Corrige-Oral-Bac

May 3, 2020

1 Corrigés des exercices de préparation à l'oral du Bac

Les énoncés des exercices se trouvent sur :
-version pdf
-version Github markdown
-version diaporama HTML
-version simple HTML
-corrigé mis à jour régulièrement

1.1 Exercice 1

Auteur : Aude Durand Senter, question nř178 Genumsi

On a saisi le code suivant :

```
def mystere(nombre) :
    while nombre > 5 :
        nombre = nombre - 5
return nombre
```

Quelle affirmation est vraie dans celles proposées ci-dessous ?
Réponses :

1. On sort de la boucle while dès que `nombre > 5`
2. On sort de la boucle while dès que `nombre < 5`
3. On sort de la boucle while dès que `nombre <= 5`
4. On continue la boucle tant que `nombre <= 5`

Réponse : 3. On sort de la boucle while dès que `nombre <= 5`

1.2 Exercice 2

Auteur : Nicolas Revéret

On considère le programme ci-dessous :

```
a = 8
b = 5
a = a + b
b = a - b
a = a - b
```

Quelles sont les valeurs des variables a et b à la fin du programme ?
Réponse : 3. En sortie de boucle, les valeurs de a et b sont échangées

```
In [1]: a = 8
        b = 5
        print(f'a = {a} et b = {a}')
        a = a + b
        print(f'a = {a} et b = {a}')
        b = a - b
        print(f'a = {a} et b = {a}')
        a = a - b
        print(f'a = {a} et b = {a}')
```

```
a = 8 et b = 8
a = 13 et b = 13
a = 13 et b = 13
a = 5 et b = 5
```

1.3 Exercice 3

Auteur : Nicolas Revéret*
On souhaite que l'utilisateur saisisse une valeur entière comprise entre 1 et 20 (inclus).
Quel code est correct ?
Réponses :

1. Réponse 1

```
a = int ( input("Saisir un nombre entre 1 et 20") )
while (a < 1 and a > 20) :
      a = int ( input("Saisir un nombre entre 1 et 20") )
```

2. Réponse 2

```
while (a < 1 and a > 20) :
    a = int ( input("Saisir un nombre entre 1 et 20") )
```

3. Réponse 3

```
a = int ( input("Saisir un nombre entre 1 et 20") )
while (a < 1 or a > 20) :
    a = int ( input("Saisir un nombre entre 1 et 20") )
```

Réponse : 3.

```
a = int ( input("Saisir un nombre entre 1 et 20") )
while (a < 1 or a > 20) :
    a = int ( input("Saisir un nombre entre 1 et 20") )
```

1.4 Exercice 4

Auteur : Nicolas Revéret

On a saisi le code suivant :

```
a = 12
for i in range(3) :
  a = a * 2
  a = a - 10
```

Quelle est la valeur de la variable `a` après l'exécution du code ?

Réponse : En sortie de boucle `a = 26`.

```
In [3]: a = 12
        for i in range(3) :
          a = a * 2
          a = a - 10
        print(a)
```

```
26
```

1.5 Exercice 5

Pour les questions suivantes, il faut utiliser obligatoirement deux boucles imbriquées et au maximum deux fois la fonction `print`.

1. Ecrire un script Python qui produit l'affichage 1 ci-dessous.
2. Ecrire un script Python qui produit l'affichage 2 ci-dessous.

Affichage 1

```
0
01
012
0123
01234
```

Affichage 2

```
01234
12345
23456
34567
45678
```

Réponse Affichage 1 :

```
In [6]: for i in range(5):
            for j in range(0, i + 1):
                print(j, end='')
            print()
```

```
0
01
012
0123
01234
```

Réponse Affichage 2 :

```
In [7]: for i in range(5):
            for j in range(i, i + 5):
                print(j, end='')
            print()
```

```
01234
12345
23456
34567
45678
```

