STRUCTURES ITERATIVES OU REPETITIVES

PROBLEMATIQUE

Ecrire un programme qui permet d'afficher "Bonjour tout le monde". Le programme doit afficher le message 10 fois sur 10 lignes.

SOLUTION :

Pour faire ce programme (jusqu'ici), on affiche le message avec l'utilisation de la fonction printf() : 10 fois de suite.

Avec cette méthode, une même instruction est répétée plusieurs fois (10 fois) dans le code. Cette méthode devient presque impossible avec un nombre d'affichage élevé. Pour pallier à cela, on met la seule instruction dans une **structure répétitive**.

INTRODUCTION

Les structures répétitives ou **itératives ou boucles** permettent de répéter plusieurs actions ou traitements dans le même programme.

Il existe trois types de boucle en C : for, while et do {…} while

I. LA BOUCLE FOR

Cette boucle est utilisée si on connait le nombre d'itérations à faire pour un traitement donné. L'utilisation de cette boucle nécessite une variable d'indice de parcours qui permet de savoir le nombre de tours effectués.

Syntaxe :

```
for (initialisation ; condition(s) ;  incrémentation/décrémentation)
{
    Instruction(s) ;
}
```

L'initialisation permet de donner une valeur de départ à l'indice de parcours. La condition représente la condition de sortie (valeur de sortie). L'incrémentation et/ou la décrémentation permet de faire avancer ou de faire reculer l'indice de parcours.

NB : Une somme est toujours initialisée à 0 avant la boucle et un produit à 1.

Exemple A : Résoudre la problématique *(à noter)*

Exercice d'application A1 *(à noter)*

Ecrire un programme qui permet de faire la somme de 75 entiers saisis au clavier.

Exercice d'application A2 *(à noter)*

Ecrire un programme qui permet de calculer la factoriel d'un nombre

Exercice d'application A3 *(à noter)*

Ecrire un programme qui permet de saisir n entiers. Le programme affiche le minimum et le maximum des entiers saisis

II. LA BOUCLE DO…WHILE

Cette boucle est utilisée en général pour contrôler la saisie. Elle est capable de faire tous les traitements possible avec la boucle for. Sa condition de sortie se trouve en bas de la boucle ce qui permet de faire le traitement au moins une fois même si la condition n'est pas vérifiée.

Syntaxe :

```
do{
    instruction(s);
}while(condition(s));
```

Exemple B : Résoudre la problématique *(à noter)*

Exercice d'application B *(à noter)*

Ecrire un programme qui permet de saisir la moyenne d'un étudiant. Le programme détermine et affiche si la personne a validé ou pas

III. LA BOUCLE WHILE

Cette boucle est aussi utilisée pour contrôler la saisie. Elle fonctionne de la même façon que do{…}while, sauf que la condition est vérifiée avant d'entrer dans la boucle.

Syntaxe :

```
while(condition(s)){
    instruction(s);
}
```

Exemple C : Résoudre la problématique *(à noter)*

Exercice d'application C1 *(à noter)*

Refaire l'application B

Exercice d'application C2 *(à noter)*

Ecrire un programme qui permet de saisir l'âge d'une personne, le programme détermine et affiche si la personne est mineure ou majeure.