Exercice 1
Dans la représentation normalisée des nombres flottant sur 32 bits :

$$x = \varepsilon \times m \times 2^e$$
$$\varepsilon = \pm 1 \quad ; \quad m \in [1, 2[\quad ; \quad e \in \mathbb{Z}$$

- Le bit de poids fort représente le signe : `1` pour négatif, `0` pour positif,
- Les 8 bits suivants représentent l'écriture binaire de $e + 127$.
- les 23 bits suivants représentent l'écriture binaire après la virgule de `m-1`.

1. Comment s'écrit le nombre `-1.0` ?
2. Comment s'écrit le nombre `1.5` ?
3. Comment s'écrit le nombre `4.75` ?
4. Quel nombre représente :

   1 01111111 11000000000000000000000

5. Quel nombre représente :

   0 10000001 00100000000000000000000

6. Quel nombre représente :

   1 01111110 10000000000000000000000

Exercice 2
Racines d'un polynôme de degré 2

1. Reprendre la fonction `trinome(a,b,c)` écrite au TD3 prenant en paramètres 3 nombres à virgule flottante `a`, `b`, `c` et qui à l'aide du discriminant retourne la ou les racines du polynôme de degré 2 : $P(X) = aX^2 + bX + c$.
2. Appliquer la recherche de racines au polynôme :

   $$X^2 + X + \frac{1 + 2^{-53}}{4}$$

   en appelant `racine_trinome(1,1,(1+2**-53)/4)`.
3. Définir la variable `e = -53`, et lancer l'instruction de comparaison avec 0 :
   ```
   >>> e = - 53
   >>> 1 + 2**e - 1 == 0
   ```
   Recommencer après l'affectation : `e = -52`. Discuter de la validité du résultat trouvé à la question précédente.
4. Faire de même avec le polynôme :

   $$X^2 - \sqrt{2}X + \frac{1}{2}$$

   Le résultat retourné est-il correct ?

Exercice 3

1. Quel est le plus petit nombre > 1 qui peut être représenté sur un ordinateur 64 bits ? Quels sont les 2 suivants ?
2. Quel est le plus petit nombre > 2 qui peut être représenté sur un ordinateur 64 bits ? Quels sont les 2 suivants ?
3. Quel est le plus petit nombre > 4 qui peut être représenté sur un ordinateur 64 bits ? Quels sont les 2 suivants ?
4. Quel est le plus petit nombre $> 1/2$ qui peut être représenté sur un ordinateur 64 bits ? Quels sont les 2 suivants ?
5. Combien de nombres entre 1 inclu et 2 exclu peut -on représenter ? Combien entre 2 et 4 ? Combien entre 1/2 et 1 ?

Exercice 4
Dans la représentation des réels en virgule flottante sur 8 bits, on utilise :
– 1 bit, celui de poids le plus fort, pour le signe,
– les 2 bits suivants pour l'exposant $e \in [[-1, 2]]$. On code $e + 1$ sur 2 bits

en binaire.
– les 5 bits restants pour la mantisse.
Le principe du codage est par ailleurs le même que dans l'écriture normalisée des nombres à virgules flottantes.

1. Comment représenter sur 8 bits le nombre $2,75$?
2. Quel nombre représente `'00111111'` ?
3. Quel est le plus petit nombre que l'on peut représenter ainsi ? Le plus grand ?

Exercice 5
Le code qui suit permet de tracer graphiquement tous les nombres réels se représentant graphiquement sur 8 bits (voir description à l'exo précédent).

1. Expliquer le rôle de chaque fonction.
2. L'exécuter. Après avoir "zoomé" sur certaines parties du graphiques, quelles constatations faites-vous ?

```
def binaire(n,N):
    s = ''
    while n>0:
        s = str(n%2) + s
        n = n//2
    return '0'*(N-len(s)) + s

def bin2dec(s):
    r = 0
    for c in s:
        r *= 2
        r += int(c)
    return r

S = bin2dec('10000000') # pour récupérer le bit de poids fort
E = bin2dec('01100000') # pour récupérer les bits de l'exposant
M = bin2dec('00011111') # pour récupérer les bits de la mantisse

def signe(n):
    if n & S == 0:
        return +1
    else:
        return -1

def exposant(n):
    e = (n & E) >> 5
    return e-1

def mantisse(n):
    m = M & n
    s = binaire(m,5)
    r = 1
    p = 1
    for i in range(5):
        p /= 2
        r += int(s[i]) * p
    return r

L = []
for n in range(256):
    L.append(signe(n)*mantisse(n)*2**exposant(n))
Y = [0] * len(L)
import matplotlib.pyplot as plt
plt.plot(L,Y,'.k')
plt.show()
```