```
# Ajouter un élément au début
ma_deque.appendleft(0)

# Supprimer le dernier élément
dernier_element = ma_deque.pop()

# Supprimer le premier élément
premier_element = ma_deque.popleft()

# Accéder aux éléments
print(ma_deque) # Affiche deque([1, 2, 3])

# Opérations spécifiques aux deques (rotate, etc.)
ma_deque.rotate(1) # Fait tourner les éléments vers la droite

print(ma_deque)
# etc.
```

Affichage :

```
deque([1, 2, 3])
deque([3, 1, 2])
```

Exercice N° 5 - Extrait de X 2015 : Enveloppes convexes dans le plan

Ce sujet a pour objectif de calculer des enveloppes convexes de nuages de points dans le plan affine, un grand classique en géométrie algorithmique. On rappelle qu'un ensemble $C \subseteq \mathbb{R}^2$ est convexe si et seulement si pour toute paire de points $p, q \in C$, le segment de droite $[p, q]$ est inclus dans C. L'enveloppe convexe d'un ensemble $P \subseteq \mathbb{R}^2$, notée $\text{Conv}(P)$, est le plus petit convexe contenant P. Dans le cas où P est un ensemble fini (appelé *nuage de points*), le bord de $\text{Conv}(P)$ est un polygône convexe dont les sommets appartiennent à P, comme illustré dans la figure 1.

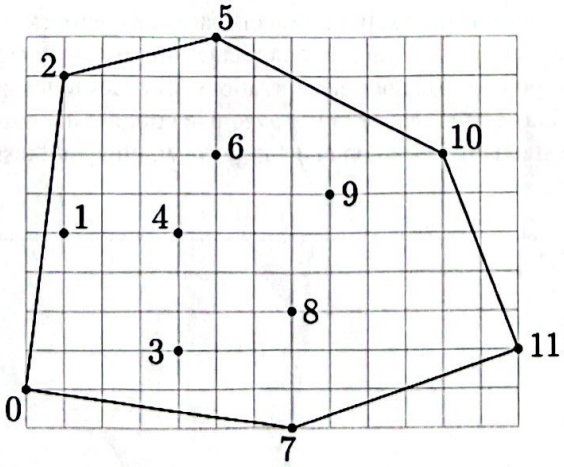

Figure 1 – Un nuage de points, numérotés de 0 à 11, et le bord de son enveloppe convexe.
Le calcul de l'enveloppe convexe d'un nuage de points est un problème fondamental en informatique, qui trouve des applications dans de nombreux domaines comme :
– la robotique, par exemple pour l'accélération de la détection de collisions dans le cadre de la planification de trajectoire,

Dans toute la suite on supposera que le nuage de points P est de taille $n \geq 3$ et en position générale, c'est-à-dire qu'il ne contient pas 3 points distincts alignés.
Ces hypothèses vont permettre de simplifier les calculs en ignorant les cas pathologiques, comme par exemple la présence de 3 points alignés sur le bord de l'enveloppe convexe. Nos programmes prendront en entrée un nuage de points P dont les coordonnées sont stockées dans un tableau tab à 2 dimensions, comme dans l'exemple ci-dessous qui contient les coordonnées du nuage de points de la figure 1 :

i \ j	0	1	2	3	4	5	6	7	8	9	10	11
0	0	1	1	4	4	5	5	7	7	8	11	13
1	0	4	8	1	4	9	6	-1	2	5	6	1

Précisons que les coordonnées, supposées entières, sont données dans une base orthonormée du plan, orientée dans le sens direct. La première ligne du tableau contient les abscisses, tandis que la deuxième contient les ordonnées. Ainsi, la colonne d'indice j contient les deux coordonnées du point d'indice j. Ce dernier sera nommé pj dans la suite.

Question 1 Ecrire une fonction plusBas(tab,n) qui prend en paramètre le tableau tab de taille $2 \times n$ et qui renvoie l'indice j du point le plus bas (c'est-à-dire de plus petite ordonnée) parmi les points du nuage P. En cas d'égalité, votre fonction devra renvoyer l'indice du point de plus petite abscisse parmi les points les plus bas.
Sur le tableau exemple précédent, le résultat de la fonction doit être l'indice 7.
Dans la suite nous aurons besoin d'effectuer un seul type de test géométrique : celui de l'orientation.

Définition 1 *Étant donnés trois points p_i, p_j, p_k du nuage P, distincts ou non, le test d'orientation renvoie +1 si la séquence (p_i, p_j, p_k) est orientée positivement, −1 si elle est orientée négativement, et 0 si les trois points sont alignés (c'est-à-dire si deux au moins sont égaux, d'après l'hypothèse de position générale).*

Pour déterminer l'orientation de (p_i, p_j, p_k), il suffit de calculer l'aire signée du triangle, comme illustré sur la figure 2. Cette aire est la moitié du déterminant de la matrice 2×2 formée par les coordonnées des vecteurs $\overrightarrow{p_i p_j}$ et $\overrightarrow{p_i p_k}$.

