Université Laval Département d'informatique et de génie logiciel
Faculté des sciences et génie

GIF-1003
PROGRAMMATION AVANCÉE EN C++

Laboratoire #3

Manipulation des tableaux, des chaînes de caractères et des fichiers, programmation modulaire

Exercice #1 Tableaux

Voici un algorithme qui lit une liste de nb nombres entiers, calcule et affiche la moyenne puis l'écart entre chaque note et cette moyenne. Vous devez traduire cet algorithme en un programme C++. Vous devez faire une fonction pour chaque bloc composant votre algorithme.
Important: votre programme doit être impérativement constitué en un fichier principal.cpp, fonctions.cpp et fonctions.h.

Constante connue : MAX : la taille maximum de la liste

Bloc B1: Calcul autour d'une série de nombres entiers
{**description** : calcul de la moyenne d'une série de nombres entiers saisis ainsi que de l'écart entre chaque note avec la moyenne calculée par manipulations arithmétiques simples
entrée: ---
sortie: ---
résultat : affiche le résultat des calculs demandés
hypothèse : nb est un entier positif inférieur ou égal à MAX, la liste contient nb éléments entiers (cardinalité de la liste)}
Début
    Saisie des données (Voir Bloc B 2)
    {A : tableau contient nb éléments entiers}
    Calcul de la moyenne (Voir bloc B3)
    {A : moyenne contient la valeur moyenne des éléments de tableau}
    Affichage des résultats à l'usager(Voir bloc B4)
    {A: la moyenne ainsi que l'écart entre chaque note et la moyenne sont affichées}
Fin
Bloc B2: Saisie des données
{**description** : Saisie de données
entrée: ---
sortie: nb, un tableau contenant nb nombres entiers
résultat : ---
hypothèse : nb est un entier positif inférieur ou égal à MAX, les nb éléments du tableau sont des entiers.}
Début
    Demander nb
    {A : nb est un entier > 0 et < = MAX}
    Répéter $\forall$ i $\in$ [1 , nb]
    Début
    Demander tableau[i]
    Fin
    {A : tableau contient nb entiers}
Fin

Bloc B3: Calcul de la moyenne

{**description** : calcul de la moyenne d'une série d'entiers : on calcule la somme de tous les éléments de la liste et on divise par le nombre d'élément
entrée: nb: la cardinalité de la liste, le tableau contenant les nb nombres
sortie: moyenne : un réel contenant la valeur moyenne des éléments de la liste
résultat : ---
hypothèse : nb est un entier positif inférieur ou égal à MAX, tableau contient nb entiers}

```
Début
        {A : nb > 0 et =< MAX}
        {A : le tableau contient nb entiers}
        somme ← 0
        Répéter ∀ i ∈ [1 , nb]
        Début
                somme ← somme + tableau[i]
        Fin
        moyenne ← somme / nb
        {A : moyenne contient la moyenne de tous les éléments de la liste }
Fin
```

Bloc B4: Affichage des résultats

{**description** : Affichage de la moyenne d'une liste de nombre entiers ainsi que de l'écart entre chaque nombre avec cette moyenne : on balaie tous les éléments de la liste, pour chacun d'eux, on fait la différence avec la moyenne
entrée: nb : la cardinalité de la liste, moyenne: la valeur moyenne des éléments de la liste, le tableau contenant les nb éléments de la liste.
sortie: ----
résultat : Affichage de la moyenne ainsi que de l'écart entre chaque nombre avec cette moyenne
hypothèse : nb est un entier positif inférieur ou égal à MAX, moyenne est un réel contenant la moyenne de tous les éléments de la liste, le tableau contient nb éléments entiers.}

```
Début
        {A : nb > 0 et <= MAX}
        {A : tableau contient nb entiers}
        {A : moyenne contient la moyenne des éléments de la liste }
        afficher "La moyenne des nb nombres est égal à " moyenne
        Répéter ∀ i ∈ [1 , nb]
        Début
                ecart ← moyenne - tableau[i]
                afficher "l'écart entre la" i +1 "notes et la moyenne est" ecart
        Fin
        {A:La moyenne ainsi que l'écart entre chaque note et la moyenne sont affichées}
Fin
```

Exercice#2 Tableaux à deux dimensions

On cherche à développer un programme qui permet de remplir deux matrices, calculer puis afficher la matrice somme.
Votre code doit, en plus du programme principal, avoir les fonctions suivantes :

1. void saisirMatrice(int p_matA[LIGNES][COLONNES], int p_matB[LIGNES][COLONNES])

2. void somme (int p_matA[LIGNES][COLONNES],int p_matB[LIGNES][COLONNES], int p_matC[LIGNES][COLONNES])

3. void affiche (int p_matC[LIGNES][COLONNES])

Exercice#3 Trouvez l'erreur dans chacun des segments de programmes suivants et expliquez comment la corriger.

a) int b[4] = {1,2,,4};

b) Présumez que int b[10]={0};

```
for (int i = 0;i <= 10; i++)
    b[i] = 1;
```

c) Présumez que int a[2][2] = {{1,2}, {3,4}} ;

```
a[1,1] = 5 ;
```

d) Considérer le programme qui affiche le contenu d'un tableau suivant :

```
int tab[10]={1,2,3,4,5,6,7,8,9} ;
for (int i = 0 ; i < 10 ; i++) cout << tab+i << endl;
```

Exercice #4 Conversion de chaines de caractères

Il s'agit, à partir d'une chaine de caractères composée d'informations de différents types, d'extraire celles-ci afin de les mémoriser dans des variables de type correspondant.

Vous devez écrire un programme qui initialise un tableau de caractères contenant des informations sur un article provenant d'un magasin de matériel vidéo sous la forme :

Référence Quantité prix Désignation

exemple

```
2979403 2 16.99 Câble DVI-D Dual-Link Mâle à Mâle de six Pieds Noir
```

Sachant que : Référence est un entier
Quantité est un entier
prix en un réel
Désignation est une chaine de caractères pouvant contenir des espaces

Valider les assignations en les visualisant en utilisant le débogueur

Indications :

Se servir d'un flux d'entrée et de son opérateur >> pour convertir

On peut initialiser un flux d'entrée istringstream à partir d'une chaine de caractères C avec

```
istringstream is_chaine(chaine_c);
```

Question :

Qu'est qui peut motiver le choix de placer la désignation en dernier (raison technique)?

Exercice #5 Fichiers texte

On cherche à écrire un programme permettant de lire un fichier contenant des mots (un par ligne), de chercher combien de mots se répètent dans celui-ci, de déterminer le nombre de caractères que compte chacun des mots, d'associer ce nombre à chaque mot et de sauvegarder le résultat obtenu de cette recherche dans un fichier.

Pour cela, commencer par développer les fonctions suivantes:

```
int lireListe (const string& p_nomFichier, string  p_liste [NOMBRE_MAX_MOTS]);
void afficheListe (string  p_liste [NOMBRE_MAX_MOTS], int p_nombreMot);
Int compteRedondances (string  p_liste [NOMBRE_MAX_MOTS], int p_nombreMot);
Void ajouterNombreCaracteres (string  p_liste [NOMBRE_MAX_MOTS], int p_nombreMot);
Void sauveListe (const string& p_nomFichier, string p_ liste [NOMBRE_MAX_MOTS]);
```

Puis développer le programme.