# TD 3

## Exercices d'application

*D'après IPT, Éditions Vuibert.*

***Savoirs et compétences :***
- ❒ *Alg – C15 : Récursivité : avantages et inconvénients.*

## Exercice 1

Soit l'algorithme suivant :

```python
def mult(n, p):
    if p == 0:
        return 0
    else :
        return n+mult(n,p-1)
```

**Question 1** *Énoncer un variant de boucle et montrer la terminaison de l'algorithme.*

**Question 2** *Énoncer un invariant de boucle et montrer la correction de l'algorithme.*

**Question 3** *Donner et justifier la complexité temporelle de la fonction* `mult`.

**Question 4** *Donner et justifier la complexité spatiale de la fonction* `mult`.

## Exercice 2

Soit l'algorithme suivant :

```python
def puiss(x, n):
    if n == 0:
        return 1
    else :
        return x*puiss(x,n-1)
```

**Question 1** *Énoncer un variant de boucle et montrer la terminaison de l'algorithme.*

**Question 2** *Énoncer un invariant de boucle et montrer la correction de l'algorithme.*

**Question 3** *Donner et justifier la complexité temporelle de la fonction* `puiss`.

**Question 4** *Donner et justifier la complexité spatiale de la fonction* `puiss`.

## Exercice 3

Soit l'algorithme suivant :

```python
def rechecheDichoRec(x, l):
    n=len(l)
    if n == 0:
        return False
    elif x<l(n//2) :
        return rechecheDichoRec(x, l(0:n//2))
    elif :
        return rechecheDichoRec(x, l(n//2:n))
    else :
        return True
```

**Question 1** *Donner et justifier la complexité temporelle de la fonction* `rechecheDichoRec`.

**Question 2** *Donner et justifier la complexité spatiale de la fonction* `rechecheDichoRec`.

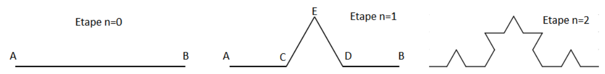
## Exercice 4 – Flocon de Von Koch

Dans cet exercice, vous utiliserez des tableaux **numpy** pour représenter les points. C'est plus pratique que les listes python pour faire les calculs vectoriels.

- Si $a$ et $b$ représentent respectivement les points $(x, y)$ et $(x', y')$ alors $a+b$ représente le point $(x+x', y+y')$.

- Si $r$ est un réel et $a$ représente le point de coordonnées $(x, y)$ alors $r * a$ représente le point $(rx, ry)$.
- Si $a$ et $b$ sont des tableaux **numpy** alors $dot(a, b)$ représente le produit matriciel $a \times b$ (si ce produit est possible). La fonction **dot** est une fonction **numpy**.

Le mathématicien suédois Von Koch a défini la courbe du même nom dont voici les premières itérations.




**Question 1** *Ecrire une fonction* `rotation` *d'argument un réel* `alpha` *qui renvoie le tableau **numpy** correspondant à la matrice de rotation d'angle* `alpha`.

**Question 2** *Pour l'étape $n = 1$, exprimer les points $C$ et $D$ en fonction de $A$ et $B$.* En utilisant une matrice de rotation, exprimer $E$ en fonction de $C$ et $D$.

**Question 3** *En déduire une fonction récursive* `koch` *d'arguments les points* `A` *et* `B` *et un entier* `n`. *Cette fonction tracera la courbe de Von Koch pour l'itération $n$ à partir des points $A$ et $B$.*

**Question 4** *Ecrire une fonction* `flocon` *d'arguments les points $A$ et $B$ et un entier $n$. Cette fonction tracera le flocon de Von Koch pour l'itération $n$ à partir des points $A$ et $B$.*