

# DEVOIR SURVEILLÉ 3 – 1 HEURE

## ALGORITHMIQUE ET PROGRAMMATION

### Avant-propos – Calcul d'une puissance

On souhaite calculer la puissance  $b$  d'un nombre  $x : x^b$  avec  $x \in \mathbb{R}$  et  $b \in \mathbb{N}$ . On utilise pour cela la fonction expo basée sur un algorithme naïf prenant comme argument un entier naturel  $b$  et un nombre réel  $x$  :



```
def expo(x,b):
    res = 1
    j=b
    inv = x
    while j>=1:
        res = res * x
        j=j-1
    return res
```

#### Question 1

Proposer une autre formulation de l'algorithme de la fonction expo en utilisant une boucle for.

#### Question 2

On conserve la fonction expo utilisant la boucle while. Montrer que  $j$  est un **variant** de boucle.

#### Question 3

On conserve toujours la fonction expo utilisant la boucle while. Montrer que la propriété  $\mathcal{P}(n) \ x^b = inv_n^{j_n} \cdot res_n$  est un **invariant** de boucle.

#### Question 4

On note  $C_e$  le coût d'une opération élémentaire (affectation, opération mathématique simple, incrémentation de boucle, comparaison). Évaluer la complexité temporelle de l'algorithme proposé dans la fonction expo.

#### Question 5

Citer une méthode plus efficace permettant de calculer  $x^b$ . Détailler brièvement son fonctionnement et préciser sa complexité temporelle.

### Calcul de polynômes

On cherche à évaluer un polynôme en différentes valeurs. On note :

$$\forall x \in \mathbb{R} \quad P(x) = \sum_{i=0}^n a_i x^i$$

Les coefficients  $a_i$  du polynôme sont des entiers positifs stockés dans un tableau  $a$  tels que  $a = [a_0, a_1, a_2, \dots, a_n]$ . La fonction suivante appelée evaluer prend comme argument un nombre flottant  $x$  et un tableau  $a$  contenant les coefficients  $a_i$  du polynôme. Ainsi, si  $a = [0, 1, 2, 3]$ , alors  $a[0] = a_0$ ,  $a[1] = a_1$ , etc. alors  $P(x) = x + 2x^2 + 3x^3$ . La fonction evaluer retourne  $P(x)$ .



```
def evaluer(a,x):
    for i in range(len(a)):
        res = res+a[i]*expo(x,i)
    return res
```

#### Question 6

La fonction evaluer a-t-elle l'effet désiré ? Si non, modifier le programme.

### Question 7

Estimer la complexité algorithmique de la fonction evaluer.

### Méthode de Horner

Afin de diminuer le coût temporel de l'évaluation d'un polynôme, il est possible d'utiliser la méthode de Horner. Elle consiste en une réécriture du polynôme  $P(x)$  :

$$P(x) = a_0 + x(a_1 + x(a_2 + x(a_3 + \dots)))$$

Ainsi le polynôme  $P(x) = x + 2x^2 + 3x^3$  est réécrit ainsi :  $P(x) = 0 + x(1 + x(2 + 3x))$ .

python

```
def horner(a,x):
    res=0
    n = len(a)-1
    while n>=0:
        res = a[n]+x*res
        n=n-1
    return res
```

### Question 8

On prend  $a = [0, 1, 2, 3]$  et  $x = 2$ . En remplissant un tableau, donner l'évolution des variables  $res$  et  $n$  à chaque incrément de boucle.

### Question 9

Expliquer en quoi l'algorithme proposé répond à la réécriture du polynôme  $P(x)$  suivant la méthode de Horner ?

### Question 10

Estimer la complexité algorithmique de la fonction horner. Conclure sur l'intérêt de cet algorithme.

### Intégration numérique

On cherche maintenant à intégrer numériquement  $P(x)$  sur l'intervalle  $[u, v]$  par la méthode des rectangles à gauche :

$$I = \int_u^v P(x) dx$$

### Question 11

Écrire la fonction `integrale_rectangle` prenant comme argument le nombre d'échantillons  $n$ , le tableau  $a$  des coefficients du polynôme ainsi que  $u$  et  $v$  les bornes de l'intégrale et retournant la valeur  $I$  de l'intégrale.

### Question 12

Quel est l'ordre de grandeur de l'erreur effectuée sur le calcul de l'intégrale.