INF1005C - PROGRAMMATION PROCÉDURALE

Travail dirigé No. 2
Programmes simples
Entrées et sorties

Objectifs : Permettre à l'étudiant de faire ses premiers pas de programmation en langage C++.
Il apprendra à manipuler la structure de base d'un programme, les types de base ainsi que les entrées et les sorties du C++.
Durée : Une séance de laboratoire.
Remise du travail : 15 mai 2019, avant 23 h 30.
Travail préparatoire : Lecture du guide de codage, des exercices, et rédaction des algorithmes.
Directives : N'oubliez pas de mettre les entêtes de fichiers et de respecter le guide de codage (voir la dernière page de ce document pour les points à respecter).
Documents à remettre : Sur le site Moodle des travaux pratiques, vous remettrez l'ensemble des *fichiers .cpp* compressés dans un fichier .zip en suivant **la procédure de remise des TDs.**

Directives particulières

- Affichez toujours un message d'invite avant chaque saisie.
- Vous n'avez pas à valider les entrées.
- Vous n'avez pas à afficher les caractères accentués.
- Vous pouvez déclarer toutes les variables désirées.
- Utilisez le type **string** pour les chaînes de caractères.
- Pour chacun des exercices, utilisez des messages appropriés lors de l'affichage. Des chiffres seuls ne suffisent pas.
- Vous ne pouvez **pas utiliser** de structures de **décision** (incluant le `?:` qui n'est pas vu dans le cours) ni de structures de **répétition**. Indiquez en commentaire les endroits où il aurait été bien d'utiliser ces structures.

1. **Jouer au prof.** Pour ce numéro, vous devez corriger et faire compiler le programme *exo2.cpp* dans les fichiers joints tout en gardant son utilité principale, c'est-à-dire demander un prix à payer, une somme d'argent donnée pour payer (tout en entier) et affiche la monnaie à rendre en billets de 100$, 20$, 10$, 5$ et en pièces de 1$. Vous devez aussi le corriger au niveau du style en suivant le guide de codage.

2. **Texte à trous.** Demander à l'utilisateur d'entrer trois noms d'animaux, trois adjectifs masculins et trois noms de fruits. Afficher ensuite la phrase «Aujourd'hui, j'ai vu un <animal> s'emparer d'un panier <adjectif> plein de <fruit>.» en sélectionnant aléatoirement l'animal, l'adjectif et le fruit à partir de ceux donnés par l'utilisateur. Ce numéro requiert l'utilisation de tableaux. La syntaxe pour la déclaration est : *type_des_éléments nom_de_la_variable_tableau***[***nombre_de_cases***];**
   La syntaxe pour l'accès est la même qu'au TD1, soit : *nom_de_la_variable_tableau***[***indice***]**

   Exemple :
   ```
   Entrer trois noms d'animaux : chevreuil chien pigeon
   Entrer trois adjectifs masculins : rouge officiel lourd
   Entrer trois noms de fruit : pommes kiwis bananes

   Aujourd'hui, j'ai vu un pigeon s'emparer d'un panier rouge plein de kiwis.
   ```

3. **Affichage formaté.** Écrire un programme qui génère aléatoirement un nombre réel entre 1000.0 et 999999.99 puis l'afficher avec deux chiffres décimaux et un espace séparant les milliers des centaines. Servez-vous de `RAND_MAX` pour générer un nombre réel.

Exemple (si on a généré 456846.8945) :

```
Le nombre obtenu : 456 846.89
```

4. **Utilisation de fichiers.** Le fichier *ventes.txt* contient 4 lignes. La première est le nom complet d'un client et les trois autres contiennent le nom (sans espaces) d'un produit, la quantité vendu et le prix unitaire. Il faut écrire un programme qui lit le contenu de *ventes.txt* et écrit la facture dans un fichier *facture.txt* avec le nom du client, le sous-total avant taxe, le montant des taxes et total avec taxes, supposant un taux de 15% de taxes.

   Les montants dans la facture doivent être écrit sous le nom du client, être alignés à droite sur 10 colonnes et avoir deux chiffres décimaux suivi d'un signe de dollar. Il faut suivre le format de l'exemple.

   Exemple

ventes.txt

```
Joe Untel
chaussures 1 249.99
chandail   3 49.99
casquette  1 34.99
```

facture.txt

```
Joe Untel
SOUS-TOTAL     434.95 $
TAXES           65.24 $
TOTAL          500.19 $
```

Les points du **guide de codage** à respecter **impérativement** pour ce TD sont les suivants :
(lire le guide de codage sur le site Moodle du cours pour la description détaillée de chacun de ces points)

3: noms des variables en lowerCamelCase
4: noms des constantes en MAJUSCULES
25: is/est pour booléen
27: éviter les abréviations (les acronymes communs doivent être gardés en acronymes)
29: éviter la négation dans les noms
33: entête de fichier avec vos noms et matricules
42: #include au début (mais après l'entête)
46: initialiser à la déclaration
47: pas plus d'une signification par variable
62: pas de nombres magiques dans le code
63-64: « double » toujours avec au moins un chiffre de chaque côté du point
79,81: espacement et lignes de séparation
85: mieux écrire le programme plutôt qu'ajouter des commentaires
87: préférer //

Plusieurs points du guide montrent comment bien indenter, mais comme dans ce TD la seule construction qui demande une indentation est la définition du « main », la forme du programme devrait être :

```
#include ...
Tous les débuts de lignes
sont alignés à gauche
...
int main()
{
    Tous les débuts de lignes
    sont alignés avec
    une seule indentation (tabulation)
    ...
}
```