UNIVERSITE IBN TOFAIL
Faculté des Sciences
Département de Mathématiques et d'Informatique
KENITRA

Année universitaire 2021-2022

POO en C++
Série 1

Exercice 1

On considère la structure suivante :

```
#define NB_MAX 5
struct T{
    int notes[NB_MAX];
    int cne;
};
```

1. Ecrire une fonction nommée "`initialise`" qui permet d'initialiser les champs d'une structure de type T.
2. Ecrire une fonction nommée "`echange`" qui permet d'échanger les contenus de deux structures de type T.
3. Ecrire un programme qui permet de tester ces fonctions.

Exercice 2

On considère les déclarations suivantes :

```
int f(int);                         // fonction 1
int f(float);               // fonction 2
void f(int,float);          // fonction 3
void f(float,int);          // fonction 4

int n,p;
float x,y;
char c;
double z;
```

Et on considère les appels suivants :

```
f(n);         f(x);         f(n,x);
f(x,n);       f(c);         f(n,p);
f(n,c);       f(n,z);       f(z,z);
```

Pour chaque appel correct, préciser le numéro de la fonction appelée.

Exercice 3

1. Ecrire une macro *max* qui permet de calculer le maximum de deux entiers, puis tester la valeur de :

```
max(++a, b);
```

   avec par exemple a=4 et b=0. Que peut-on conclure?
   Remplacer la macro par une fonction *inline* et refaites le test. Conclure?
2. Ecrire une fonction *inline* et une macro qui permettent d'échanger les contenus de deux variables, puis tester les avec deux entiers puis avec deux réels. Conclure.

Exercice 4

On considère les déclarations suivantes :

```
#define DIM 10
typedef double T;
```

1. Etant donné le type *Vecteur* défini par :

```
typedef T Vecteur[DIM];
```

   Ecrire dans un fichier source "vecteur1.cpp", les fonctions suivantes :
   - *initialiser* : qui permet d'initialiser un vecteur par le vecteur nul.
   - *saisir* : qui permet de saisir les composantes d'un vecteur
   - *afficher* : qui permet d'afficher un vecteur

   Tester ces fonctions (en écrivant la fonction "main" dans un autre fichier source "test_vecteur.cpp").

2. On modifie le type *Vecteur*, en le définissant comme suit :

```
struct Vecteur{
       T u[DIM];
}
```

   Réécrire les trois fonctions, *initialiser*, *saisir* et *afficher* pour ce nouveau type dans un fichier source "vecteur2.cpp".

3. Ajouter à "vecteur2.cpp", une fonction *element* qui permet l'accès en lecture et en écriture à une composante d'un vecteur. Tester cette fonction.

Exercice 5

1. Ecrire un programme qui permet de :
   - créer un tableau dynamique d'un type donné T, dont la taille est fournie en donnée.
   - saisir la valeur de ses éléments.
   - afficher le tableau.
2. Ecrire les fonctions suivantes :
   - *initialiser* qui alloue de l'espace mémoire pour le tableau et qui met à 0 ses éléments.
   - *saisir* qui saisie les valeurs du tableau.
   - *afficher* qui affiche le tableau.

   Puis modifier le programme précédent en utilisant ces fonctions. Le programme ne doit pas utiliser une variable globale pour le tableau.
3. Modifier le programme de telle sorte que les fonctions de manipulation du tableau et la fonction `main` (pour le test) soient dans deux fichiers séparés.

Exercice 6

En utilisant les pointeurs sur les fonctions, écrire une fonction qui permet de calculer la valeur approchée de la racine d'une fonction continue sur un intervalle avec la méthode de dichotomie. Puis, utiliser cette fonction pour trouver :

- la racine $f(x) = x^3+2x-1$ sur $[0,1]$ à 10^{-9} près.
- une valeur approchée de PI à 10^{-10} près.