Représentation des flottants

Informatique pour tous

Représentations pour les différents types de nombres

La plupart des langages de programmation stockent:

1. Les entiers positifs avec leur écriture en base 2.
2. Les entiers relatifs avec leur codage par complément à 2.
3. Les flottants avec la norme IEEE-754.

Représentations pour les différents types de nombres

La plupart des langages de programmation stockent:

1. Les entiers positifs avec leur écriture en base 2.
2. Les entiers relatifs avec leur codage par complément à 2.
3. Les flottants avec la norme IEEE-754.

Le module de calcul scientifique **numpy** permet de spécifier comment on veut représenter un nombre.

```
In [10]: import numpy as np
```

Entiers positifs

Avec p bits, on peut représenter tous les entiers positifs de:

$$< \underbrace{00...00}_{p} >_2 \quad \text{à} \quad < \underbrace{11...11}_{p} >_2 =$$

Entiers positifs

Avec p bits, on peut représenter tous les entiers positifs de:

$$< \underbrace{00...00}_{p} >_2 \quad \text{à} \quad < \underbrace{11...11}_{p} >_2 = 2^p - 1$$

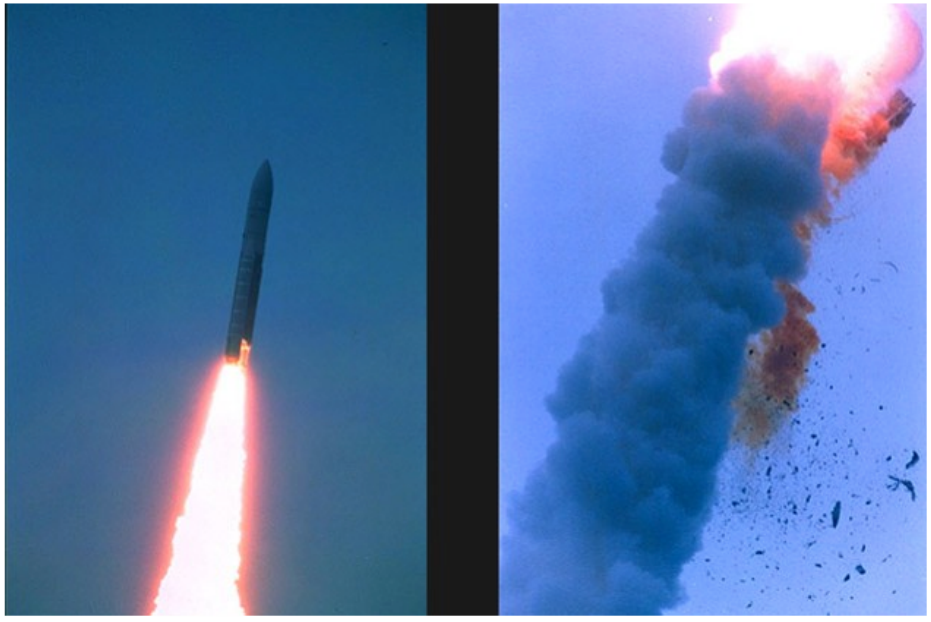
Dans numpy, les nombres positifs sont de type uint (unsigned int), suivi du nombre de bits p utilisé:

```
In [23]: np.uint32(2**32 - 1)
Out[23]: 4294967295

In [24]: np.uint32(2**32)
Out[24]: 0
```

Entiers positifs

C'est un dépassement d'entier qui a causé le crash d'Ariane 5!



Explosion d'Ariane 5 (1996).

Flottants

Question

Comment représenter les nombres à virgules?

Flottants

Question

Comment représenter les nombres à virgules?

Par définition, $0{,}1415 = 1 \times 10^{-1} + 4 \times 10^{-2} + 1 \times 10^{-3} + 5 \times 10^{-4}$.

Flottants

Question

Comment représenter les nombres à virgules?

Par définition, $0{,}1415 = 1 \times 10^{-1} + 4 \times 10^{-2} + 1 \times 10^{-3} + 5 \times 10^{-4}$.

Définition: développement en base b

$$< 0, x_1 x_2 ... >_b = x_1 \times b^{-1} + x_2 \times b^{-2} + ...$$

Flottants

Question

Comment représenter les nombres à virgules?

Par définition, $0{,}1415 = 1 \times 10^{-1} + 4 \times 10^{-2} + 1 \times 10^{-3} + 5 \times 10^{-4}$.

