Python: tester si un quadrilatère est un parallélogramme, première approche des booléens

Auteur : Benoît TRUCHETET

Niveaux scolaires:
- 2nde

Mots clés :
- Algorithme / Programmation
- Géométrie

ENONCE

On considère quatre points distincts et non alignés du plan:
$A(x_a; y_a), B(x_B; y_B), C(x_C; y_C)$ *et* $D(x_D; y_D)$.

1. **Créer une fonction qui renvoie "oui" si le quadrilatère $ABCD$ est un parallélogramme et "non" sinon. Les arguments de la fonction seront les coordonnées des sommets du quadrilatère.**

On utilisera une instruction conditionnelle et la propriété suivante:
Un quadrilatère est un parallélogramme si et seulement si ses diagonales se coupent en leur milieu.
On pourra utiliser l'instruction `and`.

2. **Tester le programme avec les quadrilatères $ABCD$ suivants:**
   a) $A(-1; 2)$ $B(3, 2)$ $C(2; 0)$ *et* $D(-1; 0)$
   b) $A(5; 2)$ $B(3; 2)$ $C(2; 1)$ *et* $D(4; 1)$

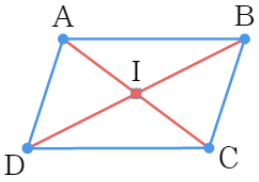
3. **Tester dans le Shell les lignes suivantes:**

```
1==1 and 2==1
1==2 and 2==2
1==1 and 2==2
```

4. **En déduire une façon de modifier la fonction `para` pour qu'elle ne fasse plus que deux lignes. La fonction renverra désormais les booléens TRUE ou FALSE.**

5. **Tester le nouveau script avec les quadrilatères de la question 3.**

1. ABCD est un parallélogramme si et seulement si les coordonnées de I point d'intersection des diagonales du quadrilatère vérifient :



$$x_I = \frac{x_A+x_C}{2} = \frac{x_B+x_D}{2} \; ; y_I = \frac{y_A+y_C}{2} = \frac{y_B+y_C}{2}$$

Une fois les coordonnées de I déterminées de deux manières différentes, l'une à partir de la diagonale [AC] et l'autre à partir de la diagonale [BD] nous allons mettre en place un test et vérifier si les deux égalités sont bien vérifiées.
Si les deux tests sont concluants la réponse "oui" est renvoyée et sinon la réponse "non" est renvoyée.

- Pour tester l'égalité, on utilise ==, il faudra être vigilant dans le cas où les coordonnées sont des flottants.
- L'instruction **and** nous permet de tester les deux égalités en une seule fois.

```
def para(XA,YA,XB,YB,XC,YC,XD,YD):
  if (XA+XC)/2==(XB+XD)/2 and (YA+YC)/2==(YB+YD)/2:
    return("oui")
  else:
    return("non")
```

2.

a) $A(-1;2)\ B(3,2)\ C(2;0)$ *et* $D(-1;0)$	**b)** $A(5;2)\ B(3;2)\ C(2;1)$ *et* $D(4;1)$
Exécutons script para.py RUN	
On appelle la fonction **para**: **para(-1,2,3,2,2,0,-1,0)**	On appelle la fonction **para**: **para(5,2,3,2,2,1,4,1)**
>>>para(-1,2,3,2,2,0, 'non'	>>>para(5,2,3,2,2,1,4 'oui'
ABCD n'est pas un parallélogramme	ABCD est un parallélogramme

3. Dans le Shell, on obtient les résultats suivants:

```
>>>1==1 and 2==1
False
>>>1==2 and 2==2
False
>>>1==1 and 2==2
True
>>>
```
RUN A⇔a CHAR

On remarque que les lignes ici renvoient directement TRUE ou FALSE qui signifient VRAI ou FAUX.
Il n'est donc pas utile de renvoyer les chaînes de caractères du type "oui" ou "non".

4. La définition de la fonction para devient:

```
def para(XA,YA,XB,YB,XC,YC,XD,YD):
  return (XA+XC)/2==(XB+XD)/2 and (YA+YC)/2==(YB+YD)/2
```

a) $A(-1;2)$ $B(3,2)$ $C(2;0)$ *et* $D(-1;0)$	b) $A(5;2)$ $B(3;2)$ $C(2;1)$ *et* $D(4;1)$
Exécutons le script para.py RUN	

On appelle la fonction **para**:
para(-1,2,3,2,2,0,-1,0)
`>>>para(-1,2,3,2,2,0,`
`False`
	On appelle la fonction **para**:
	para(5,2,3,2,2,1,4,1)
	`>>>para(5,2,3,2,2,1,4`
	`True`

ABCD n'est pas un parallélogramme	ABCD est un parallélogramme

On aurait pu utiliser des entrées/sorties (input / print) mais on voit tout de suite que cela alourdit considérablement le script par rapport à l'utilisation d'une fonction !

```
print("ABCD est il \nun parallelogramme ?")
print("Saisir les coordonnees \nde A,B,C et D")
xA=float(input("xA ?"))
yA=float(input("yA ?"))
xB=float(input("xB ?"))
yB=float(input("yB ?"))
xC=float(input("xC ?"))
yC=float(input("yC ?"))
xD=float(input("xD ?"))
yD=float(input("yD ?"))
print("Le TEST est : ")
print((xA+xC)/2==(xB+xD)/2 and (yA+yC)/2==(yB+yD)/2)
```