1.6 Exercice 6

Pour le diplôme du baccalauréat, si on note m la moyenne du candidat, quatre mentions sont possibles : *Passable* si $10 \leqslant m < 12$, *Assez bien* si $12 \leqslant m < 14$, *Bien* si $14 \leqslant m < 16$ et *Très bien* sinon. Recopier et compléter le script Python ci-dessous pour qu'il affiche la mention d'un candidat admis (on suppose sa moyenne supérieure ou égale à 10).

```
m = float(input('Moyenne du candidat ? '))
if 10 <= m < 12:
    print("Passable")
#to be completed
```

Réponse :

```
In [3]: m = float(input('Moyenne du candidat ? '))
        if 10 <= m < 12:
            print("Passable")
        elif 12 <= m < 14:
            print("Assez Bien")
        elif 14 <= m < 16:
            print("Bien")
        else:
            print("Très Bien")
```

```
Moyenne du candidat ? 15
Bien
```

1.7 Exercice 7

On considère le code suivant :

```
def f(tab):
  for i in range(len(tab)//2):
    tab[i],tab[-i-1] = tab[-i-1],tab[i]
```

Après les lignes suivantes :

```
tab = [2,3,4,5,7,8]
f(tab)
```

Quelle est la valeur de la variable `tab` ?

Réponse : La fonction `f` modifie sur place la liste `tab` et inverse l'ordre des éléments. Après `f(tab)`, `tab` contient la liste `[8,7,5,4,3,2]`

1.8 Exercice 7

Auteur : Nicolas Revéret

On considère le code suivant :

```
def f(tab):
  for i in range(len(tab)//2):
    tab[i],tab[-i-1] = tab[-i-1],tab[i]
```

Après les lignes suivantes :

```
tab = [2,3,4,5,7,8]
f(tab)
```

Quelle est la valeur de la variable `tab` ?

Réponses :

1. `[2,3,4,5,7,8]`
2. `[5,7,8,2,3,4]`
3. `[8,7,5,4,3,2]`
4. `[4,3,2,8,7,5]`

Réponse : cette fonction retourne la liste inversée `[8,7,5,4,3,2]` et donc la réponse est la 3.

1.9 Exercice 8

Auteur : Christophe Beasse

Que contient la variable `a` si on exécute ce script ?

```
def carre(val):
  return val*val

def inc(val):
  return val + 1

a = carre(inc(3.0))
```

1. 9.0
2. 12.0
3. 10.0
4. 16.0

Réponse : 4. 16

1.10 Exercice 9

Auteur : Christophe Beasse

Soit la liste suivante : `liste_pays =["France","Allemagne","Italie","Belgique"]`
Que renvoie : `liste_pays[2]` ?
Réponses :

1. `"France"`
2. `"Allemagne"`
3. `"Italie"`
4. `"Belgique"`

Réponse : 3. “Italie”

1.11 Exercice 10

_ *Auteur : Christophe Beasse*

Quelle est le résultat de : `[ (a,b) for a in range(3) for b in range(a)]` ?
Réponses :

1. `[(1,0),(2,1),(2,1)]`
2. `[(1,0),(2,1),(3,2)]`
3. `[(1,0),(2,0),(2,1)]`
4. `[(0,0),(1,1),(2,2)]`

Réponse : 3. `[(1,0),(2,0),(2,1)]`

```
In [8]: [ (a,b) for a in range(3) for b in range(a)]

Out[8]: [(1, 0), (2, 0), (2, 1)]
```

1.12 Exercice 11

Ecrire une fonction `moyenne(t)` qui prend en argument un tableau de nombres t et qui retourne sa moyenne arithmétique.