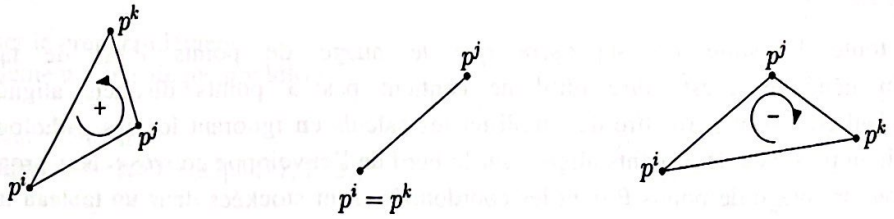

Figure 2 – Test d'orientation sur la séquence (pi,pj,pk) : positif à gauche, nul au centre, négatif à droite.

Question 2 Sur le tableau exemple précédent, donner le résultat du test d'orientation pour les choix d'indices suivants :

– i = 0, j = 3, k = 4,
– i = 8, j = 9, k = 10.

Question 3 Ecrire une fonction orient(tab,i,j,k) qui prend en paramètres le tableau tab et trois indices de colonnes, potentiellement égaux, et qui renvoie le résultat (−1, 0 ou +1) du test d'orientation sur la séquence (pi,pj,pk) de points de P

Partie II. Algorithme du paquet cadeau

Cet algorithme a été proposé par R. Jarvis en 1973. Il consiste à envelopper peu à peu le nuage de points P dans une sorte de paquet cadeau, qui à la fin du processus est exactement le bord de Conv(P). On commence par insérer le point de plus petite ordonnée (celui d'indice 7 dans l'exemple de la figure 1) dans le paquet cadeau, puis à chaque étape de la procédure on sélectionne le prochain point du nuage P à insérer.

La procédure de sélection fonctionne comme suit. Soit pi le dernier point inséré dans le paquet cadeau à cet instant. Par exemple, i = 10 dans l'exemple de la figure 3. Considérons la

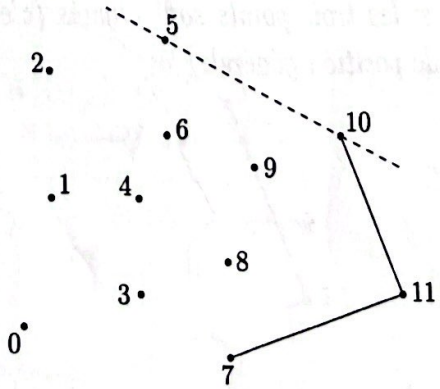

Figure 3 – Mise à jour du paquet cadeau après insertion du point p10.

relation binaire $\preceq$ définie sur l'ensemble $P \setminus \{p_i\}$ par :

$$p_j \preceq p_k \iff \text{orient}(tab, i, j, k) \leq 0.$$

Question 4 Justifier brièvement le fait que $\preceq$ est une relation d'ordre total sur l'ensemble $P \setminus \{p_i\}$, c'est-à-dire :

- (réflexivité) pour tout $j \neq i$, $p_j \preceq p_j$,
- (antisymétrie) pour tous $j, k \neq i$, $p_j \preceq p_k$ et $p_k \preceq p_j$ implique $j = k$,
- (transitivité) pour tous $j, k, l \neq i$, $p_j \preceq p_k$ et $p_k \preceq p_l$ implique $p_j \preceq p_l$,
- (totalité) pour tous $j, k \neq i$, $p_j \preceq p_k$ ou $p_k \preceq p_j$.

Ainsi, le prochain point à insérer (le point d'indice 5 dans la figure 3) est l'élément maximum pour la relation d'ordre $\preceq$. Il peut se calculer en temps linéaire (c'est-à-dire majoré par une constante fois n) par une simple itération sur les points de $P \setminus \{p_i\}$.

Question 5 Décrire une réalisation en Python de la procédure. Elle prendra la forme d'une fonction prochainPoint(tab,n,i), qui prend en paramètre le tableau tab de taille 2 × n ainsi que l'indice i du point inséré en dernier dans le paquet cadeau, et qui renvoie l'indice du prochain point à insérer. Le temps d'exécution de votre fonction doit être majoré par une constante fois n, pour tous n et i. La constante doit être indépendante de n et i, et on ne demande pas de la préciser.

Question 6 Décrire à la main le déroulement de la procédure prochainPoint sur l'exemple de la figure 3. Plus précisément, indiquer la séquence des points de P \ {p10} considérés et la valeur de l'indice du maximum à chaque itération.

On peut maintenant combiner la fonction prochainPoint avec la fonction plusBas de la question 1 pour calculer le bord de l'enveloppe convexe de P. On commence par insérer le point pi d'ordonnée la plus basse, puis on itère le processus de mise à jour du paquet cadeau jusqu'à ce que le prochain point à insérer soit de nouveau pi. A ce moment-là on renvoie le paquet cadeau ` comme résultat sans insérer pi une seconde fois.

Un détail technique : comme la taille du paquet cadeau augmente peu à peu lors du processus, et qu'à la fin elle peut être petite par rapport au nombre n de points de P, nous stockerons les indices des points du paquet cadeau dans une liste. Par exemple, sur le nuage de la figure 1, le résultat sera la liste [7, 11, 10, 5, 2, 0].

Question 7 Ecrire une fonction convJarvis(tab,n) qui prend en paramètre le tableau tab de taille 2 × n représentant le nuage P, et qui renvoie une liste contenant les indices des sommets du bord de l'enveloppe convexe de P, sans doublon. Le temps d'exécution de votre fonction doit être majoré par une constante fois nm, où m est le nombre de points de P situés sur le bord de Conv(P).