TP-03 : Fonctions

Notions : définition, paramètre avec valeur par défaut, passage d'argument par valeur, argument pointeur, passage d'argument par référence, documentation de fonctions.

0. Modèle pour les exercices

Voici le modèle du fichier à utiliser pour les exercices de ce TP. Certains éléments de syntaxe de ce code seront étudiés aux prochaines séances. Vous modifierai le fichier uniquement entre les deux lignes.

`tp3.cpp`

```cpp
#include <iostream>

// Les définitions des fonctions
int f() {
    return 1;
}

int main() {

    // Les appels des fonctions à définir
    int x = f();
    // Affichage si nécessaire
    std::cout << x << std::endl;

    return 0;
}
```

1. Fonctions

1.1. Fonction simple

Définir la fonction `incr` qui retourne son argument plus un.

1.2. Fonction avec plusieurs arguments de différents types et avec une valeur par défaut

Définir la fonction `f1` qui a 2 arguments `value` et `positive` de types int et bool qui retourne `value` si `positive` vaut true et sinon son opposé. le paramètre `positive` a une valeur par défaut qui vaut true. Ecrire 2 appels de `f1` pour montrer que le paramètre avec la valeur par défaut fonctionne.

`f1(1, false)` retourne `-1`

1.3. Grille

Définir la fonction `grid` qui a deux arguments `row` et `column`. `grid` affiche en forme de grille les coordonnées des cases d'une grille de `row` lignes et `column` colonnes.

Par exemple `grid(10, 10)` affiche ceci:

```
00 01 02 03 04 05 06 07 08 09
10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49
50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69
70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89
90 91 92 93 94 95 96 97 98 99
```

1.4. Passage d'arguments par valeur

Définir la fonction copy_plus qui a un argument de type int et qui retourne son argument plus 1 sans le modifier.

1.5. Arguments de type pointeur

Définir la fonction `decr` qui décrémente la valeur de la variable passé en argument. Tester `decr` en lui donnant en argument un

1.6. Passage d'arguments par référence

Définir la fonction `add` qui ne retourne rien et qui a deux arguments a et b qui additionne a et b et met le résultat dans a.

2. Programme

2.1. Documentation de fonctions, fonctions récursive

Définir la fonction factorielle nommée `fact` de manière récursive et écrire sa documentation. Une fonction récursive s'appelle elle même et a besoin d'une condition d'arrêt pour s'arrêter.

Définition par récurrence de *factorielle*:

0! = 1. et *Pour tout entier n > 0, n! = (n – 1)! × n.*

2.2. Programme avec déclaration de fonctions

Dans `tp3_2.cpp`, définir la fonction `fact`. Définir la fonction `arrangement` qui a 2 arguments k et n qui retourne le nombre d'arrangements.

Appeler `arrangement(3,4)` depuis le main. Placer la fonction `main` au dessus des définitions de `fact` et `arrangement`.

Définition de *arrangement*:

Ank = n! (n−k)! pour k<=n_ et _0 pour k>n