ISM MPSI/PCSI, Informatique
Année 2023/2024

Corrigé du TP Informatique 6

Exercice 1

1. La variable `k` n'est pas modifiée. Le test `k<n` est donc toujours vrai ce qui provoque une boucle infinie.
Il faut aussi modifier le test pour calculer la somme jusqu'à n et non $n-1$.
On saisit :

```
def somme(n):
    k,res=0,0
    while k<=n:
        res+=k
        k+=1
    return res
```

La fonction ainsi modifiée renvoie la valeur de la somme $\sum_{k=0}^{n} k$.

2.

```
def fact(n):
    k,res=1,1
    while k<n:
        k+=1
        res*=k
    return res
```

3. On saisit :

```
def somme1(n):
    res=0
    k=1
    while k<=n:
        res+=k**2
        k+=2
    return res
```

ou aussi

```
def somme1(n):
    res=0
    k=0
    while 2*k+1<=n:
        res+=(2*k+1)**2
        k+=1
    return res
```

4. On saisit :

```
def geom(n,q):
    res,qk=0,1
    for k in range(n):
        res+=qk
        qk*=q
    return res
```

Exercice 2

1. Le plus simple est d'effectuer une boucle par valeur descendante :

```
def w1(n):
    if n%2==0:
        res=np.pi/2
    else:
        res=1
    k=n
    while k>1:
        res*=(k-1)/k
        k-=2
    return res
```

2. On teste les cas de base $n = 0, 1$ et un cas quelconque selon si n est pair ou non.

```
def test_w1() :
    assert w1(0) == np.pi/2
    assert w1(1) == 1
    assert w1(4) == 3*np.pi/16
    assert w1(5) == 8/15
```

3. On reprend l'idée d'une boucle descendante avec un `range` décroissant :

```
def w2(n):
    if n%2==0:
        res=np.pi/2
    else:
        res=1
    for k in range(n,1,-2):
        res*=(k-1)/k
    return res
```

4. On fait de même qu'à la question 2. :

```
def test_w2() :
    assert w2(0) == np.pi/2
    assert w2(1) == 1
    assert w2(4) == 3*np.pi/16
    assert w2(5) == 8/15
```

3. On saisit :

```
for n in range(20):
    print(w1(n),w2(n),w3(n))
```

On observe :

```
1.5707963267948966 1.5707963267948966 1.5707963267948966
1 1 0.9999999999999999
0.7853981633974483 0.7853981633974483 0.7853981633974483
...
```

On peut donc conjecturer que les trois suites sont égales.

Exercice 3

1. On saisit :

```
def seuil(p):
    k,s=0,0
    while s<p:
        k+=1
        s+=k
    return s,k
```

2. On peut tester la valeur initiale de n, puis le cas où $\phi(p) = p$ pour vérifier une éventuelle erreur d'encodage entre l'inégalité stricte et l'inégalité large. On saisit :

```
def test_seuil() :
    assert seuil(0) == (0,0)
    assert seuil(3) == (3,2)
    assert seuil(4) == (6,3)
```

Exercice 4

1. On saisit :

```
def somme_log(n):
    res,k=0,1
    while k*np.log(k)<=n:
        res+=k
        k+=1
    return res
```

2. Oui, c'est possible, mais on écrit une boucle `while` déguisée puisqu'on casse la boucle `for` :

```
def somme_log_bis(n):
    res=0
    for k in range(1,n):
        if k*np.log(k)>n:
            return res
        res+=k
```

On est contraint de casser la boucle puisqu'on ne sait pas déterminer de réciproque à l'application $x \mapsto x \ln x$.