Séance 07 :

Converion et méthode de Horner

1. Passage de paramètre

On donne les commandes suivantes :

```
1  p = [1, 2, 3, 4]
2
3
4  def f(p):
5      p[0] = 5
6      return p
7
8
9  s = f(p)
10 q = p
11 p = [1, 2, 3, 4]
12
13
14 def g(p):
15     p = p[::-1]
16     p[0] = 5
17     return p
18
19
20 r = g(p)
21 u = p
```

1. ***Donner le contenu des variables*** `p`, `s`, `q`, `r` ***et*** `u` ***à la fin du programme.***

   - ❏ La ligne 9 implique : `s = [5, 2, 3, 4]`
   - ❏ La ligne 10 implique : `q = s`. En effet la fonction `f` modifie la variable globale `p` (qui lui est passée en paramètre) car `p` est une liste donc mutable, donc la ligne 5 modifie directement `p`.
   - ❏ La ligne 20 implique : `r = [5, 3, 2, 1]` car la commande `p = p[::-1]` inverse `p`.
   - ❏ La ligne 21 par contre implique : `u = p = [1, 2, 3, 4]`. En effet, cette fois-ci avant d'appliquer la commande `p[0] = 5`, on copie tout le contenu de `p` dans une nouvelle variable. Ainsi, la variable globale `p` (passée en paramètre de `g`) n'est pas modifiée, c'est cette nouvelle variable qui l'est.

   D'où :

$$\begin{cases} \texttt{s} = \texttt{q} = \texttt{[5, 2, 3, 4]} \\ \texttt{p} = \texttt{u} = \texttt{[1, 2, 3, 4]} \\ \texttt{r} = \texttt{[5, 3, 2, 1]} \end{cases}$$

2. Méthode de Horner

1. ***Ecrire un programme*** `h(P,x)` ***qui calcule*** $P(x)$ ***suivant la méthode de Horner en n'effectuant que*** $(n-1)$ ***additions ou soustractions et*** $(n-1)$ ***produits, où*** `n = len(p)`.

```
1 def h(P, x):
2     """P : liste des coefficients du polynome"""
3     y = P[0]
4     for i in P[1:]:
5         y = y * x + i
6     return y
```

3. Exercice 1

1. ***Ecrire un programme*** `alphab` ***de conversion de la base*** 10 ***à la base*** 26 ***suivant le principe donné.***

```
alphabet = 'abcdefghijklmnopqrstuvwxyz'


def alphab(n):
    word = ''
    while n != 0:
        word += alphabet[n % 26]
        n = n // 26
    return word[::-1]
```

Lorsqu'on exécute les exemples demandés, on trouve :

`alphab(53)`	=	`cb`
`alphab(189527117)`	=	`python`
`alphab(211413)`	=	`math`
`alphab(319098)`	=	`seba`

4. Exercice 2

1. ***Ecrire un programme*** `numer` ***de conversion de la base*** 26 ***à la base*** 10 ***en utilisant la fonction*** `index` ***des listes (voir mémento) qui fait la réciproque de*** `alphab`.

```
def nummerHorner(chaine):
    p = [alphabet.index(k) for k in chaine]
    return horner(p, 26)


def numer(s):
    n = 0
    for a in s[:-1:]:
        n = (n + alphabet.index(a)) * 26
    return n + alphabet.index(s[-1])
```

`numer(math)`	=	`211413`
`numer(latex)`	=	`5039707`