C++, Feuille de TP n°3 Boucles (2 séances)

NB : Les codes sources utiles pour ce TP sont disponibles sur Moodle

Objectifs

1. Se familiariser avec la syntaxe des différentes boucles.
2. Comprendre quand utiliser chacune des boucles.
3. Découvrir des usages classiques des boucles.

Consignes générales pour les exercices sur machine

1. Lire entièrement les questions avant de coder.
2. Préparer un jeu d'essai avant de coder.
3. Écrire ou modifier un programme en s'inspirant des exemples du cours, des TD et TP passés.
4. Tester en utilisant le jeu d'essai. Vos tests doivent être en commentaire à la fin de vos fichiers.

TRAVAIL à RENDRE : l'exercice 2, code source et jeu d'essais complet à la fin de le première séance et l'exercice 5 pour la seconde séance.

Exercice 1 : Les diviseurs

Écrire un programme qui saisit un entier **n** supposé strictement positif (pas de vérification) et affiche la liste de ses diviseurs. Indication : on utilise une boucle **pour** pour parcourir chaque nombre **i** entre **2** et **n-1** et on teste si **i** est diviseur de **n**. Rappel : un nombre **i** est diviseur d'un nombre n **si n%i est égal à 0**. Tester.

En ajoutant une variable, compléter le programme pour qu'il affiche à la fin le nombre de diviseurs de **n** (ne pas compter 1 et n). Tester.

Exercice 2 : Un peu de dessin

1) Écrire un programme pour saisir un entier **n** et affiche **n** étoiles. Exemple d'exécution :

Affichage: Combien d'etoiles ?
Saisie : 7
Affichage : *******

2) Compléter le programme pour vérifier que **n** est strictement positif et demander à l'utilisateur de ressaisir **n** en cas d'erreur. Exemple d'exécution :

Affichage: Combien d'etoiles ?
Saisie : -2
Affichage : Erreur. Veuillez saisir un nombre strictement positif
Affichage: Combien d'etoiles ?
Saisie : 4
Affichage : ****

3) Modifier le programme pour saisir un nombre de lignes **nbl** et afficher un rectangle de **nbl** lignes de **n** étoiles. Penser à vérifier que **nbl** est strictement positif et redemander la saisie en cas d'erreur. Exemple d'exécution :

Affichage:	Combien d'etoiles ?
Saisie :	7
Affichage:	Combien de lignes ?
Saisie :	4
Affichage :	*******
	*******
	*******
	*******

4) Modifier le programme pour permettre d'afficher un rectangle avec un caractère de son choix : saisir le caractère **unCar**, et afficher **nbl** lignes de **n** caractères **unCar**.

5) (**optionnelle pour les plus rapides)** Écrire un programme qui, après avoir saisi un nombre n affiche un triangle de base n. On vérifiera que n est impair (affichage d'un message d'erreur si n est pair). Exemple avec n=5.

```
*
* * *
* * * * *
```

Conseil Déclarer 3 variables : une pour le nombre de lignes (calculé en fonction de la base : n/2+1), une pour le nombre d'espaces en début de ligne, et une pour le nombre de caractères à afficher sur la ligne courante.

Déposer le code de la question 4 ou 5 sur Moodle en incluant le jeu d'essais.

Exercice 3 : Corriger les erreurs

Un étudiant a réalisé le programme C++ suivant, pour calculer sa moyenne (somme de notes / nombre de notes). Son programme ne lui donne pas satisfaction :

Dans certains cas, une boucle infinie se produit.

De plus, la moyenne qu'il obtient est fausse.

