Paquet Cadeau

Consignes

Les consignes suivantes **doivent** être respectées sous peine d'être pénalisé par des **points négatifs**.

Le code doit être clair.

- Évitez les lignes à rallonge. S'il faut introduire une variable intermédiaire, faîte le.
- Nommez vos variables avec du sens. Évitez toutefois des noms trop longs. Auquel cas, décrivez le rôle de la variable en commentaire.
- Ne surchargez pas de commentaires. Les commentaires utiles sont :
  - Description du rôle d'une variable
  - Description du rôle d'une boucle complexe.

Soignez la présentation de votre code.

- Marquez clairement les indentations (2 grands carreaux).
- Éviter les ratures trop nombreuses dans un code. Une ou deux, maximum.
- Écrivez votre instruction sur plusieurs lignes si elle est longue en ajoutant simplement une indentation après le retour à la ligne.

Respectez la syntaxe Python. - *0 à l'exercice si syntaxe inventée.*

- Tout ce qui n'est pas sur la fiche de syntaxe ne peut pas être utilisé.
- Les petites erreurs n'implique pas 0 (oublie de :, = au lieu de ==, …)
- Si vous avez une idée, mais que vous n'arrivez pas à l'écrire, n'inventez pas. Écrivez votre idée en commentaire.

Utilisation de `print` ou de `input` interdite - *0 à l'exercice sinon*

Pour nommer une fonction, utilisez le nom donné dans l'énoncé - *0 à l'exercice sinon*

I Algorithme Classiques

Écrivez les algorithmes ayant la spécification suivante :

Question 1. *Exponentiation rapide*

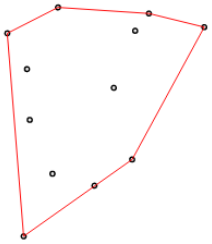
Entrées : a un entier et b un entier positifs.
Sortie : a^b par la méthode d'exponentiation rapide.

Question 2. *Comptage*

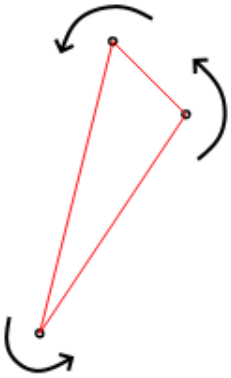
Entrées : Une liste `l` et un objet `x`
Sortie : Le nombre de fois que x apparaît dans l

II Enveloppe Convexe

Étant donné un ensemble de point du plan, l'enveloppe convexe de cet ensemble est un polyèdre dont les sommets sont des points de l'ensemble et qui contient tous les points de l'ensemble.



enveloppe convexe d'un ensemble de points



sens trigonométrique de trois points

Le but du problème est de résoudre le problème suivant : étant donné un ensemble de points du plan, qu'elle est l'enveloppe convexe de cet ensemble de point. Il existe plusieurs algorithmes permettant de résoudre ce problème. On implémentera une version de l'algorithme dit *du papier-cadeau* ou *algorithme de Jarvis*. Puis, on proposera une implémentation de l'algorithme d'Andrew, autre algorithme permettant de résoudre le problème.

Les points du plan seront représentés par des couples de flottant `(x,y)` tel que `x` est l'abscisse du point et `y` dont ordonnée.

Étant donnés deux points A et B on rappelle la définition de la distance euclidienne entre A et B, notée AB :

$$\mathrm{AB} = ((x_\mathrm{A} - x_\mathrm{B})^2 + (y_\mathrm{A} - y_\mathrm{B})^2)^{\frac{1}{2}}$$

Question 3. *Distance*

Implémenter une fonction `distance(A,B)` qui prend en entrée deux points sous la forme de deux couples de flottant et qui renvoie la distance euclidienne entre ces deux points.

Question 4. *Orientation*

On considère trois points du plan A, B et C. Intuitivement, on dit que le triplet (A, B, C) est dans le sens trigonométrique si C est à gauche de la droite (AB). Formellement pour $\mathrm{A}(x_\mathrm{A}, y_\mathrm{A})$, $\mathrm{B}(x_\mathrm{B}, y_\mathrm{B})$ et $\mathrm{C}(x_\mathrm{C}, y_\mathrm{C})$, on pose $z = (x_\mathrm{A} - x_\mathrm{B})\,(y_\mathrm{C} - y\mathrm{B}) - (y_\mathrm{A} - y_\mathrm{B})\,(x_c - x_b)$ et on donne la définition suivante :

$$(\mathrm{A},\mathrm{B},\mathrm{C}) \text{ est dans le sens trigonométrique} \iff \begin{cases} \mathrm{A} = \mathrm{B} & \text{ou,} \\ z < 0 & \text{ou,} \\ z = 0 \text{ et } \mathrm{A} \neq \mathrm{C} \text{ et } \mathrm{AC} < \mathrm{AB} \end{cases}$$

Implémenter une fonction `trigo(A,B,C)` qui prend en entrée trois points A, B et C, et qui renvoi un booléen vrai si et seulement si (A, B, C) sont dans le sens trigonométrique.

