- Contrôle continu d'algorithmique géométrique -

- Corrigé rapide -

Barême :
Exercice 1 sur 3 points : 1-1 (0.5 si il manque un truc)-1
Exercice 2 sur 7 points : 1-1-1.5 (avec précisions)-1-1-2 (preuve propre)
Exercice 3 sur 6 points : 0.5-0.5-1.5 (avec détails)-1-1-1.5 (avec détails)
Total sur 16...

- Exercice 1 - Enveloppe convexe, *question de cours/td/tp* - 3pts -

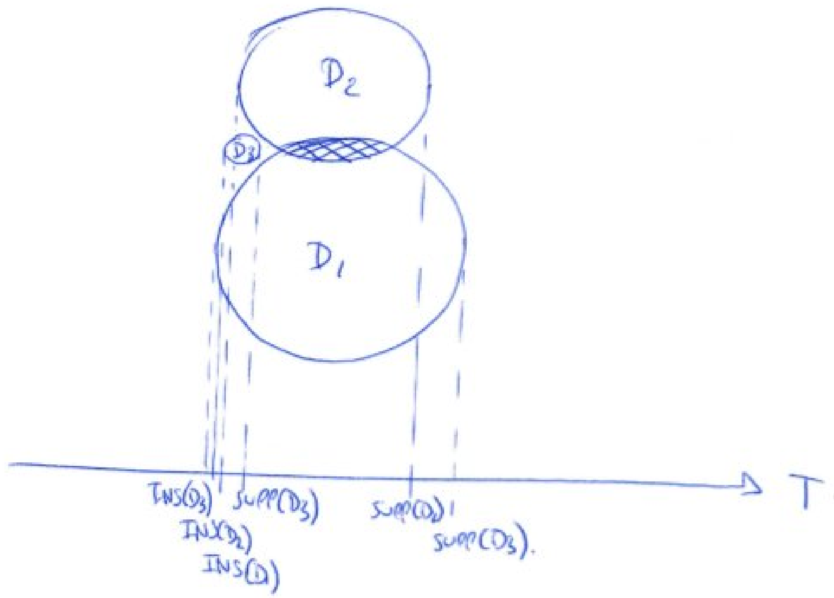
Cf Cours...

- Exercice 2 - Intersection de disques - 6pts -

1. On a D_1 et D_2 s'intersectent si et seulement si $dist(p_1, p_2) \leq r_1 + r_2$. En élevant au carré on obtient la condition $(x_1 - x_2)^2 + (y_1 - y_2)^2 \leq (r_1 + r_2)^2$ (qui a en plus l'avantage d'être un test entier...).
2. Pour chaque paire d'entier $\{i, j\}$ entre 1 et n, on teste si les disques D_i et D_j s'intersectent à l'aide de la procédure de la question précédente. Comme le test d'intersection se fait en $O(1)$ et qu'il y a $n(n-1)/2$ paires d'entiers entre 1 et n, on obtient un algo en $O(n^2)$.
3. (a) On s'inspire de l'algo d'intersections de segment. On suppose qu'une date dans T est associée avec un évènement : 'ajout/retrait du disque i'.

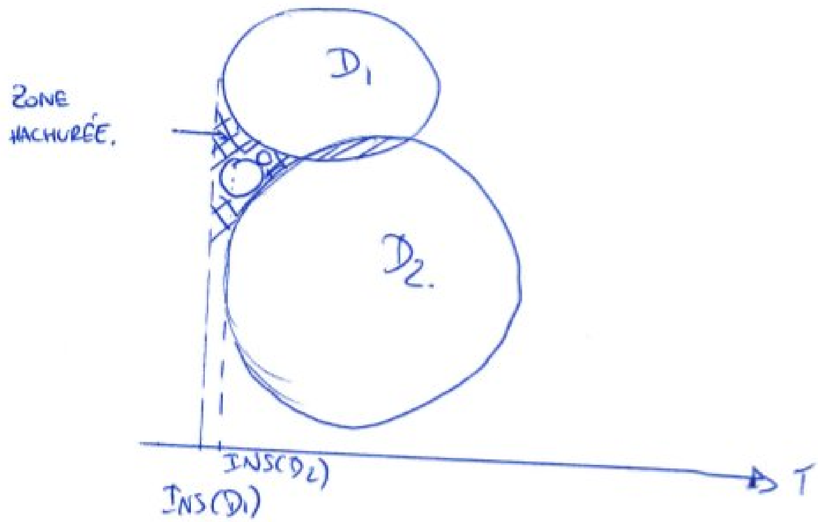
```
Trier T;
pour tous les d ∈ T, suivant l'ordre sur T faire
    si d correspond à l'insertion du disque i (d = x_i − r_i) alors
        INSERER(D_i,O);
        TESTER-INTERSECTION (D_i,PRÉDÉCESSEUR (D_i, O));
        TESTER-INTERSECTION (D_i,SUCCESSEUR (D_i, O));
    si d correspond à la suppression du disque i (d = x_i + r_i) alors
        TESTER-INTERSECTION (PRÉDÉCESSEUR (D_i, O),SUCCESSEUR (D_i, O));
        SUPPRIMER(D_i,O);
```

   (b) Par exemple, dans la figure suivante, l'intersection entre les disques D_1 et D_2 sera détectée à la suppression de D_3.



