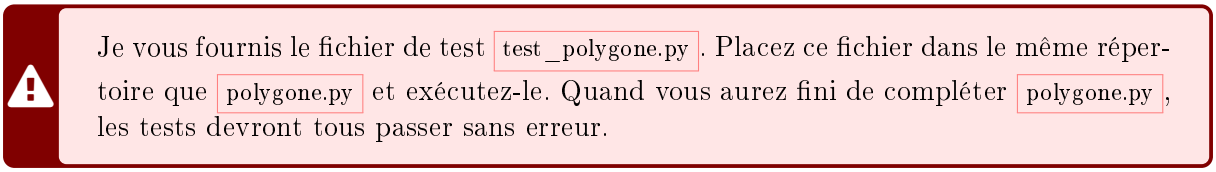
Dans cet exercice, on traite d'un problème de géométrie : on souhaite calculer l'aire d'un polygone quelconque.

À faire

Vous devrez implémenter les fonctions nécessaires en Python. Les fonctions à compléter se trouvent dans le fichier `polygone.py`.
La signature des fonctions et leur documentation – commentaire sous la signature de chaque fonction – vous donnent des informations utiles. Lisez-les !



Je vous fournis le fichier de test `test_polygone.py`. Placez ce fichier dans le même répertoire que `polygone.py` et exécutez-le. Quand vous aurez fini de compléter `polygone.py`, les tests devront tous passer sans erreur.

La suite du document est là pour vous donner des explications supplémentaires.

1 Coordonnées

Dans tous l'exercice, on se place dans un repère orthonormé $(O; I, J)$. Chaque point aura donc une paire de coordonnées $(x; y)$ que l'on représentera par un tuple. Par exemple `(4,7)` représente les coordonnées $(4; 7)$.

En géométrie, un point peut avoir un nom. On parle ainsi du point A, du point B... Mais cela n'a rien d'obligatoire. On ne s'embarrasse donc pas de donner un nom aux points, on ne donne que leurs coordonnées.

Les coordonnées peuvent aussi être des coordonnées de vecteur. Par exemple le vecteur reliant le point de coordonnées `(4,17)` au point de coordonnées `(12,−5)` a les coordonnées `(8,−22)`.

2 Polygone

2.1 Définition

Un polygone peut être défini par ses points, donnés dans un ordre précis.

Dans notre programme, le polygone sera représenté par un tableau de coordonnées.
Exemples :

- `[(1,1), (5,1), (1,4)]` définit un triangle. C'est d'ailleurs un triangle rectangle.
- `[(2,0), (10,2), (9,6), (1,4)]` définit un quadrilatère. C'est d'ailleurs un rectangle.
- `[(3,0), (9,−2), (4,−6), (−1,−3), (0,0)]` définit un pentagone (5 côtés).

2.2 Nettoyage

Est-ce que `[(3,0), (9,−2), (4,−6), (4−6), (9,−2), (−1,−3), (0,0), (3,0)]` est un polygone acceptable ? C'est un tableau de coordonnées, est-ce que cela suffit ?

Non ! On souhaite que les sommets consécutifs soient distincts.

- Ici on constate que `(4,−6)` est répété consécutivement.
- De même `(3,0)` est le premier et le dernier sommet.
- En revanche, `(9,−2)` est répété mais pas consécutivement.

On prévoit donc une fonction `clean(polygone)` qui crée une copie de polygone sans les doublons. Dans l'exemple, la fonction renverrait donc `[(3,0), (9,−2), (4−6), (9,−2), (−1,−3), (0,0)]`

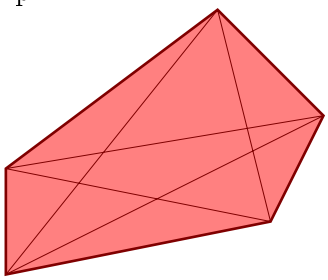
À faire

Implémenter la fonction ne fonction `clean(polygone)`.

