IUT ANNECY UNIVERSITÉ SAVOIE MONT BLANC

INFORMATIQUE

Sequence 3 : Fonctions

TD 3.1

v2025-09-24

IUT d'Annecy, 9 rue de l'Arc en Ciel, 74940 Annecy

Exercice 1 : Identifier les fonctions

Voici un extrait de programme :

```c
#include <stdio.h>

int somme(int a, int b);

int main(void) {
    int x = 4;
    int y = 5;
    int z = somme(x, y);
    printf("%d\n", z);
}

int somme(int a, int b) {
    return a + b;
}
```

Question 1 Donner le nom de la fonction définie par le programme.

Question 2 Quels sont ses paramètres ? Quelle est sa valeur de retour ?

Question 3 Où la fonction est-elle appelée ? Que va afficher le programme ?

Exercice 2 : Compléter une fonction

On souhaite écrire une fonction `difference(a,b)` qui renvoie la différence absolue entre deux nombres entiers.

Question 1 Dessiner cette fonction sous la forme d'une boite avec entrées et sorties

Question 2 Donner le prototype de la fonction

Question 3 Donner le corps de la fonction

Exercice 3 : Calculer et afficher

Question 1 Écrire une fonction `int carre(int n)` qui renvoie le carré d'un entier.

Question 2 Ecrire une fonction `void afficher_carre(int n)` qui affiche la phrase :

`Le carré de n est ...`

Question 3 Comment la deuxième fonction peut-elle utiliser la première ?

Question 4 Pourquoi est-il utile de séparer les deux fonctions ?

Exercice 4 : Fonctions avec plusieurs appels

Question 1 Ecrire une fonction minimum qui renvoie le minimum entre deux nombres.

On veut une fonction `int minimum3(int a, int b, int c)` qui renvoie le plus grand de trois entiers, en réutilisant la fonction `minimum`.

Question 2 Écrire la fonction `minimum3`.

Question 3 Quel intérêt y a-t-il à réutiliser une fonction déjà écrite ?

Question 4 Comment tester cette fonction rapidement dans le `main` ?

Exercice 5 : Heure en seconde

Question 1 Donner le prototype et le corps d'une fonction qui prend en paramètre le nombre d'`heures`, `minutes` et `secondes` et qui donne en sortie le nombre de secondes écoulées depuis le début de la journée.

Exercice 6 : Fonctions booléennes

On définit une fonction `est_pair` qui teste si un nombre est pair :

Question 1 Ecrire le prototype puis le code de la fonction.

Question 2 Proposer une autre fonction `est_multiple_de(int n, int m)`.

Exercice 7 : S'appeler soi-même

Soit la fonction suivante :

```c
int partez(int n){
    if (n > 0){
        printf("%i, ",n);
        partez(n-1);
    }
    else{
        printf("GO !");
    }
}
```

Question 1 Prédire l'affichage produit par les appels suivants

- `partez(0)`
- `partez(1)`
- `partez(3)`
- `partez(5)`

Une fonction qui s'appelle elle-même est appelée une **fonction récursive.**

Exercice 8 : Fonctions normales ou récursives ?

On souhaite calculer la factorielle d'un entier positif n.

Question 1 Rappeler la définition mathématique de la factorielle.

Question 2 Donner une fonction qui calcule la factorielle d'un nombre donné en paramètres.

Question 3 **Difficile :** Sans utiliser de boucle, écrire une fonction qui calcule la factorielle en s'appelant elle-même.

Exercice 9 : Nombre binaire

Question 1 Ecrire une fonction qui convertie un nombre décimal en binaire avec la méthode de la division.

Question 2 Même question avec une fonction récursive.