Définition: développement en base b

$$< 0, x_1 x_2 ... >_b = x_1 \times b^{-1} + x_2 \times b^{-2} + ...$$

Exemples: $< 0, 101 >_2 =$

Flottants

Question

Comment représenter les nombres à virgules?

Par définition, $0{,}1415 = 1 \times 10^{-1} + 4 \times 10^{-2} + 1 \times 10^{-3} + 5 \times 10^{-4}$.

Définition: développement en base b

$$< 0, x_1 x_2 ... >_b = x_1 \times b^{-1} + x_2 \times b^{-2} + ...$$

Exemples: $< 0, 101 >_2 = \frac{1}{2} + \frac{1}{2^3} = 0,625 \ (= < 0, 625 >_{10})$

Flottants

Question

Comment représenter les nombres à virgules?

Par définition, $0{,}1415 = 1 \times 10^{-1} + 4 \times 10^{-2} + 1 \times 10^{-3} + 5 \times 10^{-4}$.

Définition: développement en base b

$$< 0, x_1 x_2 ... >_b = x_1 \times b^{-1} + x_2 \times b^{-2} + ...$$

Exemples: $< 0, 101 >_2 = \frac{1}{2} + \frac{1}{2^3} = 0,625 \ (= < 0, 625 >_{10})$

$< 1, 01010101... >_2 =$

Flottants

Question

Comment représenter les nombres à virgules?

Par définition, $0{,}1415 = 1 \times 10^{-1} + 4 \times 10^{-2} + 1 \times 10^{-3} + 5 \times 10^{-4}$.

Définition: développement en base b

$$< 0, x_1 x_2 ... >_b = x_1 \times b^{-1} + x_2 \times b^{-2} + ...$$

Exemples: $< 0, 101 >_2 = \frac{1}{2} + \frac{1}{2^3} = 0,625 \ (= < 0, 625 >_{10})$

$$< 1, 01010101... >_2 = 1 + \frac{1}{2^2} + \frac{1}{2^4} + \frac{1}{2^6} \ ... = \frac{1}{4^0} + \frac{1}{4} + \frac{1}{4^2} + ...$$

Flottants

Question

Comment représenter les nombres à virgules?

Par définition, $0{,}1415 = 1 \times 10^{-1} + 4 \times 10^{-2} + 1 \times 10^{-3} + 5 \times 10^{-4}$.

Définition: développement en base b

$$< 0, x_1 x_2 ... >_b = x_1 \times b^{-1} + x_2 \times b^{-2} + ...$$

Exemples: $< 0, 101 >_2 = \frac{1}{2} + \frac{1}{2^3} = 0,625 \ (= < 0, 625 >_{10})$

$$< 1, 01010101... >_2 = 1 + \frac{1}{2^2} + \frac{1}{2^4} + \frac{1}{2^6} \ ... = \frac{1}{4^0} + \frac{1}{4} + \frac{1}{4^2} + ...$$

$$< 1, 01010101... >_2 = \frac{1}{1 - \frac{1}{4}} = \frac{4}{3}.$$

Flottants

Question

Comment passer d'un développement de $0 < x < 1$ en base 10 à un développement dans une autre base b?

On veut écrire x sous la forme $x = < 0, x_1 x_2 x_3 ... >_b$.

Flottants

Question

Comment passer d'un développement de $0 < x < 1$ en base 10 à un développement dans une autre base b?

On veut écrire x sous la forme $x = < 0, x_1 x_2 x_3 ... >_b$.

1. $x \times b =$

Flottants

Question

Comment passer d'un développement de $0 < x < 1$ en base 10 à un développement dans une autre base b?

On veut écrire x sous la forme $x = < 0, x_1 x_2 x_3 ... >_b$.

1. $x \times b = < x_1, x_2 x_3 ... >_b$, donc $x_1 =$

Flottants

Question

Comment passer d'un développement de $0 < x < 1$ en base 10 à un développement dans une autre base b?

On veut écrire x sous la forme $x = < 0, x_1 x_2 x_3 ... >_b$.

1. $x \times b = < x_1, x_2 x_3 ... >_b$, donc $x_1 = \lfloor x \times b \rfloor$.
2. Pour trouver x_2, on refait la même chose sur $x \times b - x_1 = < 0, x_2 x_3 ... >_b$.
3. ...

Flottants

Question

Comment passer d'un développement de $0 < x < 1$ en base 10 à un développement dans une autre base b?

On veut écrire x sous la forme $x = < 0, x_1 x_2 x_3 ... >_b$.