Remarque : Lors d'une répétition consécutive, c'est la deuxième occurrence que l'on supprime. Si le dernier point est identique au premier, c'est le dernier que l'on supprime.

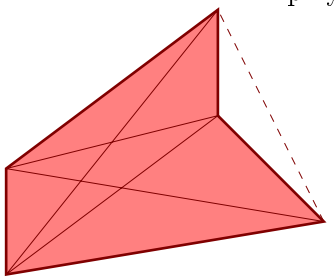
2.3 Convexité

Ci-dessous, les figures vous montre ce que l'on appelle **convexité** pour un polygone. Celui de droite n'est pas convexe car une des diagonales passe en dehors de l'intérieur du polygone.



`[(0,0) ,(5,1) ,(6,3) ,(4,5) ,(0,2) ]`

fig 1. Polygone convexe



`[(0,0) ,(6,1) ,(4,3) ,(4,5) ,(0,2) ]`

fig 2. Polygone non convexe

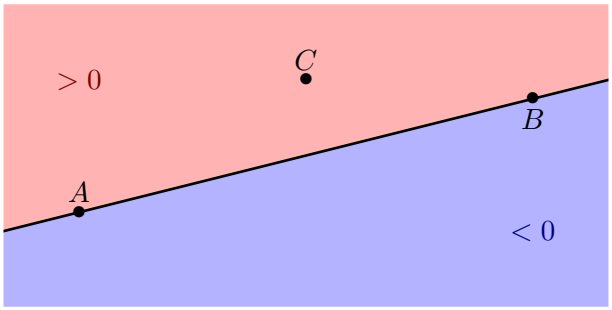
Remarque : un triangle est toujours convexe. La question ne se pose qu'à partir de 4 sommets.

Notre objectif est maintenant de réaliser la fonction `is_convexe(polygone)` qui détermine si un polygone est convexe ou non. Pour cela, nous allons développer des fonctions annexes.

2.3.1 De quel côté ?

On cherche à savoir de quel côté est un point par rapport à une droite.

On se donne deux points, par exemple A et B dont on connaît les coordonnées. On suppose que $A \neq B$ pour pouvoir déterminer la droite (AB). On se demande si un point C est d'un côté ou de l'autre de la droite, ou même sur la droite.



La méthode sera la suivante :

i. Déterminer les coordonnées du vecteur $\overrightarrow{AB}$.

Calcul : on sait que $\overrightarrow{AB}\begin{pmatrix} x_B - x_A \\ y_B - y_A \end{pmatrix}$ ou encore $x_{\overrightarrow{AB}} = x_B - x_A$ et $y_{\overrightarrow{AB}} = y_B - y_A$.

ii. Calculer les coordonnées de $\overrightarrow{AC}$

iii. Faire un produit particulier entre $\overrightarrow{AB}$ et $\overrightarrow{AC}$, on l'appelle le produit vectoriel.

Calcul : $\delta = \overrightarrow{AB} \times \overrightarrow{AC} = x_{\overrightarrow{AB}} \cdot y_{\overrightarrow{AC}} - y_{\overrightarrow{AB}} \cdot x_{\overrightarrow{AC}}$

iv. Décision

- Si $\delta = 0$, alors $C \in (AB)$.

- Si $\delta > 0$, alors C sur la gauche quand on regarde B depuis A.
- Si $\delta < 0$, alors C sur la droite.

À faire

1) Compléter la fonction `vecteur(A,B)` où `A` est un tuple contenant les coordonnées de A, idem pour B.
   La fonction renvoie un tuple contenant les coordonnées de $\overrightarrow{AB}$.
2) Compléter la fonction `product(vAB,vAC)` où `vAB` et `vAC` sont des tuples contenant les coordonnées des vecteurs $\overrightarrow{AB}$ =et $\overrightarrow{AC}$ et qui renvoie le coefficient δ.
3) Compléter la fonction `get_side_of_AB(A,B,C)` où `A` est un tuple contenant les coordonnées de A, idem pour B et C.

