I. Suites définies explicitement

Exercice 1

Soit (u_n) la suite définie sur $\mathbb{N}$ par $u_n = 5n^3 - 4n^2 - 10$.

1. Écrire une fonction python `suite` de paramètre n qui renvoie la valeur de u_n.

```
def suite(n):
    return 5*n**3-4*n**2-10
```

2. A l'aide de cette fonction, donner la valeur de u_{100}.

```
>> suite(100)
4959990
```

On obtient $v_{100} = 4959990$

3. Vérifier votre réponse en calculant à la main u_{100}.

$u_{100} = 5 \times (100)^3 - 4 \times (100)^2 - 10 = 4959990$

II. Suites définies par récurrence

Exercice 2

Soit (p_n) la suite définie sur $\mathbb{N}$ par $\begin{cases} p_0 &= 0,5 \\ p_{n+1} &= 0,3p_n + 0,1 \end{cases}$

1. Écrire une fonction python `suite2` de paramètre n qui renvoie la valeur de p_n.

```
def suite2(n):
   p=0.5
   for i in range(n):
        p=0.3*p+0.1
   return p
```

2. Utiliser cette fonction pour déterminer p_1, p_2 et p_3.

```
>> suite2(1)
0.25

>> suite2(2)
0.175

>> suite2(3)
0.1525
```

3. Vérifier vos réponses en calculant à la main p_1, p_2 et p_3.

$p_1 = 0,3p_0 + 0,1 = 0,3 \times 0,5 + 0,1 = 0,25$
$p_2 = 0,3p_1 + 0,1 = 0,3 \times 0,25 + 0,1 = 0,175$
$p_3 = 0,3p_2 + 0,1 = 0,3 \times 0,175 + 0,1 = 0,1525$

Exercice 3

Soit (v_n) la suite définie par $v_0 = -5$ et pour tout entier naturel n, $v_{n+1} = v_n + n^2 + 1$.

1. Calculer à la main v_4.

$v_1 = v_0 + 0^2 + 1 = v_0 + 1 = -5 + 1 = -4$
$v_2 = v_1 + 1^2 + 1 = v_1 + 2 = -4 + 2 = -2$
$v_3 = v_2 + 2^2 + 1 = v_2 + 5 = -2 + 6 = 3$
$v_4 = v_3 + 3^2 + 1 = v_3 + 10 = 3 + 10 = 13$

2. Compléter la fonction python suivante `suite3` de paramètre n qui renvoie le terme v_n.

```
def suite3(n):
    v=5
    for i in range(n):
        v=v+i**2+1
    return v
```

3. Utiliser cette fonction pour retrouver la valeur de v_4.

```
>> suite2(4)
13
```

On obtient $v_4 = 13$.

III. Somme des premiers termes d'une suite

Exercice 4

Soit (u_n) la suite géométrique de premier terme $u_0 = 15$ et de raison $q = 2$.
On pose, pour tout entier naturel n, $S_n = u_0 + u_1 + ... + u_n$.

1. En utilisant une formule du cours, calculer : $S_{10} = u_0 + u_1 + ... + u_{10}$.

$$\begin{aligned} S_{10} &= u_0 \frac{1-q^{11}}{1-q} \\ &= 15 \frac{1-2^{11}}{1-2} \\ &= 15\ (2^{11}-1) \\ &= 30705 \end{aligned}$$

2. En utilisant une formule du cours, calculer : $S_{20} = u_0 + u_1 + ... + u_{20}$.

$$\begin{aligned} S_{20} &= u_0 \frac{1-q^{21}}{1-q} \\ &= 15 \frac{1-2^{21}}{1-2} \\ &= 15\ (2^{21}-1) \\ &= 31457265 \end{aligned}$$

3. On considère la fonction python somme de paramètre n, qui renvoie la valeur de S_n.

```
def somme(n):
    u=15
    S=15
    for i in range(1,n+1):
        u=2*u
        S=S+u
    return S
```

Que renvoie somme(10) ?

```
>> somme(10)
30705
```

4. Utiliser la fonction python somme pour retrouver le résultat de la question **3.**

```
>> somme(20)
31457265
```

Exercice 5

On considère la suite (u_n) de premier terme $u_0 = 5$ et tel que $u_{n+1} = 7u_n - 5$ pour tout entier naturel n.

1. Compléter la fonction python somme2 suivante pour qu'elle renvoie la somme $u_0 + u_1 + ... + u_n$

```
def somme2(n):
    u=5
    S=5
    for i in range(1,n+1):
        u=7*u-5
        S=S+u
    return S
```

2. Utiliser cette fonction pour calculer $u_0 + u_1 + ... + u_{10}$.

```
>> somme2(10)
1373143580
```

On obtient $u_0 + u_1 + ... + u_{10} = 1373143580$

Exercice 6

Soit (v_n) la suite définie sur $\mathbb{N}$ par $v_n = n^2 + 3$.

1. Compléter la fonction python somme3 de paramètre n suivante pour qu'elle renvoie la somme $v_0 + v_1 + ... + v_n$

```
def somme3(n):
    S=0
    for i in range(n+1):
        S = S + i**2 + 3
    return S
```

2. Utiliser cette fonction pour calculer $v_0 + v_1 + ... + v_{10}$.

```
>> somme3(10)
418
```

On obtient $v_0 + u_1 + ... + v_{10} = 418$

IV. Un exercice classique

Exercice 7

Un coureur cycliste, Thomas a programmé un entraînement hebdomadaire afin de se préparer à une course qui aura lieu dans quelques mois. Son objectif est de parcourir une distance totale de 1400 km pendant la période d'entraînement de 20 semaines (de la semaine 1 à la semaine 20).

Pour tout $n \in \mathbb{N}^*$, on note u_n la distance parcourue par Thomas la n-ième semaine.
On a $u_1 = 40$ et pour tout $n \in \mathbb{N}^*$, $u_{n+1} = u_n + 4$.

Pour tout $n \in \mathbb{N}^*$, on note $d_n = u_1 + u_2 + ... + u_n$ la distance totale parcourue par Thomas à la fin de la n-ième semaine.

1. Compléter la fonction python distance de paramètre n qui renvoie la valeur d_n.

```
def distance(n):
    u=40
    d=40
    for i in range(1,n):
        u=u+4
        d=d+u
    return d
```

2. Utiliser la fonction suivante pour déterminer si Thomas atteint son objectif de 1400 km à la 20-ième semaine ?

```
>> distance(20)
1560
```

Au bout de la 20-ième semaine, il aura parcouru au total 1560 km, donc il aura atteint son objectif de 1400 km.

3. Compléter la fonction python objectif pour qu'elle renvoie la plus petite valeur de n telle que $d_n \geq 1400$. On pourra utiliser la fonction python distance.

```
def objectif():
    n=0
    while distance(n)<1400:
        n=n+1
    return n
```

4. Interpréter dans le contexte de l'exercice.

```
>> objectif()
19
```

Il aura atteint son objectif de 1400 km au bout de la 19-ième semaine.