Réponse : ci-dessous

```
In [9]: def moyenne(t):
            s = 0
            for e in t:
                s = s + e
            return s/ len(t)
```

1.13 Exercice 12

Auteur : Nicolas Revéret

On dispose d'un tableau d'entiers, ordonné en ordre croissant.
On désire connaître le nombre de valeurs distinctes contenues dans ce tableau.
Quelle est la fonction qui ne convient pas ?
Réponses :

1. Réponse 1

```
def compte(t):
    cpt = 1
    for i in range(1,len(t)):
        if t[i] != t[i-1]:
            cpt = cpt + 1
    return cpt
```

2. Réponse 2

```
def compte(t):
  cpt = 0
  for i in range(0,len(t)-1):
    cpt = cpt + int(t[i] != t[i+1])
  return cpt
```

3. Réponse 3

```
def compte(t):
  cpt = 0
  for i in range(0,len(t)-1):
    if t[i] != t[i+1]:
      cpt = cpt + 1
  return cpt+1
```

Corrigé: la bonne réponse est la 3, voir ci-dessous

```
In [4]: def compte(t):
            cpt = 1
            for i in range(1,len(t)):
                if t[i] != t[i-1]:
                    cpt = cpt + 1
                print(t[i], t[i-1], cpt)
            return cpt


        def compte2(t):
            cpt = 0
            for i in range(0,len(t)-1):
             #cpt = cpt + int(t[i] != t[i+1])
```

```
            if t[i] != t[i+1]:
                cpt = cpt + 1
            return cpt

        def compte3(t):
            cpt = 0
            for i in range(0,len(t)-1):
                if t[i] != t[i+1]:
                    cpt = cpt + 1
            return cpt + 1
```

```
In [5]: compte([1,1,2,3,3,4])
```

```
1 1 1
2 1 2
3 2 3
3 3 3
4 3 4
```

```
Out[5]: 4
```

```
In [6]: 0 != 2
```

```
Out[6]: True
```

```
In [7]: 0 == 2
```

```
Out[7]: False
```

```
In [8]: int(0 != 2)
```

```
Out[8]: 1
```

```
In [9]: int( 0 == 2)
```

```
Out[9]: 0
```

1.14 Exercice 13

Auteur : Eric Rougier

Quel est le résultat de l'évaluation de l'expression Python suivante ?
`[2 ** n for n in range(4)]`
Réponses :

1. `[0, 2, 4, 6, 8]`
2. `[1, 2, 4, 8]`
3. `[0, 1, 4, 9]`
4. `[1, 2, 4, 8, 16]`

Corrigé : La réponse est la 2.

```
In [3]: [2 ** n for n in range(4)]
```

```
Out[3]: [1, 2, 4, 8]
```

1.15 Exercice 14

Auteur : Germain Becker, question nř326 Genumsi

Quel est le tableau `t` construit par les instructions suivantes ?

```
tab = [1, 2, -3, 7, 4, 10, -1, 0]
t = [e for e in tab if e >= 0]
```

Réponses :

1. `t = [1, 2, 7, 4, 10, 0]`
2. `t = [e, e, e, e, e, e]`
3. `t = [1, 2, 7, 4, 10]`
4. `t = [-3, -1, 0]`

Corrigé : La réponse est la 3.

```
In [5]: tab = [1, 2, -3, 7, 4, 10, -1, 0]
        t = [e for e in tab if e >= 0]
        t

Out[5]: [1, 2, 7, 4, 10, 0]
```

1.16 Exercice 15

Auteur : Germain Becker, question nř339 Genumsi

On considère le tableau t suivant.
`t = [[1, 2, 3], [2, 3, 4], [3, 4, 5], [4, 5, 6]]`
Quelle est la valeur de t[1][2] ?
Réponses :

1. 1
2. 3
3. 4
4. 2

Corrigé : La réponse est la 3.

```
In [8]: t = [[1, 2, 3], [2, 3, 4], [3, 4, 5], [4, 5, 6]]
        t[1][2]

Out[8]: 4
```

1.17 Exercice 16

Auteur : Eric Rougier, question nř150 Genumsi

Quelle est la valeur de la variable image après exécution du script Python suivant ?

```
image = [[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
for i in range(4):
    for j in range(4):
        if (i+j) == 3:
            image[i][j] = 1
```

Réponses :

1. `[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [1, 1, 1, 1]]`
2. `[[0, 0, 0, 1], [0, 0, 0, 1], [0, 0, 0, 1], [0, 0, 0, 1]]`
3. `[[0, 0, 0, 1], [0, 0, 1, 0], [0, 1, 0, 0], [1, 0, 0, 0]]`
4. `[[0, 0, 0, 1], [0, 0, 1, 1], [0, 1, 1, 1], [1, 1, 1, 1]]`