   La fonction doit renvoyer :
   - 0 si C est sur (AB)
   - 1 si C est à gauche quand on est en A et qu'on regarde B
   - -1 si C est à droite quand on est en A et qu'on regarde B

   Précondition : A et B distincts.

2.3.2 convexe ou pas ?

Munis de la fonction `get_side_of_AB(A,B,C)`, on va pouvoir déterminer si un polygone est convexe et compléter la fonction `is_convexe(polygone)`. En effet, un polygone convexe tourne toujours dans le même sens, toujours à gauche ou toujours à droite. Nous allons parcourir les sommets et déterminer, avec `get_side_of_AB(A,B,C)`, dans quel sens tourne le polygone. S'il tourne parfois à gauche et parfois à droite, alors il n'est pas convexe.

FONCTION `is_convexe`
Entrées : polygone
Sorties : Vrai si convexe, Faux sinon

```
1  début
2      soit T un tableau vide
3      pour chaque point du polygone faire
4          soit A ce point
5          soient B et C les points suivants
6          soit c le côté de C par rapport à (AB)
7          stocker c dans T
8      fin
9      si T contient 1 et -1 alors
10         renvoyer Faux
11     sinon
12         renvoyer Vrai
13     fin
14 fin
```

À faire

Compléter la fonction `is_convexe(polygone)` en implémentant cet algorithme.

3 Calcul d'aire

3.1 Aire d'un triangle

Soit $A(x_A\,;\,y_A)$, $B(x_B\,;\,y_B)$ et $C(x_C\,;\,y_C)$, pour calculer l'aire de ABC :

$$\mathcal{A} = \frac{1}{2} \cdot \left| \overrightarrow{AB} \times \overrightarrow{AC} \right|$$

où $\times$ est le produit de vecteurs déjà rencontré.

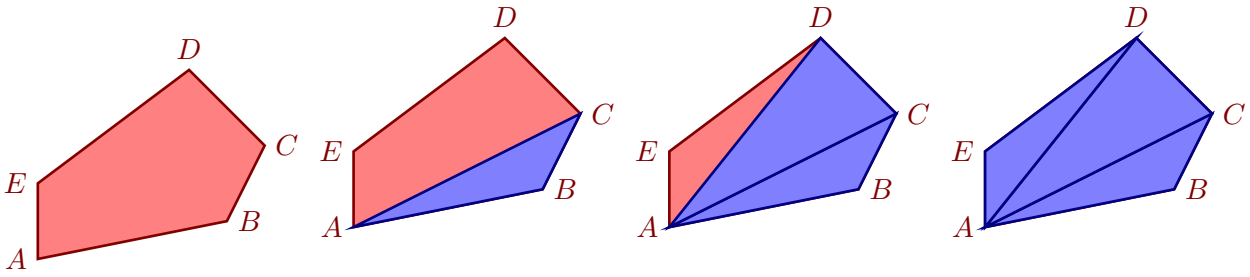
À faire

Compléter la fonction `aire_triangle(A, B, C)` ou `A` est un tuple contenant les coordonnées de A, idem pour B et C. La fonction renvoie l'aire de ABC.

Remarque : Cette fonction fonctionne sans problème si deux des points sont confondus.

3.2 Aire d'un polygone

On se limite au cas d'un polygone convexe – *C'est donc une précondition de notre fonction.*
Exemple : $ABCDE =$ `[(0,0) ,(5,1) ,(6,3) ,(4,5) ,(0,2) ]`



FONCTION `aire_polygone`
Entrées : polygone
Sorties : Aire du polygone

```
début
    soit t initialement à 0
    soit A le premier point du polygone
    pour chaque point du polygone, à part premier et dernier faire
        soit B ce point
        soient C le point suivant
        soit s l'aire de ABC
        ajouter s à t
    fin
    renvoyer t
fin
```

À faire

Compléter la fonction `aire_polygone(polygone)` en implémentant cet algorithme.
Précondition : Le polygone est convexe.