Programmation en C

Atelier 05 : Les fonctions

Pour mener à bien cet atelier, chaque étudiant devra créer un projet par exercice en suivant la notation : *Atl05_ExoX_Prenom_Nom*.

Le *X* doit être remplacé le numéro correspondant de l'exercice. *Prenom* et *Nom* doit être remplacé par votre prenom et nom sans les accents.

Exercice 1

Ecrire en C une fonction permettant d'afficher un rectangle d'étoile "*" suivant les dimensions saisies par l'utilisateur. Ces dimensions doivent être envoyées en paramètres à la fonction.

Exercice 2

Écrire une fonction qui convertit les kilomètres en miles (1 mile = 1,609 km). Ecrire dans le main le code principal qui permet de tester la fonction.

Exercice 3

Écrire une fonction qui convertit les degrés Fahrenheit en degrés centigrades. Ecrire dans le main le code principal qui permet de tester la fonction.

$$\textbf{Formule :}\ \theta_{\mathrm{C}} = \frac{5}{9}(\theta_{\mathrm{F}} - 32)$$

Exercice 4

Écrire une fonction qui calcule le volume d'une sphère. Ecrire dans le main le code principal qui permet de tester la fonction.

Exercice 5

Écrire une fonction ayant en paramètres le nombre d'heures exécutées par un salarié dans la semaine et son salaire horaire, qui retourne sa paye hebdomadaire.

On prendra en compte les heures supplémentaires (au-delà de 35 heures) payées à 150%.

Ecrire dans le main le code principal qui permet de tester la fonction.

Exercice 6

Ecrire une fonction qui prend en entrée les coefficients d'une équation du second degré et affiche les racines réelles s'il y en a. Ecrire dans le main le code principal qui permet de tester la fonction.

Exercice 7

Soit la fonction Moyenne qui retourne la moyenne de trois réels

- Ecrire le programme de la fonction
- Ecrire un programme principal de test qui contient sa déclaration ainsi que les instructions l'appelant

Exercice 8

Ecrire un programme déclarant une fonction calculant le volume d'un cône de révolution et affiche le résultat. Ecrire le programme principal permettant de la tester.

Exercice 9

Créez une fonction `sommeTableau` qui renvoie la somme des valeurs contenues dans le tableau (utilisez un `return` pour renvoyer la valeur). Ecrire le programme principal pour la tester.

Prototype de la fonction à créer :

```
int sommeTableau(int tableau[], int tailleTableau);
```

Exercice 10

Créez une fonction `moyenneTableau` qui calcule et renvoie la moyenne des valeurs. Ecrire le programme principal pour la tester.

Prototype :

```
double moyenneTableau(int tableau[], int tailleTableau);
```

Exercice 11

Créez une fonction `copierTableau` qui prend en paramètre deux tableaux. Le contenu du premier tableau devra être copié dans le second tableau. Ecrire le programme principal pour la tester.

Prototype :

```
void copie(int tableauOriginal[], int tableauCopie[], int tailleTableau);
```

Exercice 12

Créez une fonction `maximumTableau` qui aura pour rôle de remettre à 0 toutes les cases du tableau ayant une valeur supérieure à un maximum. Cette fonction prendra en paramètres le tableau ainsi que le nombre maximum autorisé (`valeurMax`). Toutes les cases qui contiennent un nombre supérieur à `valeurMax` doivent être mises à 0. Ecrire le programme principal pour la tester.

Prototype

```
void maximumTableau(int tableau[], int tailleTableau, int valeurMax);
```

Exercice 13

Créez une fonction `ordonnerTableau` qui classe les valeurs d'un tableau dans l'ordre croissant. Ainsi, un tableau qui vaut `{15, 81, 22, 13}` doit à la fin de la fonction valoir `{13, 15, 22, 81}`. Ecrire le programme principal pour la tester.

Prototype :

```
void ordonnerTableau(int tableau[], int tailleTableau);
```

Exercice 14

Écrivez la fonction qui retourne l'indice du premier élément strictement négatif parmi les n premiers éléments d'un tableau de `double` (`-1` si aucun élément n'est négatif). Ecrire le programme principal pour la tester.

Prototype :

```
int indice_premier_negatif (double t[], int n);
```

Exercice 15

Écrivez la fonction qui retourne la valeur du plus grand des n premiers éléments d'un tableau de `double`. Ecrire le programme principal pour la tester.

Prototype :

```
double valeur_plus_grand (double t[], int n);
```

Exercice 16

Écrivez la fonction qui retourne l'indice du plus grand des n premiers éléments d'un tableau de `double` (en cas d'ex-æquo, l'indice du premier d'entre eux). Ecrire le programme principal pour la tester.

Prototype :

```
int indice_plus_grand (double t[], int n);
```

Exercice 17

A partir de deux tableaux saisis, écrivez une fonction qui calcule le schtroumpf des deux tableaux. Ecrire le programme principal pour la tester.

Pour calculer le schtroumpf, il faut multiplier chaque élément du tableau 1 par chaque élément du tableau 2, et additionner le tout. Par exemple si l'on a :

Tableau 1 :

4	8	7	12

Tableau 2 :

3	6

Le Schtroumpf sera :

3 * 4 + 3 * 8 + 3 * 7 + 3 * 12 + 6 * 4 + 6 * 8 + 6 * 7 + 6 * 12 = 279

Exercice 18

Ecrire une fonction *somme* qui permet de faire l'addition de deux nombres complexes

Ecrire le programme principal qui :

- saisie les parties réelles et les parties imaginaires de deux nombres complexes,
- calcule la somme de deux nombres complexes (fait appel à la fonction somme),
- affiche le résultat de la somme