1. $x \times b = < x_1, x_2 x_3 ... >_b$, donc $x_1 = \lfloor x \times b \rfloor$.
2. Pour trouver x_2, on refait la même chose sur $x \times b - x_1 = < 0, x_2 x_3 ... >_b$.
3. ...

Exemples: $0{,}625 = < 0, ? >_2$

Flottants

Question

Comment passer d'un développement de $0 < x < 1$ en base 10 à un développement dans une autre base b?

On veut écrire x sous la forme $x = < 0, x_1 x_2 x_3 ... >_b$.

1. $x \times b = < x_1, x_2 x_3 ... >_b$, donc $x_1 = \lfloor x \times b \rfloor$.
2. Pour trouver x_2, on refait la même chose sur $x \times b - x_1 = < 0, x_2 x_3 ... >_b$.
3. ...

Exemples: $0{,}625 = < 0, 1? >_2$

1. $0{,}625 \times 2 = 1, 25$

Flottants

Question

Comment passer d'un développement de $0 < x < 1$ en base 10 à un développement dans une autre base b?

On veut écrire x sous la forme $x = < 0, x_1 x_2 x_3 ... >_b$.

1. $x \times b = < x_1, x_2 x_3 ... >_b$, donc $x_1 = \lfloor x \times b \rfloor$.
2. Pour trouver x_2, on refait la même chose sur $x \times b - x_1 = < 0, x_2 x_3 ... >_b$.
3. ...

Exemples: $0{,}625 = < 0, 10? >_2$

1. $0{,}625 \times 2 = 1, 25$
2. $0, 25 \times 2 = 0, 5$

Flottants

Question

Comment passer d'un développement de $0 < x < 1$ en base 10 à un développement dans une autre base b?

On veut écrire x sous la forme $x = < 0, x_1 x_2 x_3 ... >_b$.

1. $x \times b = < x_1, x_2 x_3 ... >_b$, donc $x_1 = \lfloor x \times b \rfloor$.
2. Pour trouver x_2, on refait la même chose sur $x \times b - x_1 = < 0, x_2 x_3 ... >_b$.
3. ...

Exemples: $0{,}625 = < 0, 101 >_2$

1. $0{,}625 \times 2 = 1, 25$
2. $0, 25 \times 2 = 0, 5$
3. $0, 5 \times 2 = 1$

Flottants

Question

Écrire un algorithme Python pour calculer la liste des chiffres en base 2 d'un flottant.

Flottants

Question

Écrire un algorithme Python pour calculer la liste des chiffres en base 2 d'un flottant.

```
x = 0.625
L = []
while x != 0:
    x = x * 2
    L.append(int(x))
    x = x - int(x)
```

Flottants

⚠ x peut avoir un développement décimal fini mais infini en base 2:

$$0{,}1 = <0,0001100110011001100110011001100...>_2$$

Flottants

⚠ x peut avoir un développement décimal fini mais infini en base 2:

$$0{,}1 = <0,0001100110011001100110011001100...>_2$$

Comme on ne peut stocker qu'un nombre fini de chiffres sur un PC, Python fait une troncature donc une approximation.

```
In [54]: 0.1 + 0.1 + 0.1
Out[54]: 0.30000000000000004

In [55]: 0.1 + 0.1 + 0.1 == 0.3
Out[55]: False
```

Il ne faut donc jamais tester une égalité entre deux flottants!

Flottants

Il ne faut donc jamais tester une égalité entre deux flottants!

Si x et y sont deux flottants, plutôt qu'écrire:

```
if x == y:
    ...
```

On testera si la différence est suffisamment petite:

```
if abs(x - y) < 0.0001:
    ...
```

Flottants

Pour convertir un flottant d'une base à une autre, on convertit sa partie entière et sa partie fractionnaire.

Question

Convertir à la main $< 1010,11 >_2$ en base 10

Flottants

Pour convertir un flottant d'une base à une autre, on convertit sa partie entière et sa partie fractionnaire.

Question

Convertir à la main $< 1010,11 >_2$ en base 10

Question

Convertir à la main 22,5625 en base 2

Flottants

Question

Comment stocker un flottant en mémoire?

Flottants

Question

Comment stocker un flottant en mémoire?

On utilise la notation scientifique:

$$932,134 = 9,32134 \times 10^2$$

$$< 1001,011 >_2 =$$

Flottants

Question

Comment stocker un flottant en mémoire?