Corrigé : La réponse est la 3.

```
In [11]: image = [[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
         for i in range(4):
             for j in range(4):
                 if (i+j) == 3:
                     image[i][j] = 1
         print(image)
```

```
[[0, 0, 0, 1], [0, 0, 1, 0], [0, 1, 0, 0], [1, 0, 0, 0]]
```

1.18 Exercice 17

Auteur : Nicolas Réveret, question nř379 Genumsi

On a saisi le code suivant :

```
liste = [0, 1, 2, 3]
compteur = 0
for i in range(len(liste)-1) :
    for j in range(i,len(liste)) :
        compteur += 1
```

Que contient la variable compteur à la fin de l'exécution de ce script ?

Corrigé : La variable `compteur` contient $4 + 3 + 2 = 9$

```
In [13]: liste = [0, 1, 2, 3]
         compteur = 0
         for i in range(len(liste)-1) :
             for j in range(i,len(liste)) :
                 compteur += 1
         print(compteur)
```

```
9
```

1.19 Exercice 18

Quelle est la valeur référencée par la liste L après exécution du programme ci-dessous ?

```
L = [731, 732, 734]
L[0], L[1] = L[1], L[0]
M = L
M[1] = 732
```

1. [732, 731, 734]
2. [731, 732, 734]
3. [732, 732, 734]

Corrigé : la valeur référencée par la liste L après exécution du programme est `[732, 732, 734]`

```
In [14]: L = [731, 732, 734]
         L[0], L[1] = L[1], L[0]
         M = L
         M[1] = 732
         print(M)
```

```
[732, 732, 734]
```

1.20 Exercice 19

On considère l'extrait de code suivant :

```
while (a < 20) or (b > 50):
    .......
    .......
```

Quelles conditions permettent de mettre fin à cette boucle ?

1. **Réponse 1 :** la boucle prend fin lorsque `a < 20 ou b > 50`
2. **Réponse 2 :** la boucle prend fin lorsque `a < 20 et b > 50`
3. **Réponse 3 :** la boucle prend fin lorsque `a >= 20 ou b <= 50`
4. **Réponse 4 :** la boucle prend fin lorsque `a >= 20 et b <= 50`

Corrigé : La condition de sortie de boucle est la négation de la condition d'entrée de boucle, donc **réponse 4**

1.21 Exercice 20

On exécute le script suivant :

```
L = [12,0,8,7,3,1,5,3,8]

a = [elt for elt in L if elt < 4]
```

Quelle est la valeur de a à la fin de son exécution ?
Réponses :

1. **Réponse 1 :** `[12,0,8]`
2. **Réponse 2 :** `[12,0,8,7]`
3. **Réponse 3 :** `[0,3,1,3]`
4. **Réponse 4 :** `[0,3,1]`

Corrigé : La liste en compréhension sélectionné les éléments de `L` qui sont inférieurs à 4, donc la réponse est la 3 `[0,3,1,3]`

```
In [15]: L = [12,0,8,7,3,1,5,3,8]

         a = [elt for elt in L if elt < 4]

         print(a)
```

```
[0, 3, 1, 3]
```

1.22 Exercice 21

On définit : `L = [10,9,8,7,6,5,4,3,2,1]`.
Quelle est la valeur de `L[L[3]]` ?
Réponses :

1. **Réponse 1 :** 3
2. **Réponse 2 :** 4
3. **Réponse 3 :** 7

Corrigé : `L[3]` a pour valeur 7 donc `L[L[3]]` retourne l'élément d'index 7 donc le huitième donc 3

```
In [16]: L = [10,9,8,7,6,5,4,3,2,1]
         print(L[L[3]])
```

```
3
```

1.23 Exercice 22

On définit une grille G remplie de 0, sous la forme d'une liste de listes, où toutes les sous-listes ont le même nombre d'éléments.

```
G = [ [0, 0, 0, ..., 0],
[0, 0, 0, ..., 0],
[0, 0, 0, ..., 0],

......
[0, 0, 0, ..., 0]]
```

On appelle *hauteur* de la grille le nombre de sous-listes contenues dans G et *largeur* de la grille le nombre d'éléments dans chacune de ces sous-listes. Comment peut-on les obtenir ?

