ISM MPSI/PCSI, Informatique
Année 2025/2026

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. On saisit :

```
def heron(n) :
    a = 1
    for k in range(n) :
        a = (a+2/a)/2
    return a
```

3. On teste l'initialisation pour $n = 0$ et un cas quelconque pour vérifier qu'on renvoie bien u_n et non u_{n-1} ou u_{n+1}.

```
def test_heron() :
    assert heron(0) == 1
    assert heron(1) == 1.5
```

4. On saisit :

```
def heron2(n) :
    a = 1
    k = 0
    while k<n :
        a = (a+2/a)/2
        k += 1
    return a
```

5.

```
def test_heron2() :
    assert heron2(0) == 1
    assert heron2(1) == 1.5
```

6. On saisit :

```
def seuil_heron(e) :
    a0 = 1
    a1 = (a0+2/a0)/2
    k = 0
```

```
    while abs(a0-a1)>e :
        a0,a1 = a1,(a1+2/a1)/2
        k += 1
    return k
```

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 $x \mapsto x \ln x$.