CI 2 : Algorithmique & Programmation

Chapitre 2 – Introduction à l'algorithmique
Exercices d'application 1

Compétences

- *Alg – C1 :* comprendre un algorithme et expliquer ce qu'il fait ;
- *Alg – C2 :* modifier un algorithme existant pour obtenir un résultat différent ;
- *Alg – C4 :* expliquer le fonctionnement d'un algorithme ;
- *Alg – C10 :* concevoir l'en-tête (ou la spécification) d'une fonction, puis la fonction elle-même ;
- *Alg – C11 :* traduire un algorithme dans un langage de programmation ;
- *Alg – C13 :* rechercher une information au sein d'une documentation en ligne, analyser des exemples fournis dans cette documentation ;
- *Alg – C14 :* documenter une fonction, un programme plus complexe.

Exercice 1

Question 1 *Traduire cet algorithme en Python en implémentant la fonction is_even renvoyant True si un entier est pair, False sinon. Vous n'oublierez pas de documenter la fonction.*

Pseudo Code

Algorithme 1 : Pair ou impair ?
Données : n : un entier
Résultat : r : un booléen vrai si n est pair, faux si n est impair.

Si n **mod** $2 == 0$ **alors**
    $r \leftarrow$ Vrai
Sinon
    $r \leftarrow$ Faux
Fin Si

Exercice 2

Question 1 *Réaliser l'algorithme en utilisant une boucle Pour permettant de calculer la somme des n premiers entiers. Vous utiliserez la syntaxe Python ou Pseudo code.*

Question 2 *Réaliser l'algorithme en utilisant une boucle Tant que permettant de calculer la somme des n premiers entiers. Vous utiliserez la syntaxe Python.*

Exercice 3 – Calcul de 2^n

Question 1 *Implémenter la fonction P2_explicite(n) en utilisant la méthode n disponible dans la bibliothèque de fonction math.*

python

```
>>> help(pow)
Help on built-in function pow in module builtins:

pow(...)
    pow(x, y[, z]) -> number

    With two arguments, equivalent to x**y.  With three arguments,
    equivalent to (x**y) % z, but may be more efficient (e.g. for ints).
```

Question 2 *Implémenter la fonction P2_iterative(n) en utilisant une boucle Tant que.*

Exercice 4

L'algorithme suivant permet de calculer le nième terme de la suite de Syracuse.

Pseudo Code

Algorithme 4 : Suite de Syracuse

```
Syracuse(n)
    syr ← n
    Tant que syr ≠ 1 faire :
        Si syr mod 2 == 0 alors :
            syr ← syr/2
        Sinon
            syr ← 3*syr+1
        Fin si
    Fin Tant que
    Retourner syr
```

Question 1 *Calculer Syracuse(10) et Syracuse(12) et observer l'évolution de la variable syr. En déduire la conjecture de Syracuse.*

Question 2 *Donner les spécifications de la fonction.*

Question 3 *Donner l'énoncé mathématique de la suite de Syracuse.*

Question 4 *On appelle **temps de vol** le plus petite indice n tel que $u_n = 1$. Modifier l'algorithme pour le calculer.*

Question 5 *On appelle **altitude** la valeur maximale de la suite. Modifier l'algorithme pour la calculer.*

Exercice 5

On donne l'algorithme suivant.

Pseudo Code

Algorithme 5 : Insertion d'un élément dans une liste de nombres triés par ordre croissant

Données :
- T : une liste de nombres triés par ordre croissant $T[1..n]$;
- x : un nombre.

Résultat : T : une liste de nombre triés par ordre croissant $T[1..n+1]$

```
Insertion_element(T,x)
    i ← n
    Tant que T[i] > x ou i > 0 faire :
        T[i+1] ← T[i]
        i ← i-1
    Fin Tant que
    T[i+1] ← x
```

Question *Expliquer le processus permettant d'insérer un élément dans un tableau. Vous pourrez sur les exemples suivants :*
- $T = [1,2,4,5]$, *Insertion_element(T,3)* ;
- $T = [1,2,4,5]$, *Insertion_element(T,6)* ;