Travaux Dirigés d'algorithmique n°4

Cours d'Informatique de Première Année

—DEUG Sciences S2-SM/STPI—

Fonctions

Dans cette séance, nous définissons nos premières fonctions. Celles-ci seront vues comme des fonctions mathématiques à éventuellement plusieurs variables. Elles retourneront une valeur qui traduira le résultat d'une succession d'opérations.

Les fonctions permettent d'isoler des suites d'instructions afin de pouvoir s'en servir ultérieurement à volonté (en évitant de récrire le même code). Il en résulte un programme mieux structuré.

Quelques mots clefs à connaître :

- la **valeur de retour** d'une fonction ;
- les **paramètres** que prend une fonction ;
- les **arguments d'appel** que l'on donne à une fonction.

Les nombres de paramètres et d'arguments d'appel doivent être identiques.
La syntaxe est la suivante :

```
type myfunction( type param1, ... , type paramN )
{
  /* Suite d'instructions */

  return val;
}
```

▸ **Exercice 1.** ***(Table de multiplication)***
Écrire une fonction `Multiples` *prenant en paramètre un entier* `n` *et qui affiche sur une ligne les 11 premiers multiples de* n. *Par exemple,* `Multiples(2)` *affichera :*

0 2 4 6 8 10 12 14 16 18 20

▸ **Exercice 2.** ***(Positivité d'un entier)***
Reprendre le programme — du TD précédent — qui lit un entier jusqu'à ce qu'il soit positif, modifier celui-ci en utilisant une fonction qui retournera l'entier positif à la fonction appelante.

▸ **Exercice 3.** ***(Exponentielle)***
Écrire une fonction `Exponentielle` *qui prend en paramètre deux entiers* n *et* m *(avec* $m \geq 0$*) et qui retourne* n^m*.*

▸ **Exercice 4.** ***(Somme)*** *On définit la somme* $S_p(n)$ *par :*

$$S_p(n) = 1^p + 2^p + \ldots + n^p \ ,$$

pour $n > 0$ *et* $p \geq 0$*. Écrire une fonction* `Somme` *qui prend en paramètre deux entiers* n *et* p *et qui retourne* $S_p(n)$*.*

▸ **Exercice 5.** ***(Minimum de deux entiers)***
Écrire une fonction `Min` *qui prend en paramètre deux entiers* n *et* m *et qui retourne le minimum de ces deux entiers.*

▸ **Exercice 6.** ***(Plus grand commun diviseur)***
Écrire une fonction `Pgcd` *qui prend en paramètre deux entiers* n *et* m *avec* $n, m \neq 0$ *et qui retourne le plus grand diviseur commun de* n *et* m*. On utilisera la méthode naïve décrite ci-dessous :*
On pose $p = \min(n, m)$ *puis on cherche les restes* r_1 *et* r_2 *de la division de* n *et* m *par* p*. Si* $r_1 = 0$ *et* $r_2 = 0$*, le pgcd est* p*. Sinon, on effectue de nouveau les divisions avec* $p - 1$*, ainsi de suite…*

▸ **Exercice 7.** ***(Plus petit multiple commun)***
Écrire une fonction `Ppcm` *qui prend en paramètre deux entiers* n *et* m *et qui retourne le plus petit multiple commun de* n *et* m*.*