Réponses :

1. **Réponse 1:** `python hauteur = len(G[0]) largeur = len(G)`
2. **Réponse 2 :** `python hauteur = len(G) largeur = len(G[0])`
3. **Réponse 3 :**

   ```
   hauteur = len(G[0])
   largeur = len(G[1])
   ```

4. **Réponse 4 :** `python hauteur = len(G[1]) largeur = len(G[0])`

Corrigé : La réponse est la 2.

1.24 Exercice 23

Quelle est la valeur de l'expression `[[0] * 3 for i in range(2)]` ?

Réponses :

1. **Réponse 1 :** `[[0,0], [0,0], [0,0]]`
2. **Réponse 2 :** `[[0,0,0], [0,0,0]]`
3. **Réponse 3 :** `[[0.000], [0.000]]`
4. **Réponse 4 :** `[[0.00], [0.00], [0.00]]`

Corrigé : La réponse est la 2.

1.25 Exercice 24

On exécute le script suivant :

```
a = [1, 2, 3]
b = [4, 5, 6]
c = a + b
```

Que contient la variable `c` à la fin de cette exécution ?

Réponses :

1. **Réponse 1 :** `[5,7,9]`
2. **Réponse 2 :** `[1,4,2,5,3,6]`
3. **Réponse 3 :** `[1,2,3,4,5,6]`
4. **Réponse 4 :** `[1,2,3,5,7,9]`

Corrigé : La réponse est la 3, l'opérateur + réalise la concaténation des deux listes.

1.26 Exercice 25

On exécute le script suivant :

```
asso = []
L=[['marc','marie'],['marie','jean'],
    ['paul','marie'], ['marie','marie'],
    ['marc','anne']]
for c in L :
    if c[1]=='marie':
        asso.append(c[0])
```

Que vaut asso à la fin de l'exécution ?
Réponses :

1. **Réponses 1:** `['marc', 'jean', 'paul']`
2. **Réponses 2 :** `[['marc','marie'], ['paul','marie'], ['marie','marie']]`
3. **Réponses 3 :** `['marc', 'paul', 'marie']`
4. **Réponses 4 :** `['marie', 'anne']`

Corrigé : La réponse est la 3.

```
In [17]: asso = []
         L=[['marc','marie'],['marie','jean'],
             ['paul','marie'], ['marie','marie'],
             ['marc','anne']]
         for c in L :
             if c[1]=='marie':
                 asso.append(c[0])
         print(asso)

['marc', 'paul', 'marie']
```

1.27 Exercice 26

Quelle est la valeur de la variable n à la fin de l'exécution du script ci-dessous ?

```
n = 1
while n != 20:
    n = n + 2
```

Réponses :

1. **Réponse 1:** 1
2. **Réponse 2:** 20
3. **Réponse 3:** 22
4. **Réponse 4:** le programme ne termine pas, la boucle tourne indéfiniment

Corrigé : La réponse est la 4, la variable `n` ne prend comme valeurs que des entiers impairs donc elle n'atteindra jamais la valeur 20.

1.28 Exercice 27

On définit en Python la fonction suivante :

```
def f(L):
    S = []
    for i in range(len(L)-1):
        S.append(L[i] + L[i+1])
    return S
```

Quelle est la liste renvoyée par `f([1, 2, 3, 4, 5, 6])` ?