Pour aider à le corriger :

- ✓ copier le fichier erreur.cpp dans votre répertoire personnel.
- ✓ tester le programme.
- ✓ trouver quand la boucle infinie se produit, puis corriger l'erreur.
- ✓ trouver quand les résultats sont faux, puis corriger l'erreur ou les erreurs.
- ✓ commenter le programme afin que l'étudiant repère bien les différentes étapes du traitement effectué.

```
#include <iostream>
using namespace std;
int main()
{
   float note, totalnote, nbnotes, moyenne;
   char rep;

   do {
      totalnote = 0;
      nbnotes = 0;
      cout << "Saisir une note" << endl;
      cin >> note;
      while (note < 0 || note > 20) {
         cout << "Erreur : La note doit etre entre 0 et 20" << endl;
         cout << "Recommencez la saisie" << endl;
      }
      totalnote = totalnote + note;
      nbnotes = nbnotes + 1;

      cout << "Y a-t-il d'autres notes a saisir ?" << endl;
      cout << "(repondre par o ou n)" << endl;

      cin >> rep;
   } while (rep == 'o');

   moyenne = totalnote / nbnotes;
   cout << "La moyenne est " << moyenne << "." << endl;

   return 0;
}
```

Exercice 4 : Traitement dans un intervalle de valeurs

1) Ecrire un programme qui demande à l'utilisateur de saisir un entier max strictement positif et affiche tous les entiers divisibles par 7 inférieurs à max
2) Modifier votre programme pour qu'il demande aussi la saisie d'un entier min strictement positif (vérifiez que min<max) et qu'il n'affiche ensuite que les entiers divisibles par 7 compris entre min et max (bornes comprises).
3) On souhaite rendre l'affichage plus clair. Modifier votre programme pour qu'il n'affiche pas plus de 10 valeurs par ligne.

Exercice 5 : (à déposer sur Moodle) Jeu du pendu mathématique

Ecrire un programme C++ pour réaliser le jeu suivant : le joueur doit trouver un nombre compris entre 1 et 100 (généré par l'ordinateur de façon aléatoire). À chaque essai, le joueur obtient les indications suivantes :

- le nombre proposé est un multiple du nombre à deviner
- le nombre proposé est un diviseur du nombre à deviner
- le nombre proposé n'a rien à voir avec le nombre à deviner

Compléter le programme pour limiter le nombre d'essais à 10 (dans les cas a) et b) l'essai n'est pas compté). Le programme indiquera alors également le numéro de son essai et le nombre d'essais restants.

Pour générer un nombre aléatoire **alea** compris entre 1 et 100 en C++ :

```
#include <cstdlib>  // pour srand et rand
#include <ctime>          // pour time
srand(time(NULL));//Initialisation du générateur à partir de l'heure courante
alea = rand()%100 + 1;
```

Déposer le code (incluant le jeu d'essais en commentaire) sur Moodle.

Conserver le code qui vous sera utile pour un prochain TP sur les sous-programmes et pour le projet.

Exercice 6 : Compter le nombre de chiffres d'un entier

Etant donné un entier positif **nb**, on veut écrire un programme pour compter le nombre de chiffres qui le composent. La méthode consiste à diviser **nb** par 10 tant qu'on n'obtient pas zéro.

Jeu d'essais proposé :		Résultat attendu
Essai 1 : 12345	*Cas général*	5
Essai 2 : 1	*Un seul chiffre*	1
Essai 3 : 100	*Le nombre est un multiple de 10*	3
Essai 4 : 0	*zéro*	**1**

Exercice 7 : (optionnel) Recherche de nombre premier

Écrire un programme qui permet de déterminer si un nombre entier supérieur à 2, saisi au clavier par l'utilisateur, est premier :

1) Pour cela, le diviser par tous les entiers qui lui sont inférieurs. Si aucune division n'a un reste nul, le nombre est premier.

On peut limiter le nombre de division en contrôlant si le nombre est pair. S'il l'est, il n'est pas premier ; s'il ne l'est pas, il est inutile de le diviser par les multiples de 2. On peut arrêter les divisions lorsque le diviseur est supérieur à la moitié du nombre.

Exercice 8 : (optionnel) Recherche de PGCD

Ecrire un algorithme qui permet de trouver le PGCD (plus grand commun diviseur) de deux nombres entiers par l'algorithme d'Euclide :

- Faire une division entière du plus grand nombre par le plus petit.
- Remplacer le plus grand par le plus petit, le plus petit par le reste de la division entière.
- Continuer jusqu'à ce que le reste soit nul. Le PGCD est le dernier reste non nul.