liberté dans votre méthode de résolution des exercices.
- Faites attention aux contraintes, et ne soyez pas fainéant, vous pourriez manquer beaucoup de ce que les exercices ont à offrir
- Ce n'est pas un problème si vous avez des fichiers additionnels. Vous pouvez choisir de séparer votre code dans plus de fichiers que ce qui est demandé, tant qu'il n'y a pas de moulinette.
- Même si un sujet est court, cela vaut la peine de passer un peu de temps dessus afin d'être sûr que vous comprenez bien ce qui est attendu de vous, et que vous l'avez bien fait de la meilleure manière possible.

Table des matières

Chapitre II

Exercice 00 : Megaphone

	Exercice : 00
Exercice 00 : Megaphone	
Dossier de rendu : $ex00/$	
Fichiers à rendre : `Makefile, megaphone.cpp`	
Fonctions interdites : `Aucune`	

Afin d'être sûr que tout le monde est bien reveillé, écrivez un programme dont le comportement est le suivant :

```
$>./megaphone "shhhhh... I think the students are asleep..."
SHHHHH... I THINK THE STUDENTS ARE ASLEEP...
$>./megaphone Damnit " ! " "Sorry students, I thought this thing was off."
DAMNIT ! SORRY STUDENTS, I THOUGHT THIS THING WAS OFF.
$>./megaphone
* LOUD AND UNBEARABLE FEEDBACK NOISE *
$>
```

Chapitre III

Exercice 01 : My Awesome PhoneBook

	Exercice : 01
Exercice 01 : My Awesome PhoneBook	
Dossier de rendu : *ex*01/	
Fichiers à rendre : `Makefile, *.cpp, *.{h, hpp}`	
Fonctions interdites : `Aucune`	

Bienvenue dans les années 80 et leur technologie incroyable ! Écrivez un incroyable programme qui se comporte comme un annuaire `nul`.
Prenez le temps de donner à votre exécutable un nom pertinent. Lorsque le programme démarre, il demande l'input de l'utilisateur : vous devez accepter la commande `ADD`, la commande `SEARCH` ou la commande `EXIT`. Toute autre entrée est supprimée.

Le programme commence vide (pas de contacts), n'utilise pas d'allocation dynamique et ne peut pas stocker plus de 8 contacts. Si un neuvième contact est ajouté, veuillez définir un comportement pertinent.

http://www.cplusplus.com/reference/string/string/ et bien sûr
http://www.cplusplus.com/reference/iomanip/

- Si la commande est `EXIT` :
  - Le programme se ferme et les contacts sont perdus à jamais
- Sinon, si la commande est `ADD` :
  - Le programme invitera l'utilisateur à saisir de nouvelles informations de contact, un champ à la fois, jusqu'à ce que chaque champ soit rempli.
  - Un contact contient les champs suivants : `first name`, `last name`, `nickname`, `login`, `postal address`, `email address`, `phone number`, `birthday date`, `favorite meal`, `underwear color` et `darkest secret`.
  - Un contact DOIT être représenté comme une instance de classe dans votre code. Vous êtes libre de concevoir la classe comme vous le souhaitez, mais l'évaluation vérifiera la cohérence de vos choix. Allez regarder les vidéos à nouveau si vous ne comprenez pas ce que je veux dire (et je ne veux pas dire "utiliser tout" avant que vous ne le demandiez).
- Enfin, si la commande est `SEARCH` :
  - Le programme affichera une liste des contacts disponibles non vides dans 4 colonnes : `index`, `prénom`, `nom de famille` et `pseudo`.
  - Chaque colonne doit avoir une largeur de 10 caractères, le contenu doit être aligné à droite, et chaque colonne séparée par le caractère `'|'`. Toute sortie plus longue que la largeur des colonnes est tronquée et le dernier caractère affichable est remplacé par un point (`'.'`).
  - Ensuite, le programme demandera à nouveau pour l'index de l'entrée souhaitée et affichera les coordonnées du contact, un champ par ligne. Si l'entrée n'a aucun sens, définir un comportement pertinent.
- Sinon, l'input est ignoré. Lorsqu'une commande a été correctement executée, le programme attend une nouvelle commande `ADD` ou `SEARCH` et il se termine si la commande `EXIT` est utilisée.