1. `[3, 5, 7, 9, 11, 13]`
2. `[1, 3, 5, 7, 9, 11]`
3. `[3, 5, 7, 9, 11]`
4. cet appel de fonction déclenche un message d'erreur

Corrigé : La réponse est la 3, la fonction retourne une liste `S` contenant les sommes de tous les couples d'éléments d'index successifs.

```
In [18]: def f(L):
             S = []
             for i in range(len(L)-1):
                 S.append(L[i] + L[i+1])
             return S

         print(f([1, 2, 3, 4, 5, 6]))
```

```
[3, 5, 7, 9, 11]
```

1.29 Exercice 28

Le codage en base deux de l'entier 26 en base dix est :

1. 11010
2. 10010
3. 11001
4. 110010

Corrigé : La réponse est la 1.

```
In [10]: bin(26)

Out[10]: '0b11010'
```

1.30 Exercice 29

Le résultat de la somme $\overline{101101}^2 + \overline{101111}^2$ est :

1. $\overline{1100100}^2$
2. $\overline{1110101}^2$
3. $\overline{1011100}^2$
4. $\overline{1111100}^2$

Corrigé : La réponse est la 3.

```
In [12]: 0b101101 + 0b101111

Out[12]: 92

In [13]: bin(92)

Out[13]: '0b1011100'
```

1.31 Exercice 30

Le plus grand entier qu'on peut représenter en base deux sur 8 bits a pour écriture décimale :

1. 11111111
2. 10000000
3. 255
4. 256

Corrigé : L'écriture binaire du plus grand entier représentable en base deux sur 8 bits est 11111111 et son écriture décimale est $2^8 - 1 = 255$.

1.32 Exercice 31

Quelle est l'écriture décimale de l'entier qui s'écrit 1010 en binaire ?

Réponses :

1. 5
2. 10
3. 20
4. 22

Corrigé : L'écriture binaire 1010 correspond à l'écriture décimale $2^3 + 2^1 = 10$.

1.33 Exercice 32

Combien d'entiers positifs ou nuls (entiers non signés) peut-on représenter en machine sur 32 bits ?

Réponses :

1. $2^{32} - 1$
2. 2^{32}
3. 2×32
4. 32^2

Corrigé : Sur 32 bits, on peut représenter 2^{32} entiers.

1.34 Exercice 33

Combien de bits faut-il au minimum pour coder le nombre décimal 4085 ?

Réponses :

1. 4
2. 12
3. 2042
4. 2043

Corrigé : $2^{11} = 2048$ et $2^{12} = 4096$ donc $2^{11} \leqslant 4085 < 2^{12}$ donc il faut 12 bits au minimum pour représenter 4085.

1.35 Exercice 34

Parmi les propositions suivantes, laquelle est la représentation binaire de 761 ?

Réponses :

1. 11 1100 1101
2. 11 1110 0101
3. 10 0111 1001
4. 10 1111 1001

Corrigé : Réponse 4.

```
In [19]: bin(761)
```

```
Out[19]: '0b1011111001'
```

1.36 Exercice 35

Le codage d'une couleur se fait à l'aide de trois nombres compris chacun, en écriture décimale, entre 0 et 255 (code RVB).

La couleur ń vert impérial ż est codée, en écriture décimale, par (0,86, 27).

Le codage hexadécimal (base 16) correspondant est :

Réponses :

1. (0, 134, 39)
2. (0, 134, 1B)
3. (0, 56, 1B)
4. (0, 56, 39)

Corrigé : Réponse 3.

```
In [20]: [hex(c) for c in (0,86,27)]
```

```
Out[20]: ['0x0', '0x56', '0x1b']
```

1.37 Exercice 36

Quelle est l'écriture binaire du nombre entier 183 ?

Réponses :

1. 0100 1000
2. 1110 1101
3. 1011 0111
4. 1001 0101

Corrigé : Réponse 3.

```
In [22]: bin(183)
```

```
Out[22]: '0b10110111'
```

1.38 Exercice 37

Le codage en base seize de l'entier 255 est :

1. $\overline{F1}^{16}$
2. $\overline{A1}^{16}$
3. $\overline{FA}^{16}$
4. $\overline{FF}^{16}$

Corrigé : Réponse 4.

```
In [24]: hex(255   )
```

```
Out[24]: '0xff'
```

1.39 Exercice 38

L'adresse MAC de la carte Wifi d'un smartphone est `c8:60:00:a4:89:ab` avec six octets codés en base seize.

