COMPÉTENCE 11 : Récursivité

❖ Exercice 11.1 : Avanti ! (⋆)

Écrire une fonction récursive sommeArithRec(n,a,r) calculant la somme des n premiers termes d'une suite arithmétique de premier terme a et de raison r donnés.

❖ Exercice 11.2 : Suite géo ! (⋆)

Même chose pour une suite géométrique sommeGeoRec(n,a,r)

❖ Exercice 11.3 : Un calcul du PGCD récursif (⋆)

Écrire une fonction récursive pgcdRec(a,b) calculant le PGCD de deux entiers positifs a et b (non tous deux nuls) par l'algorithme d'Euclide.

❖ Exercice 11.4 : Inverser une enîahc récursivement (⋆)

- Écrire une fonction récursive inverseRec(chaine) renvoyant l'inverse d'une chaîne de caractères passée en argument.

Par exemple :

```
>>> inverse("blabla")
albalb
>>> inverse("") #chaine vide
''
>>> inverse("ressasser")
ressasser
```

- En déduire une fonction booléenne palindrome(chaîne) testant si la chaîne donnée en argument est égale à son inverse.

❖ Exercice 11.5 : N! itératif versus N! récursif (⋆)

1. Écrire une fonction récursive factRec(n) qui renvoie la factorielle d'un entier n donné.

2. Écrire à présent une fonction itérative fact(n) qui renvoie la factorielle d'un entier n donné.

3. Écrire une fonction genereTabAlea(N,max) qui retourne une liste de N entiers compris entre 0 et max.

4. Calculer les factorielles des éléments du tableau par chacune des deux méthodes, stocker les résultats des calculs, mesurer les deux temps d'exécution. Proposer une explication à cette contre-performance.

❖ Exercice 11.6 : calcul de x^n, exercice fondamental (⋆⋆)

1. Écrire une fonction puissanceRec(x,n) prenant en entrée deux paramètres x et n et calculant récursivement x^n, en exploitant l'égalité $x^n = x \times x^{n-1}$ valable pour tout n ≥ 1. On conviendra que $x^0 = 1$ pour tout réel x.

2. Écrire une fonction puissance(x,n) itérative qui effectue le même calcul.

3. Estimer la complexité de chacune des fonctions précédentes.

❖ Exercice 11.7 : une Vraie copie ! (⋆)

```
3  def puissanceRec(x,n):
4      if n==0:
5          return(1)
6      else:
7          q,r=n//2,n%2
8          u=puissanceRec(x,q)
9          if r==0:
10             return(u*u)
11         else:
12             return(u*u*x)
```

1. Démontrer la terminaison et la correction de puissanceRec.
2. Montrer que puissanceRec() s'exécute en temps $O(log(n))$, en considérant uniquement les opérations arithmétiques.
3. Montrer comment calculer le n ième terme de la suite de Fibonacci en temps O(ln(n)). On pourra penser matriciellement.

❖ Exercice 11.8 : Fonctions mutuellement récursives ! (⋆)

Soient les deux fonctions récursives suivantes, s'appelant mutuellement l'une l'autre.

```
14 def pair(n):
15     if n==0:
16         return(True)
17     else:
18         return(impair(n-1))
19
20 def impair(n):
21     if n==0:
22         return(False)
23     else:
24         return(pair(n-1))
```

Démontrer la terminaison et la correction des deux fonctions ci-dessus.

⁂