LIFAPI – TD 2 : Algorithmes un peu moins simples

Objectifs : Approfondir les notions vues dans le TD précédent (boucles, conditions, structures de données, entrées / sorties, …)

```
Conditionnelle à choix multiple (sélective)

SELON (sélecteur) FAIRE
              Cas <liste de valeurs-1> : <suite d'action (s)-1>
              [Cas <liste de valeur-2> : <suite d'action (s)-2>
               ……. ]
              [Autrement : <suite d'action (s)-n> ]
FINSELON

Le sélecteur est une variable de type entier ou caractère
```

1. Écrire l'algorithme d'un programme permettant de simuler le fonctionnement d'une calculatrice simple (+, -, *, /). Dans cet exercice, l'utilisateur saisira les deux opérandes, l'opérateur et le programme lui affichera le résultat correspondant. Dans le cas d'une division, on vérifiera bien que le dénominateur est non nul !
    a. avec des `si` imbriqués
    b. avec un sélecteur `selon`

2. Écrire un algorithme permettant de calculer la factorielle d'un entier n donné par l'utilisateur. On écrira une version avec une boucle conditionnelle et une avec une boucle inconditionnelle.
   Exemple : valeur saisie : 6 ➔ résultat : `720 (= 1 * 2 * 3 * 4 * 5 * 6)`

3. Écrire un algorithme permettant de calculer la somme des n premiers nombres impairs.
   Exemple : valeur saisie : 6 ➔ résultat : `36 (= 1 + 3 + 5 + 7 + 9 + 11)`

4. Écrire un algorithme permettant d'afficher toutes les combinaisons possibles de valeurs sur 2 dés. Résultat : `1-1, 1-2, 1-3, 1-4, 1-5, 1-6, 2-1, 2-2, … , 6-5, 6-6`.

5. Modifier le programme précédent pour ne pas afficher les doublons (`1-2` et `2-1` par exemple).

6. Écrire un algorithme qui calcule la moyenne de `n` valeurs saisies par l'utilisateur, `n` étant choisi préalablement par l'utilisateur. On recommencera la saisie de `n` tant qu'il n'est pas strictement positif.

Exercices pour aller plus loin …

1. Écrire un algorithme permettant de lire 20 nombres entiers au clavier. Si le nombre x saisi est pair, on affiche la valeur (x / 2) sinon on affiche (3*x + 1). Attention, on ne mémorisera pas les 20 valeurs saisies.

2. Afficher tous les nombres pairs compris entre 0 et 20 inclus
    a. en utilisant une boucle `pour`
    b. en utilisant une boucle `tant que`