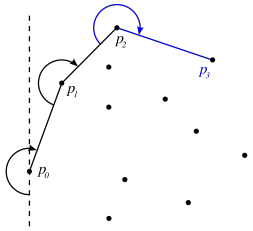
Question 5. *Point minimum*

On rappelle que, pour deux couples de flottants $A = (x_A, y_A)$ et $B = (x_B, y_B)$ la comparaison `A < B` en python renvoie `True` si et seulement si ($x_A < x_B$ ou si $x_A = x_B$ et $y_A < y_B$)

Implémentez une fonction `point_min(l)` qui prend en entrée un ensemble de points disjoints du plan, sous la forme d'une liste de couples de flottants, et qui renvoie l'unique point de l'ensemble qui est strictement plus petit que tous les autres au sens de <. Si l'ensemble est vide la fonction renverra `None`.

III Jarvis

L'idée de l'algorithme du papier-cadeau et de partir du plus petit point de l'ensemble et de chercher le point *le plus à gauche* pour l'ajouter dans l'enveloppe, comme si on enroulait du papier-cadeau autour des points.



Jarvis

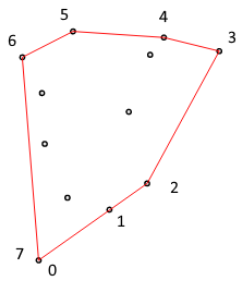
Question 6. *Point le plus à gauche*

On fixe un point P et un ensemble de points $\mathcal{E}$. Le point le plus à gauche de P dans E est défini comme le point $Q \in E$ tel que $\forall Q' \in E$, (P, Q', Q) est dans le sens trigonométrique.

Implémentez une fonction `plus_a_gauche(P, E)` qui prend en entrée un point P, représenté par un couple de flottant, et un ensemble de points E non vide, représenté par une liste de couples de flottant, et qui renvoie le point le plus à gauche de P dans E.

Question 7. *Papier Cadeau*

L'enveloppe convexe d'un ensemble de points sera représentée par une liste de points $[e_0, e_1, ..., e_n]$ tels que $e_0 = e_n$ et pour tout $i < n$, tous les points p de l'ensemble, (e_i, e_{i+1}, p) *n'est pas* dans le sens trigo.



liste des 8 points de l'enveloppe convexe dans l'ordre.

L'algorithme du papier-cadeau prend en entrée un ensemble $\mathcal{E}$ de deux points ou plus, sous la forme d'une liste de couples de flottants, renvoie l'enveloppe convexe de l'ensemble sous la forme d'une liste comme décrit ci-dessus, et fonctionne comme ceci :

1. Initialise une liste `env` vide.
2. Calcule e_0 le plus petit point de l'ensemble et l'ajoute à la fin de `env`.
3. Calcule e comme le point le plus à gauche du dernier élément de `env` dans $\mathcal{E}$ et l'ajoute à la fin de `env`.
4. Tant que le dernier élément de `env` n'est pas égal au premier élément de `env`, répéter l'étape 3.

Écrire une fonction `jarvis` qui implémente l'algorithme du papier-cadeau.

IV Algorithme d'Andrew

L'algorithme d'Andrew tri d'abord l'ensemble des points selon < pour diminuer le nombre de comparaisons à réaliser.

Question 8. *Tri des points*

Pour trier les points, on décide d'utiliser l'algorithme de tri nommé *tri par sélection*. Supposons qu'on cherche à trier l. Le tri par sélection réalise les étapes de calcul suivantes :

1. Calculer le minimum, m_0, de l et échanger sa place dans la liste avec le premier élément.
2. Calculer le minimum, m_1, de l privée de m_0 et échanger sa place avec le deuxième élément.
3. Calculer le minimum, m_2, de l privée de m_0 et m_1 et échanger sa place avec le troisième élément.
4. et ainsi de suite jusqu'à ce que l soit entièrement triée.

Implémenter une fonction `tri_points(E)` qui prend en entrée un ensemble de point, sous la forme d'une liste de couples de flottant, et trie la liste par la méthode du tri par sélection. La fonction ne renvoie rien.

Question 9. *Algorithme d'Andrew*

L'algorithme d'Andrew est décrit par les étapes suivantes sur une liste `l` de points :

1. Définir une copie `t` triée de `l`.
2. Initialiser deux listes vides `haut` et `bas`.
3. Pour tout point P de t
   (a) On note H et G le dernier et l'avant-dernier points de `haut` s'ils existent.
   (b) Tant que H et G existent et que (P, H, G) n'est pas dans le sens trigonométrique, supprimer H de `haut`
   (c) Ajouter P à la fin de `haut`
   (d) On note B et C le dernier et l'avant-dernier points de `bas` s'ils existent.
   (e) Tant que B et C existent et que (C, B, P) n'est pas dans le sens trigonométrique, supprimer B de `bas`
   (f) Ajouter P à la fin de `bas`
4. On concatène `haut` privé de son dernier élément et `bas` renversé, privée de son dernier élément.

Écrire une fonction `andrew` qui implémente l'algorithme d'Andrew.