La transcription en base dix de cette adresse MAC est :

1. 200 : 96 : 0 : 164 : 137 : 171
2. 20 : 6 : 0 : 14 : 17 : 21
3. 96 : 0 : 0 : 40 : 72 : 110
4. 140 : 6 : 0 : 74 : 152 : 186

Corrigé : Réponse 1.

```
In [29]: ':'.join([str(int(c, 16)) for c in 'c8:60:00:a4:89:ab'.split(':')])

Out[29]: '200:96:0:164:137:171'
```

1.40 Exercice 39

La fonction ci-dessous doit retourner la liste des chiffres en base deux d'un entier n par ordre décroissant des poids de gauche à droite.

```
def nombre2chiffres(n):
    t = []
    while n >= 2:
        ..........
        n = n // 2
    ..............
    t.reverse()
    return t
```

Quelle instruction peut-on écrire en lignes 4 et 6 ?

1. `t.append(n)`
2. `t.append(n % 2)`
3. `t.append(n // 2)`
4. `t = t + [n % 2]`

Corrigé : Réponse 2.

1.41 Exercice 40

1. Représenter en binaire le nombre d'écriture décimale 49.
2. Représenter en base dix, le nombre dont l'écriture en base deux est 1010110.
3. Déterminer le successeur en base deux des entiers :
   - 111
   - 10011
   - 10111
4. Déterminer le nombre de caractères qu'on peut coder sur un octet.

Corrigé :

1. En binaire le nombre d'écriture décimale 49 s'écrit 110001.
2. Le nombre dont l'écriture en base deux est 1010110 a pour écriture décimale 86.
3. Déterminer le successeur en base deux des entiers :
   - 111 a pour successeur 1000
   - 10011 a pour successeur 10100
   - 10111 a pour successeur 11000
4. Sur un octet, soit 8 bits, on peut code $2^8 = 256$ caractères.

```
In [30]: bin(49)

Out[30]: '0b110001'

In [31]: 0b1010110

Out[31]: 86
```

1.42 Exercice 41

On souhaite convertir 25 de base 10 en base 2. On obtient en binaire :

Réponses :

1. 11001
2. 10110
3. 10011
4. 11000

Corrigé : Réponse 1

```
In [32]: bin(25)

Out[32]: '0b11001'
```

1.43 Exercice 42

Quelle est la valeur affichée par l'exécution du test suivant ?

```
In [1]: 0.1 + 0.2 == 0.3

Out[1]: False
```

Réponses :

1. `True`
2. `False`
3. 0.3

Corrigé : Réponse 22. Les décimaux `0.1`, `0.2` et `0.3` sont représentés de façon approchée

```
In [33]: import decimal

In [40]: decimal.Decimal.from_float(0.1)

Out[40]: Decimal('0.1000000000000000055511151231257827021181583404541015625')

In [41]: decimal.Decimal.from_float(0.2)

Out[41]: Decimal('0.200000000000000011102230246251565404236316680908203125')

In [42]: decimal.Decimal.from_float(0.3)

Out[42]: Decimal('0.29999999999999998889776975374843459576368331909179687 5')
```

1.44 Exercice 43

Combien de bits sont nécessaires pour représenter 15 en binaire ?

Réponses :

1. 2
2. 3
3. 4
4. 5

Corrigé : $2^3 \leqslant 15 < 2^4$ donc il faut 4 bits pour représenter le décimal 15 en binaire.

1.45 Exercice 44

Quelle est la valeur affichée a l'exécution du programme Python suivant ?

```
x = 1
for i in range(11):
    x = x * 2
print(x)
```

Réponses :

1. 1024
2. 2048
3. 23
4. $2^{2^{11}}$

Corrigé : $2^3 \leqslant 15 < 2^4$ donc il faut 4 bits pour représenter le décimal 15 en binaire.

