Exercice 5

a) Écrire une fonction *complement(b)* qui calcule le complément à 2 d'un nombre binaire donné sous forme d'un tableau de taille quelconque contenant uniquement les chiffres 0 et 1. La fonction renverra un nouveau tableau, de même taille que b.

b) Écrire une fonction *evalue(t,b)* qui renvoie la valeur du nombre en base b dont les chiffres sont contenus dans le tableau t.

Par exemple, *evalue*([9,8,7],10) doit renvoyer 987

Principe : représentation des flottants dans un ordinateur

Reprenons l' exemple de l'activité avec la volonté de coller à la norme IEEE-754

$(10{,}59375)_{10}=(1010{,}10011)_2$ → RAPPEL : vous pouvez utiliser la fonction *reelPositifEnBinaire(valeur, bits)* crée

Dans $(1010{,}10011)_2$, le bit de signe est positif : donc ce sera 0.

Ensuite, on va décaler la virgule : $1010{,}10011=1{,}01010011\,.\,2^{11}$ en binaire $(11)_2=(3)_{10}$ correspond bien à un décalage de 3 rangs de la virgule en base 2 d'où le 2^{11}

Il faudra pouvoir aussi utiliser des puissances négatives : pour cela, on décale systématiquement les exposants de $(127)_{10}=(1111111)_2$. Or $(11)_2+(1111111)_2=(10000010)_2$ (en base 10, on aurait pu aussi écrire 127 + 3 = 130 et convertir 130 en base 2)

Notre nombre va s'écrire $1{,}01010011\,.\,2^{10000010}$ **On a alors (en « simple précision » sur 32 bits)**

0	1	0	0	0	0	0	1	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s	n…avec zéros ajoutés si besoin								m……avec zéros ajoutés pour faire 23 bits																						

Vous pouvez vérifier ce résultat sur https://baseconvert.com/ieee-754-floating-point

Exercice 6

On tape en Python l'expression `>>>(1e25+16) – 1e25`

Quel est le résultat attendu ? Quel est le résultat obtenu ? Pourquoi ?

Exercice 7

a) Encodez $(0{,}1)_{10}$ en utilisant ce qui a déjà été fait à l'exercice 3

b) Encodez de nouveau selon cette norme $(128)_{10}$ puis $(-32{,}75)_{10}$

Exercice 8

Appliquez ce principe à l'encodage de $(0{,}25)_{10}$ puis pour la « première fois » à une fraction $\left(\frac{1}{3}\right)_{10}$