Chapitre IV

Exercice 02 : The Job Of Your Dreams

	Exercice : 02
Exercice 02 : The Job Of Your Dreams	
Dossier de rendu : *ex*02/	
Fichiers à rendre : `Account.class.cpp`	
Fonctions interdites : `Aucune`	

`Cet exercice ne rapporte pas de points, mais demeure interessant dans le cadre de votre piscine. Vous n'êtes pas obligés de le faire.`

C'est votre premier jour de travail sur `GlobalBanksters United`. Vous avez passé avec succès les tests d'embauche de l'équipe de développement grâce à quelques astuces sur `Microsoft Office` qu'un ami vous a montré. Mais vous savez que c'était votre rapide installation de `Adobe Reader` qui a vraiment impressionné le recruteur. Cela vous a donné le petit avantage nécessaire pour battre vos adversaires pour ce travail.

Quoi qu'il en soit, vous l'avez fait et votre patron vous a confié votre première tâche. Il vous a expliqué que quelqu'un a perdu le code source qui gère le comptes des clients de la banque. Votre patron a décidé de recompiler et redémarrer le programme pour récupérer le fichier. Votre prédécesseur a été remercié parce qu'il a essayé d'expliquer au patron comment faire son travail : il a mentionné que "ce n'est pas comme ça que ça doit fonctionner", "tu as supprimé ce fichier, idiot !" et "on aurait dû utiliser Git comme je l'avait dit". Quel petit arrogant, n'est-ce pas ?

Après une quarantaine de minutes de plainte sur le manque d'expertise de votre prédécesseur, vous êtes chargé de l'écriture du fichier source manquant pour demain. Votre patron aurait adoré le faire tout seul, mais vous savez, il a des tâches de manager à ac-

complir. Il vous a donc envoyé le fichier `Account.class.hpp` par email, protégé par une clé `gpg`, ainsi que sa clé privée. "La sécurité est importante, les hackers peuvent frapper n'importe où", a déclaré votre patron en guise de conclusion lorsque vous quittez son bureau.

Mal à l'aise dans votre costume et transpirant abondamment, vous passez devant l'imprimante et le photocopieur dans le long couloir vers l'espace libre pour trouver votre bureau. Vous êtes dans la zone centrale et tout le monde peut voir votre écran. Vous remarquerez également le grand panneau sur le mur qui indique de manière joyeuse et colorée "Les casques empêchent l'esprit d'équipe ! Le silence améliore l'esprit d'équipe !". La journée va être longue.

Une fois assis à votre bureau, vous remarquez la fenêtre de connexion `Windows XP` sur votre écran. Personne ne vous a donné de login/password ce matin, alors vous demandez très poliment à votre voisin s'il sait où se les procurer. Sa réponse, "Pour les questions, on écrit un billet, tout le monde le sait, DUH !", Ne vous aide pas vraiment. De plus, vous imaginez bien que demain, vous découvrirez que vous êtes également responsable des tickets car vous semblez être le seul employé IT du bâtiment. Sans rien perdre, vous essayez `admin/admin` dans la fenêtre de connexion. L'ordinateur se déverrouille et au bout d'une minute environ, les icônes apparaissent et vous pouvez utiliser l'ordinateur.

Ensuite, vous passez quelques heures à comprendre les choses. Vous comprenez que `PuTTy` vous permet de `ssh` en tant que root sans mot de passe sur un serveur quelque part. Le serveur est un ancien serveur Ubuntu et semble exécuter la plupart des services de l'entreprise. Vous pouvez créer votre compte avec des identifiants pertinents pour enfin arrêter de vous connecter en root et vous bloquez le ssh root. Vous remarquez également que vous n'avez pas de compte de messagerie et que, par conséquent, vous ne recevrez jamais l'e-mail contenant le fichier joint de votre patron.

Grâce à votre `bash-fu`, vous pouvez enfin localiser les sources du projet que vous êtes censé fixer dans un dossier nommé `"test2_REAL_ONE_DONT_DELETE/"`. Le dossier contient quelques fichiers écrits entre 1989 et 1992 par un certain Brad MacLane. Le fichier `Account.class.hpp` est présent et une compilation rapide confirme qu'un fichier `Account.class.cpp` est manquant. Il existe également un fichier avec quelques tests et un ancien logfile qui semble contenir la sortie correspondante.

Ensuite, vous écrivez rapidement environ 150 lignes de `C ++` génial et après quelques compilations échouées, votre programme compile et passe les tests avec une sortie parfaite, à l'exception des timestamps. Bon sang, vous êtes fort ! Mais vos problèmes vous rattrapent lorsque vous entendez votre patron crier dans le couloir : "Qui a foiré avec le serveur de production ? ! Je ne peux plus me connecter !"

Bonus: Ajoutez un attribut à la classe qui compte le nombre d'appels de "int checkAmount(void) const;". Faites cela sans changer quoi que ce soit au prototype de cette fonction. Vous avez besoin d'un nouveau mot-clé, et cette situation est parfaite pour le décrouvrir. Bien entendu, ce mot-clé n'est pas dans les vidéos.