C++ - 2023-2024 Feuille TP n°1
Initiation à Code::Blocks – Variables – Affichages

NB : LES CODES SOURCES UTILES POUR CE TP SONT DISPONIBLES SUR MOODLE

Objectifs

1. Découvrir un environnement de programmation
2. Compiler et exécuter des programmes simples
3. Respecter la syntaxe de C++ pour écrire des programmes simples
4. Tester et corriger un programme

Consignes générales pour les exercices sur machine

a) Sur votre espace réseau personnel de l'IUT, créez un répertoire réservé à vos exercices de programmation (que vous appellerez par exemple C++), dans lequel vous aurez un répertoire par feuille de TP (par exemple un répertoire TP1 pour le TP1, etc..).

b) Téléchargez tous les fichiers qui se trouvent dans l'espace Moodle correspondant au TP1 et copiez les dans votre dossier TP1.

c) N'oubliez pas de sauvegarder votre travail sur votre espace réseau de l'IUT : il sera ainsi accessible depuis n'importe quel poste (y compris chez vous).

TRAVAIL à RENDRE : vous devrez compléter au fur et à mesure que vous avancez dans les exercices la fiche réponse ficheTP1.docx et la déposer à la fin de la séance sur l'espace Moodle (au format docx et au format pdf). Ne perdez pas de temps dans la mise en page : vos réponses doivent précises et succinctes.

Exercice 1 : Première utilisation de Code::Blocks

a) Si ce n'est pas encore fait, créez dans votre compte un répertoire C++ dans lequel vous créerez un sous-répertoire TP1.

b) Lancez l'environnement de programmation Code::Blocks depuis le menu Démarrer.
Cet environnement, comme beaucoup d'autres, fonctionne par **projets**. Il faudra donc dorénavant créer un projet pour chaque exercice.

c) Nous allons voir dans un premier temps la démarche à suivre pour créer un projet prêt à l'emploi (à utiliser donc lorsque vous avez tous les programmes à saisir vous-mêmes).
Cliquez sur "***Create a new project***" puis sur "***Console Application***", puis "***C++***".
Validez simplement la fenêtre suivante.

d) Donnez un nom à votre projet (par exemple exo1), **vérifiez son emplacement** (dans notre exemple il doit être dans C++\TP1) et validez.
Un sous-répertoire de même nom que le projet est automatiquement créé.
Cliquez enfin sur "***Finish***". Le projet s'ouvre avec un fichier main.cpp généré automatiquement (cliquer sur le '+' du répertoire « Sources » pour le visualiser).

e) Compilez le projet en cliquant sur l'icône en forme d'engrenage jaune en haut à gauche, ou en sélectionnant "***Build***" dans le menu **Build**, ou encore en tapant **Ctrl-F9**.
Deux sous-répertoires se sont encore ajoutés à l'architecture : un pour les fichiers objets (obj) et un pour les fichiers exécutables (bin). **Recherchez et notez leurs emplacements dans la fiche réponse.**

f) Exécutez le projet en cliquant sur l'icône suivant en forme de triangle vert, ou en sélectionnant "***Run***" dans le menu **Build**, ou encore en tapant **Ctrl-F9**.
Remarque : on peut compiler puis exécuter avec une seule commande (***Build and run***, ou **F9**, ou la troisième icône).

g) Modifiez le programme pour qu'il affiche non plus "**`Hello world`**" mais un message de votre choix.
Recompilez (cela sauvegarde automatiquement les modifications) et exécutez à nouveau.
Avant de passer à l'exercice suivant, faites un clic droit sur le nom de votre projet et sélectionnez "***Clean***". **Cela effacera les fichiers exécutables et objet** et libérera ainsi de l'espace disque.

h) Enfin, vous pouvez fermer le projet en cliquant droit sur son nom et en sélectionnant "***Close project***". Cela n'est pas obligatoire mais avoir plusieurs projets ouverts simultanément peut être troublant.

Exercice 2 : Projet depuis un fichier existant

Parfois (voire assez souvent en TP) vous aurez à recopier un fichier déjà saisi pour commencer un projet. Dans ce cas vous n'avez pas besoin du fichier `main.cpp`, il constitue même une gêne. La création du projet va donc varier légèrement.

a) Créez un nouveau projet (Menu → File → New → Project) et choisissez "***Empty Project***". Ensuite, donnez-lui un nom comme précédemment, et continuez jusqu'à "***Finish***".
Comme auparavant, un nouveau répertoire a été créé (dans mon exemple il s'appellera exo2)

b) Copiez le fichier **`erreur.cpp`** fourni dans ce nouveau répertoire (dans l'exemple C++\TP1\exo2)

c) Ajoutez ensuite le fichier au projet. Pour cela faites un clic droit sur son nom et sélectionnez "***Add files***", puis choisissez le fichier à copier et "***Select All***" pour qu'il soit bien ajouté à l'exécutable.

d) Commencez par rendre le programme lisible, c'est-à-dire :
   - une seule instruction par ligne
   - en respectant l'indentation (on décale à gauche quand on ouvre un bloc, et à droite à sa fermeture).

