- Examen d'algorithmique géométrique -

- Corrigé rapide -

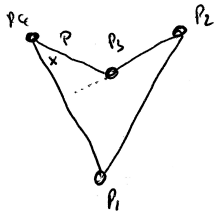
Barême :
Exercice 1 sur 8 points : 1) 0.5pts - 2) 1pt - 3) 1pt - 4) 1pt - 5) 0.5pts - 6) 1.5pts - 7) 2.5pts -
Exercice 2 sur 4 points : 1) 0.5pts - 2) 1pt - 3) 0.5pts - 4) 0.5pts - 5) 1.5pts -
Exercice 3 sur 8 points : 1) 1pt - 2) 1pt - 3) 1pt - 4) 1.5pts - 5) 2pts - 6) 1.5pts -

- Exercice 1 - Test d'appartenance à un polygone convexe -

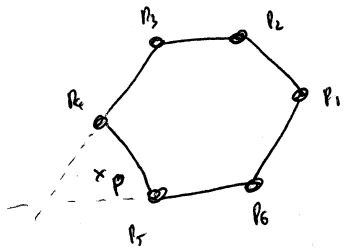
1. Voir cours, ou CC...
2. L'algo suivant marche :

```
pour tous les i = 1 à n faire
    si p est à droite de [p_i p_{i+1}] orienté de p_i à p_{i+1} alors
        retourner FAUX;
retourner VRAI;
```

3. Non, ce test ne fonctionne plus si $\mathcal{P}$ n'est pas convexe. Par exemple, dans l'exemple ci-dessous, p est dans $\mathcal{P}$ mais est à droite du segment $[p_2p_3]$ orienté de p_2 vers p_3.



4. On peut lancer un algorithme de calcul d'enveloppe convexe sur l'ensemble de points $\{p_1, \ldots, p_n\}$. À partir du point p_1, on déroule l'enveloppe convexe à partir de p_1. $\mathcal{P}$ est convexe si et seulement si on tombe sur la liste $(p_1, p_2, \ldots, p_n)$ (en supposant que tous les p_i soient disjoints). Si on utilise l'algo de Graham, on obtient une complexité en $O(n \log n)$.
5. Voir exemple ci-dessous.



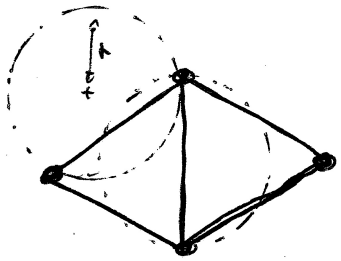
6. Soit p est à droite du segment $[p_i p_k]$ orienté de p_i vers p_k, soit il est à droite de $[p_k p_j]$ orienté de p_k vers p_j, soit il est à gauche de ces deux segments et il est dans le triangle $p_i p_j p_k$ et donc dans le polygone $\mathcal{P}$.
7. On peut alors construire un algo par dichotomie :

```
si p est à gauche de [p_1 p_n] orienté de p_1 à p_n alors
    retourner NON;
i ← 1;
j ← n;
tant que j − i > 1 faire
    k ← ⌊(i+j)/2⌋;
    si p est à droite de [p_i p_k] orienté de p_i à p_k alors j ← k;
    sinon
        si p est à droite de [p_k p_j] orienté de p_k à p_j alors i ← k;
        sinon retourner OUI;   // p est dans le triangle p_i p_j p_k, donc dans P
retourner NON;
```

À chaque tour de boucle, $j - i$ diminue de moitié. On fait donc au plus $O(\log n)$ tours de boucle. Chaque autre opération élémentaire s'effectue en $O(1)$, donc globalement l'algo fonctionne en $O(\log n)$.

