

TD 9 : Les nombres Flottants

Problème de conversion

1. Écrire une fonction `arrondi()` et y inclure le code ci-dessous.

```
i = 0.
while i < 1:
    print(i)
    i += 0.1
```

2. Montrer que ce programme doit terminer. *le plus proche de 1*
3. Faire un appel à cette fonction dans le corps principal, lancer le programme et conclure.
4. Écrire une fonction `gentleman()` et y inclure le code ci-dessous.

```
i = 1
x = 1.
y = x + 1.
while y - x == 1.:
    x = x * 2.
    y = x + 1.
    i += 1
    print(i, x, y)
```

5. Montrer que ce programme ne doit pas terminer. *car y dépend de x*
6. Faire un appel à cette fonction dans le corps principal, lancer le programme et conclure.

Conversion des flottants

On souhaite transformer un nombre binaire décimal en base 10. Pour simplifier, on va déjà écrire un programme qui transforme la partie décimale en binaire, puis dans un deuxième temps on transformera la partie entière.

1. Créer une fonction avec le prototype suivant :
`decimale(binaire : str) -> float`
 - `binaire` : chaîne binaire de la partie décimale
 - la fonction doit renvoyer le nombre flottant correspondant
2. Ajouter une fonction qui convertisse en base 10 la partie entière du binaire. Le prototype est le suivant :

`entiere(binaire : str) -> int`

3. Écrire un programme `bin2float.py` qui convertisse la partie décimale d'un nombre binaire en flottant.

NB : on pourra utiliser la méthode `split()` de l'objet `str` pour traiter la partie décimale et la partie entière séparément.

4. Transformer $0,0101010101_2$ et $11100,10001_2$ en base 10.

0,3330078125

28,53125

TD 9 : Les nombres Flottants

On souhaite maintenant réaliser l'opération inverse et transformer un flottant en binaire. Pour cela, on va encore séparer le problème en deux sous problèmes : d'abord transformer la partie décimale, puis terminer avec la partie entière.

5. Écrire un programme float2bin qui convertisse la partie décimale d'un nombre flottant en binaire.

On donne l'algorithme :

algorithme float2bin

{ converti un nombre décimal < 1 en binaire }

variables

precision : entier := 16

decimal : réel := 0.347

binaire : chaîne[16] := ""

i : entier := 0

début

tant que decimal $\neq 0$ ET $i < \text{precision}$ faire

decimal := decimal * 2

si decimal < 1 alors

binaire := binaire + "0"

sinon

binaire := binaire + "1"

decimal := decimal modulo 1

i := i + 1

fin

6. Donner la signification de l'instruction : decimal := decimal modulo 1 *pour conserver uniquement la partie fractionnaire du dec*
7. Montrer que ce programme se termine. *quand i est supérieur à 16 car il prend 1 à chaque fois*
8. Tester le programme. Résultat attendu : 0101100011010100.
9. Modifier le programme précédent pour créer une fonction avec le prototype suivant :
bin_decimal(decimal : float, precision : int) -> str
 - décimal : partie décimale du nombre flottant
 - precision : précision souhaité dans le calcul
 - la fonction doit renvoyer la chaîne binaire correspondante
10. Transformer 0,5625 et 0,15 en base 2. *1001 0010011001100110*
11. Ajouter une fonction qui convertisse en base 2 la partie entière du nombre flottant. Le prototype est le suivant : bin_entiere(entier : int) -> str
12. Transformer 12,9 en base 2.

Challenge

Écrire un programme qui convertisse un nombre réel en binaire selon la norme IEEE 754.

