

**Informatique PCSI**  
**TP 2 : boucles imbriquées**

## 1 Énoncé des exercices

### 1.1 Exercice 1

Combien de fois la fonction `print` est-elle appelée dans le code suivant ?

```
for i in range(5):  
    for j in range(i+1, 5):  
        print(i + j)
```

### 1.2 Exercice 2

Pour chacun des trois scripts, déterminer le nombre d'additions effectuées. Exprimer ce nombre en fonction de  $n$  pour les deux derniers.

Script 1

```
x = 0  
for i in range(2):  
    x = x + i  
    for j in range(3):  
        x = x + j
```

Script 2

```
x = 0  
for i in range(2):  
    x = x + i  
    for j in range(n):  
        x = x + j
```

Script 3

```
x = 0  
for i in range(n):  
    x = x + i  
    for j in range(n):  
        x = x + j
```

### 1.3 Exercice 3

Après le code Python qui suit, quelles sont les valeurs finales de  $x$  et de  $y$  ?

```
x = 4
while x > 0:
    y = 0
    while y < x:
        y = y + 1
        x = x - 1
```

### 1.4 Exercice 4

On considère le texte "ababababab" et le motif "abc". Déterminer le nombre de comparaisons effectuées par une fonction implémentant un algorithme naïf. En déduire le nombre de comparaisons effectuées si le texte "ababab..." contient 100 caractères, (50 fois "ab"), puis une expression de ce nombre en fonction de  $n$  si le texte contient  $n$  caractères sur le même modèle.

### 1.5 Exercice 5

Créer une liste de 10000 caractères choisis de manières aléatoires parmi les caractères "A", "C", "G" et "T". Rechercher dans cette liste le motif "CAGCAG" à l'aide d'une fonction.

*Un programme de recherche textuelle peut être utilisé avec des chaînes de caractères ou des listes de caractères.*

### 1.6 Exercice 6

On se donne un ensemble  $E$  de  $n$  points réels, un entier  $k$  inférieur à  $n$ , et un point  $x$  qui n'est pas dans  $E$ . Il s'agit de trouver parmi les points de  $E$  les  $k$  plus proches de  $x$ . Le mot « proche » sous-entend une notion de distance.

Voici un exemple d'algorithme où on construit une liste appelée voisins qui contient les  $k$  plus proches voisins d'un point  $x$  parmi les éléments d'un ensemble  $E$  représenté par une liste :

```
Pour i allant de 0 à k - 1
    placer le point E[i] dans la liste voisins
Fin pour
Pour i allant de k à n - 1
    Si la distance entre x et E[i] est inférieure à la distance entre x et
    un point de la liste voisins
        supprimer de la liste voisins le point le plus loin de x
        placer dans la liste voisins le point E[i]
    Fin si
Fin pour
```

Définir une fonction `distance`.

Écrire une fonction `proches_voisins`, implémentant l'algorithme, qui prend en paramètres  $x$ ,  $k$  et la liste des points.