1.46 Exercice 45

Quelle est la valeur affchée a l'exécution du programme Python suivant ?

```
x = 2
for i in range(10):
    x = x ** 2
print(x)
```

Réponses :

1. 1024
2. 2048
3. $2^{2^{10}}$
4. $2^{2^{11}}$

Corrigé : Réponse 3 $2^{2^{10}}$

```
In [44]: x = 2
         for i in range(10):
             x = x ** 2
         print(x == 2 ** (2 ** 10))
```

```
True
```

1.47 Exercice 46

1. Convertir 3970 en base 6.
2. Convertir en base dix l'entier naturel dont l'écriture en base six est 4321.
3. Écrire en Python une fonction `base6(L)` qui renvoie la valeur entière correspondant à la liste des chiffres de l'écriture en base 6.

Exemple : `base6([1, 3, 2])` doit renvoyer 56 car $1 \times 6^2 + 3 \times 6 + 2 = 56$
Corrigé :

1. Le décimal 3970 s'écrit 30214 en base 6.
2. L'entier dont l'écriture en base six est 4321, s'écrit 985 en base dix.
3. Voir ci-dessous.

```
In [47]: def base10To6(n):
             """Convertit un décimal en base 6"""
             dec = []
             while n >= 6:
                 dec.append(n % 6)
                 n = n // 6
             dec.append(n % 6)
             return dec[::-1]
```

```
In [48]: base10To6(985)
```

```
Out[48]: [4, 3, 2, 1]

In [49]: base10To6(3970)

Out[49]: [3, 0, 2, 1, 4]

In [45]: def base6(L):
             n = 0
             for c in L:
                 n = 6 * n + c
             return n

In [46]: base6([4,3,2,1])

Out[46]: 985
```

1.48 Exercice 47

1. Pour déterminer la liste des chiffres en base dix d'un entier naturel, un élève a écrit la fonction ci-dessous : ~~python def liste_chiffres(n): L = [n % 10] while n > 0: n = n // 10 L.insert(0, n % 10) return L~~

   Malheureusement sa fonction ne retourne pas le résultat attendu pour l'entier 730 :

   ```
   >>> liste_chiffres(730)
   [0, 7, 3, 0]
   ```

   Proposer une version corrigée de la fonction `liste_chiffres`.

2. Compléter la fonction `somme_chiffres_base2(n)` pour qu'elle retourne la somme des chiffres en base deux de l'entier `n` passé en paramètre.

   ```
   def somme_chiffres_base2(n):
       s = 0
       while n > 0:
           s = s + n % 2
           ..............
       return s
   ```

3. Déterminer une valeur possible de la variable `secret` telle que :

   ```
   >>> somme_chiffres_base2(secret)
   734
   ```

4. Écrire une fonction `maximum_chiffre(n)` qui retourne le plus grand chiffre de l'écriture en base dix de l'entier naturel `n` passé en paramètre.

Corrigé :

1. Voir ci-dessous.
2. Voir ci-dessous.

3. Il suffit de prendre le nombre avec 734 fois le chiffre 1 dans son écriture binaire qui est $2^{734} - 1$ en décimal.

```
In [51]: def liste_chiffres(n):
             L = [n % 10]
             while n >= 10:
                 n = n // 10
                 L.insert(0, n % 10)
             return L

In [52]: liste_chiffres(730)

Out[52]: [7, 3, 0]

In [53]: def somme_chiffres_base2(n):
             s = 0
             while n > 0:
                 s = s + n % 2
                 n = n // 2
             return s

In [54]: somme_chiffres_base2(2 ** 734 - 1)

Out[54]: 734

In [57]: def maximum_chiffre(n):
             """Plus grand chiffre en base 10"""
             cmax = 0
             while n > 0:
                 c = n % 10
                 if c > cmax:
                     cmax = c
                 n = n // 10
             return cmax

In [58]: maximum_chiffre(40756)

Out[58]: 7
```

1.49 Exercice 48

Le nombre de chiffres en base 2 d'un entier naturel n est :

- 1 si n est égal à 0 ;
- l'unique entier p tel que $2^{p-1} \leqslant n < 2^p$ si n supérieur à 0.

Compléter la fonction ci-dessous pour qu'elle retourne le nombre de chiffres en base 2 de l'entier `n` passé en paramètres.