SÉANCE
GÉOMÉTRIE 1

Guillaume Pérution-Kihli

Université de Montpellier
26 mars 2022

PLAN

COMPARAISON DE FLOTTANTS

```python
def compareFloatNum(a, b):
    if (a == b):
        print("The numbers are equal")
    else:
        print("The numbers are not equal")
if __name__ == '__main__':
    a = (0.3 * 3) + 0.1
    b = 1
    compareFloatNum(a, b)
```

Quel est le message affiché par ce code source ?

COMPARAISON DE FLOTTANTS

```
def compareFloatNum(a, b):
    if (a == b):
        print("The numbers are equal")
    else:
        print("The numbers are not equal")
if __name__ == '__main__':
    a = (0.3 * 3) + 0.1
    b = 1
    compareFloatNum(a, b)
```

Quel est le message affiché par ce code source ?

Les nombres ne sont pas égaux !

COMPARAISON DE FLOTTANTS

Une solution : faire la valeur absolue de la différence et la comparer avec un très petit nombre.

```python
epsilon = 1e-9
def compareFloatNum(a, b):
    if (abs(a - b) < epsilon):
        print("The numbers are equal ");
    else:
        print("The numbers are not equal ");
if __name__ == '__main__':
   a = (0.3 * 3) + 0.1;
   b = 1;
   compareFloatNum(a, b);
```

Quel est le message affiché par ce code source ?

COMPARAISON DE FLOTTANTS

Une solution : faire la valeur absolue de la différence et la comparer avec un très petit nombre.

```
epsilon = 1e-9
def compareFloatNum(a, b):
    if (abs(a - b) < epsilon):
        print("The numbers are equal ");
    else :
        print("The numbers are not equal ");
if __name__ == '__main__':
    a = (0.3 * 3) + 0.1;
    b = 1;
    compareFloatNum(a, b);
```

Quel est le message affiché par ce code source ?

Les nombres sont égaux !

COMPARAISON DE FLOTTANTS

Une solution : faire la valeur absolue de la différence et la comparer avec un très petit nombre.

```
epsilon = 1e-9
def compareFloatNum(a, b):
    if (abs(a - b) < epsilon):
        print("The numbers are equal ");
    else:
        print("The numbers are not equal ");
if __name__ == '__main__':
    a = (0.3 * 3) + 0.1;
    b = 1;
    compareFloatNum(a, b);
```

Quel est le message affiché par ce code source ?

Les nombres sont égaux !

COMPARAISON DE FLOTTANTS

Autre solution, utiliser les modules standards de Python :

- `fractions` : Représentation exacte des nombres rationnels (à privilégier par rapport aux flottants car pas d'arrondi)
- `decimal` : Flottants décimaux à précision ajustable

Attention : pour éviter les erreurs de précision avec le module fractions, vous devez absolument éviter de créer des fractions à partir de flottants :

- $Fraction(0.3) * Fraction(3) + Fraction(0.1) = 36028797018963967/36028797018963968$
- $Fraction(3, 10) * Fraction(3) + Fraction(1, 10) = 1$

Attention : le module *decimal* ne permet pas de corriger tous les problèmes, certains nombres peuvent avoir un nombre infini de chiffres derrière la virgule en binaire ($0, 3$ par exemple !).

PLAN

ENVELOPPE CONVEXE

Dans un plan, l'enveloppe convexe peut être comparée à la région limitée par un élastique qui englobe tous les points qu'on relâche jusqu'à ce qu'il se contracte au maximum.

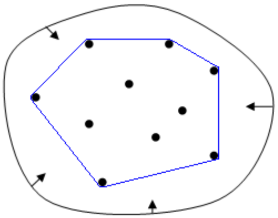

Fig. : Analogie de l'élastique entourant tous les points.

Il existe des algorithmes en $O(n \log n)$ dans le pire cas ou en moyenne. L'algorithme d'Andrew est l'un des plus courts à écrire.

PLAN

QUELQUES FORMULES

On a un polygone p simple avec n points classés dans un ordre anti-horaire. Pour les deux dernières formules, les points de p doivent avoir des coordonnées entières.

- Surface S de p : $\frac{1}{2}\sum_{i=0}^{n-1}(x_i y_{i+1} - x_{i+1} y_i)$
- Nombre de points n_b ayant des coordonnées entières sur le contour de p, pour chaque segment $[a, b]$ de p, on pose $x = abs(a.x - b.x)$ et $y = abs(a.y - b.y)$:
  - y si $x = 0$
  - x si $y = 0$
  - $pgcd(x, y)$ sinon

  Attention à ne pas compter les points a et b deux fois.
- Nombre de points n_i ayant des coordonnées entières à l'intérieur de p (théorème de Pick) : $n_i = A - \frac{n_b}{2} + 1$

PLAN

QUELQUES AUTRES PROBLÈMES

- Étant donné un ensemble de points, trouver l'écart minimal entre deux points : algorithme en temps linéaire en moyenne
- Étant donné un polygone rectilinéaire (i.e. dont tous les segments sont horizontaux ou verticaux), vérifier si celui-ci est simple : algorithme par balayage en $O(n \log n)$

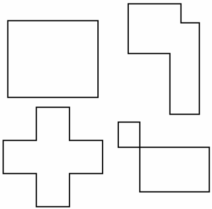

Fig. : Polygones rectilinéaires

Ces algorithmes seront ajoutés au document de référence