On utilise la notation scientifique:

$$932,134 = 9,32134 \times 10^2$$

$$< 1001,011 >_2 = < 1,\underbrace{001011}_{\text{mantisse}} >_2 \times \underbrace{2^3}_{2^{\text{exposant}}}$$

Norme IEEE-754

Python code un flottant x en utilisant **64 bits**:

1. 1 bit pour le signe (0 si $x \geq 0$, 1 si $x < 0$)
2. 11 bits pour l'exposant: de $-2^{10} + 1 = -1023$ à $2^{10} = 1024$
3. 52 bits (chiffres significatifs) pour l'écriture en base 2 de la mantisse: de $0,\underbrace{00...00}_{52}$ à $0,\underbrace{11...11}_{52}$

Norme IEEE-754

Python code un flottant x en utilisant **64 bits**:

1. 1 bit pour le signe (0 si $x \geq 0$, 1 si $x < 0$)
2. 11 bits pour l'exposant: de $-2^{10} + 1 = -1023$ à $2^{10} = 1024$
3. 52 bits (chiffres significatifs) pour l'écriture en base 2 de la mantisse: de $0,\underbrace{00...00}_{52}$ à $0,\underbrace{11...11}_{52}$

Exemple:

1	**0**	...	**0**	**1**	**1**	**0**	...	**0**	**1**	**1**	**0**

représente:

Norme IEEE-754

Python code un flottant x en utilisant **64 bits**:

1. 1 bit pour le signe (0 si $x \geq 0$, 1 si $x < 0$)
2. 11 bits pour l'exposant: de $-2^{10} + 1 = -1023$ à $2^{10} = 1024$
3. 52 bits (chiffres significatifs) pour l'écriture en base 2 de la mantisse: de $0,\underbrace{00...00}_{52}$ à $0,\underbrace{11...11}_{52}$

Exemple:

1	**0**	...	**0**	**1**	**1**	**0**	...	**0**	**1**	**1**	**0**

représente: $-1,6 \times 2^3$

Flottants

⚠ Ainsi, il peut y avoir dépassement de flottant en Python:

```
In [70]: 2.**1023
Out[70]: 8.98846567431158e+307

In [71]: 2.**1024
---------------------------------------------------------------------------
OverflowError                                Traceback (most recent call last
<ipython-input-71-503cd93858fd> in <module>()
----> 1 2.**1024

OverflowError: (34, "Le résultat numérique est en dehors de l'intervalle")
```

Flottants

```
In [1]: 2**52 + 0.5 - 2**52
Out[1]: 0.0
```

Flottants

```
In [1]: 2**52 + 0.5 - 2**52
Out[1]: 0.0
```

⚠ Seulement 52 chiffres significatifs peuvent être stockés, on peut avoir des problèmes d'arrondis.

Flottants

```
In [1]: 2**52 + 0.5 - 2**52
Out[1]: 0.0
```

⚠ Seulement 52 chiffres significatifs peuvent être stockés, on peut avoir des problèmes d'arrondis.

Question

Que fait l'algorithme suivant?

```
somme = 2**53
while somme < 2**53 + 3:
    somme += 0.1
print(somme)
```

Question 3 :

Parmi les affirmations suivantes lesquelles sont vraies :

A) Un nombre entier naturel qui est représenté en binaire par une suite de 0 et de 1 et qui se termine par 1 est pair.

B) Le nombre décimal 0,1 possède une représentation binaire finie.

C) Sur un octet de mémoire le plus grand entier naturel représentable par une suite de 0 ou de 1 est 255.

D) 110 en binaire représente 10 en base dix.

Question 2 Parmi les affirmations suivantes, indiquez celle ou celles qui sont vraies.

A) Si un nombre réel admet une écriture décimale finie, alors il possède une écriture binaire finie.

B) Si un nombre réel admet une écriture binaire finie, alors il possède une écriture décimale finie.

C) Tous les nombres réels admettent une écriture binaire finie.

D) Tous les nombres entiers naturels admettent une écriture binaire finie.

Question 3 Parmi les affirmations suivantes, indiquez celle ou celles qui sont vraies.

A) L'utilisation de nombres flottants peut provoquer des erreurs d'arrondis, mais celles-ci ne sont jamais graves car les erreurs d'arrondis sont minimes.

B) L'utilisation de nombres flottants peut provoquer de graves erreurs d'arrondis.

C) L'utilisation de nombres flottants ne provoque pas d'erreur d'arrondis.

D) Pour ne pas avoir d'erreur d'arrondis, il suffit de coder les flottants sur 64 bits plutôt que sur 32 bits.