   Vous pouvez le faire soit manuellement, soit en sélectionnant "***Source code formater***" dans le menu "***Plugins***" (attention, la deuxième solution peut laisser des lignes étranges).

e) Compilez. Que constatez-vous ?

   Corrigez les erreurs de compilation ***une par une*** en interprétant les messages d'erreur obtenus, sauvegardez et recompilez, jusqu'à disparition des erreurs de compilation.

f) Exécutez le programme. Que constatez-vous ? Est-ce que tout se passe comme le vouliez ? Un code source peut donc être transformé en exécutable (étape de compilation réussie avec succès), mais avec une exécution qui ne correspond pas à l'attente de l'utilisateur. Modifier le programme pour qu'il produise le résultat attendu.

Exercice 3 : Portée des variables

a) En procédant de la même façon qu'à l'exercice 2, créez un nouveau projet vide pour y recopier le programme du fichier **`variables.cpp`** fourni.

b) Compilez : que se passe-t-il ? Pourquoi ?

c) Dans quel(s) bloc(s) pouvez-vous déclarer la variable n ? Pourquoi ?

d) Déclarez n de type int dans un de ces blocs et compilez puis exécutez le programme. Comment expliquez-vous la valeur affichée ?

e) Déclarez n dans le bloc b1, et affectez lui la valeur 3 dans ce même bloc. Compilez et exécutez.

Exercice 4 : À vous la main

Créez un projet "Console application" comme à l'exercice 1, et modifiez le programme pour qu'il demande à l'utilisateur de saisir son nom puis affiche le message **`"Bonjour ... "`** en utilisant le nom de l'utilisateur.

Pour pouvoir exécuter votre programme en dehors de Code::Blocks il y a une manipulation à réaliser : aller dans les paramètres de compilation (menu Settings) et sélectionner les 2 librairies statiques libgcc et libstdc++. Activer les librairies statiques ralentit énormément la compilation, vous pourrez les désélectionner après cet exercice.

Vous devez pouvoir maintenant exécuter le programme en dehors de l'environnement Code::Blocks. Dans quel répertoire l'exécutable a-t-il été enregistré ? Quel est son nom ?
Testez en l'exécution à l'extérieur de Code::Blocks.

NB : Sous Windows, la fenêtre d'exécution se ferme dès la fin du programme. Pour garder cette fenêtre ouverte ajoutez l'instruction `system("PAUSE");` juste avant l'instruction return (nécessite l'inclusion : `#include <cstdlib>`). Attention : pas compatible avec Unix.

Exercice 5 : Premier travail à rendre

Écrivez un programme qui lit deux variables de type **double**, et affiche leurs valeurs, puis échange leurs contenus et affiche de nouveau leurs valeurs. Testez.
Recopiez le code dans la fiche réponse, en indiquant les valeurs que vous avez utilisées pour en tester l'exécution.

Exercice 6 : Vous avez dit bizarre ?

Un programme qui compile n'est pas forcément un programme correct. Il peut subsister des erreurs logiques que le test syntaxique ne peut pas détecter.

En procédant de la même façon qu'à l'exercice 2, créez un nouveau projet vide et ajoutez-lui le code du fichier `bizarre.cpp` fourni.

a) Compilez et exécutez le programme avec les valeurs 10 et 3, puis 10 et 0. Ne pas lancer le débogueur. Résultat ?

b) Modifiez le type de la variable **`x`** de **int** à **double** et exécutez-le de nouveau avec les mêmes valeurs. Résultat ?

Exercice 7 : Quelle est la température moyenne ?

a) Écrivez un programme qui demande à l'utilisateur de saisir 2 températures (utiliser des variables de type **int)** observées dans une journée (par exemple une pour le matin, l'autre pour le soir) puis qui calcule et affiche la moyenne de la température de la journée.

b) Modifiez ensuite le code pour tenir compte du fait qu'on relève une température de plus dans la journée.

c) Modifiez votre code pour ne déclarer qu'une seule variable pour saisir toutes les températures observées. Vous pouvez bien entendu garder en plus la variable pour calculer la moyenne.

d) Placez en commentaire à la fin de votre code la liste des essais que vous avez effectués pour tester complètement la validité de votre programme.

e) Recopiez le code dans la fiche réponse. Déposez-le également (uniquement le fichier `main.cpp`) sur la plateforme Moodle. **Ne pas déposer le projet complet.**