Informatique en CPGE (2018-2019)
Corrigé TD 3 : récursivité

Exercice 1 : suite de Fibonacci

```
# une boucle while
def fibo0(n):
    a,b,c=1,1,1
    while c<n:
        a,b,c=b,a+b,c+1
    return b
print('fibo0(350)',fibo0(350))

# une fonction non récursive
def fibo1(n):
    a,b=1,1
    for i in range(n-1):
        a,b=b,a+b
    return b
print('fibo1(350)',fibo1(350))

# une fonction récursive non terminale

def fibo2(n):
    if n==0 or n==1:
        return 1
    else:
        return fibo2(n-1)+fibo2(n-2)
for i in range(36):
    print('fibo2('+str(i)+') = \t',fibo2(i))

# une fonction récursive terminale
def fibo3(n,i,a,b):
    if i==n:
        return b
    else:
        return fibo3(n,i+1,a+b,a)
print('fibo3(350)',fibo3(350,0,1,1))
```

Exercice 2 : algorithme de Hörner

```
# polynome
#c=[1.3,-3,1,4,8,-4,2,1,5,-6,3] # test coef reels
#c=[1,-3,1,4,8,-4,2,1] # test coef entiers


#1- algorithme basique
```

```
def polynome(coef,x):
    p=0
    for i in range(len(coef)):
        f=1
        for j in range (1,i+1):
            f*=x
        p+=coef[i]*f
    return p

from time import time
st1=time()
for x in range(10000): # calcul de P(x) pour x de 0 à 9999
    polynome(c,x)
print('Pour algo basique, temps = ',time()-st1)

#2- algo horner
def horner(coef,x):
    n=len(coef)-1
    p=coef[n]
    for i in range(1,n+1):
        p=p*x+coef[n-i]
    return p

st2=time()
for x in range(10000):
    horner(c,x)
print('Pour algo horner, temps = ',time()-st2)

#2- algo horner recursif
def horner_rec(coef,x):
    copie=coef[:]  # pour que coeff ne soit pas modifié
    if len(copie)==1:
        return copie[0]
    copie[-2]+=x*copie[-1]
    return horner_rec(copie[:-1],x)

st3=time()
for x in range(10000):
    horner_rec(coef,x)
print('Pour algo horner recursif, temps = ',time()-st3)
```