- Exercice 2 - α-shape -

1. Voir cours...
2. Voir exemple ci-dessous.

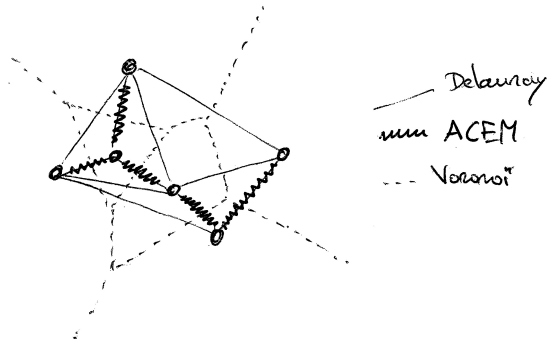


3. Si $\alpha < \frac{1}{2}\min\{pq \ : \ p \in \mathcal{P},\ q \in \mathcal{P},\ p \neq q\}$ alors un disque de rayon α est trop petit et ne peut avoir deux points de $\mathcal{P}$ comme diamètre. Donc, l'α-shape de $\mathcal{P}$ est vide.
4. Si pq est une arête de l'α-shape, alors par définition il existe un disque de rayon α contenant p et q sur son bord et aucun point de $\mathcal{P}$ dans son intérieur. Par le Lemme 1, pq est une arête de la triangulation de Delaunay.
5. Dans ce cas l'α-shape est l'enveloppe convexe de $\mathcal{P}$. En effet, supposons que ce ne soit pas le cas, l'α-shape contiendrait une arête pq bordée par deux triangles T_1 et T_2 de la triangulation de Delaunay de $\mathcal{P}$. Comme pq est une arête de l'α-shape de $\mathcal{P}$, il existe un disque D de rayon α contenant p et q sur son bord et aucun point de $\mathcal{P}$ dans son intérieur. Comme T_1 et T_2 sont de part et d'autre de pq, un de ces triangles, disons T_1 est du même côté que D par rapport à pq. Par hypothèse, le rayon de D (α) est strictement plus grand que le rayon du cercle circonscrit à T_1. Donc D doit contenir

le troisième point (différent de p et q) du triangle T_1, ce qui contredit la définition de D.

- Exercice 3 - Arbre euclidien minimum -

1. L'exemple suivant répond à la question :



2. On constuit un graphe complet dont les sommets sont les points de $\mathcal{P}$. Chaque arête pq est valuée par la distance de p à q. On lance l'algorithme de Kruskal sur le graphe construit. L'arbre retourné est un ACEM de $\mathcal{P}$. Le graphe constuit possède $n(n-1)/2$ arêtes. L'algo proposé tourne donc en temps $O(n^2 \log n)$.
3. On a $AC \leq AO + OC$ (inégalité triangulaire, 'la ligne droite est le plus court chemin...'). De plus, on a $AO = r$, où r désigne le rayon de $\mathcal{D}$, et $OC < r$. Finalement, on obtient $AC < 2r = AB$.
4. Notons p et q les deux points les plus proches de $\mathcal{P}$. Soit $\mathcal{D}$ le disque de rayon $[pq]$. Si $\mathcal{D}$ contient dans son intérieur un sommet r de $\mathcal{D}$, par la question précédente, on aurait $pr < pq$, ce qui contredit le choix de p et de q. L'intérieur de $\mathcal{D}$ ne contient ainsi pas de points de $\mathcal{P}$. Par le Lemme 1, pq est une arête de la triangulation de Delaunay.
5. On généralise la réponse de la question précédente. Soit pq une arête de T. Le graphe $T - pq$ contient deux composantes connexes qui sont des arbres : T_p contenant p et T_q contenant q. Soit $\mathcal{D}$ le disque de rayon $[pq]$. Si $\mathcal{D}$ contient dans son intérieur un sommet r de $\mathcal{D}$. Supposons que $r \in T_q$. Par la question 3, on a $pr < pq$. Mais alors, $(T - pq) + pr$ serait un arbre couvrant de $\mathcal{P}$ (car pr est une arête reliant T_p et T_q et de longueur total strictement inférieure à celle de T, ce qui contredit la définition de T. Si $r \in T_p$, on raisonne de même pour aboutir à une contradiction. Ainsi D ne contient pass de point de $\mathcal{P}$ dans son intérieur et pq est une arête de la triangulation de Delaunay de $\mathcal{P}$.
6. On calcule la triangulation de Delaunay de $\mathcal{P}$ en temps $O(n \log n)$. Celle-ci contient moins de $3n$ arêtes ($3(n-1) - |EC(\mathcal{P})|$ pour être précis...). On applique alors l'algorithme de Kruskal en temps $O(n \log n)$ sur ce graphe dont on value les arêtes par les distances entre les points (comme à la question 1). Finalement, on obtient un ACEM de $\mathcal{P}$ en temps $O(n \log n)$.