(c) T contient n date, son tri demande un temps en $O(n \log n)$.
À une date d'insertion : l'insertion le calul du prédécesseur et du successeur demande un temps en $O(\log n)$, les tests d'intersection se font en temps constant. Donc en tout, en temps en $O(\log n)$.
À une date de supression : de même, le tout se fait en $O(\log n)$.
Comme on fait n insertions et n suppressions, l'algo met en tout un temps en $O(n \log n)$.

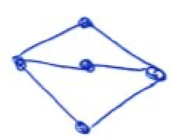
(d) Si l'algo détecte une intersection, elle existe et tout va bien. Supposons que l'algo ne détecte pas une intersection alors que celle-ci existe. Notons D_1 et D_2 deux disques qui s'intersectent et tels que leur intersection I commence le plus à gauche possible. De plus, on peut supposer que l'insertion de D_1 à lieu avant celle de D_2 et que D_1 est au dessus de D_1 (voir figure ci-dessous). Si à l'insertion de D_2, le successeur de D_2 est D_1 alors I est détectée. Sinon, celà signifie qu'il existe des disques entre D_1 et D_2 au moment de l'insertion de D_2 (zone hachurée sur la figure).



On considère alors D_3 le disque inclus dans cette zone et terminant le plus à droite. Comme I est l'intersection de disques la plus à gauche possible, D_3 n'intersecte ni D_1, ni D_2. Du coup, à la supression de D_3, on doit détecter l'intersection de D_1 et D_2, ce qui contredit l'hypothèse. L'algo détecte ainsi bien une intersection si il en existe une parmi l'ensemble de disques de départ.

- Exercice 3 - Quadrangulation - 6 pts -

1. Comme la surface considérée est sans trou, on peut la déformer continuement pour obtenir une sphère (de façon imagée, 'en soufflant dedans'). Ensuite, on a vu en cours qu'un graphe dessiné sans croisement sur une sphère est un graphe planaire.
2. Deux cycles de longueur 4 partageant une arête convient par exemple (voir figure).



3. On somme pour toutes les faces de G le nombre d'arêtes bordant la face en question. On note S la somme obtenue. D'une part, chaque arête est comptée deux fois : $S = 2m$ et d'autre part, chaque face contient exactement 4 arêtes donc $S = 4f$.
   On a donc $2m = 4f$ et $n - m + f = 2$, en éliminant f, on obtient $m = 2n - 4$.
4. La somme des degré des sommets d'un graphe vaut deux fois son nombre d'arêtes. Si tous les degrés sont au moins 4, on a $2m \geq 4n$ soit $m \geq 2m$, ce qui contredit $m \leq 2n-4$. Ainsi, il existe un sommet de G de degré inférieur ou égal à 3.
5. Si G n'a que des sommets de degré 4 ou plus, en sommant les degrés de tous les sommets du graphes, on obtiendrait au moins $4n$. Comme la somme des degrés d'un graphe vaut deux fois son nombre d'arêtes, on obtient $2m \geq 4n$ soit $m \geq 2n$, ce qui contredit la réponse à la question précédente.

6. Si on reprend la formule des degré de la question précédente, on obtient $2m = 3n$. Avec $m = 2n - 4$ prouvé à la question 3., on obtient $n = 8$ et $m = 12$. À partir de là, on peut argumenter : on part d'une face F d'une représentation de G, c'est un cycle de longueur 4. Les troisièmes voisins de chacun des sommets de F sont distincts (à préciser possiblement...). On a ainsi nos 8 sommets. les sommets qui n'appartiennent pas à F sont reliès par un cycle de longueur 4, et on obtient finalement bien un cube.
7. On suppose que chaque sommet de G est repéré par un numéro (inférieur ou égal à n, donc encodé sur au plus $\log n$ bits). On pose $G_0 = G$ puis, pour $i = 0, \ldots, n-1$, on choisit un sommet v_i de degré inférieur ou égal à 3 dans G_i et on note $G_{i+1} = G_i \setminus \{v_i\}$. De plus, pour $i = 0, \ldots, n-1$, on note L_i la liste des voisins de v_i dans G_i. La liste L_i contient au plus 3 éléments donc se code en au plus $3 \log n$ bits. De plus lorsqu'on savoir si une arête $v_i v_j$ existe, on cherche si v_j appartient à L_i puis, ce n'est pas le cas, si v_i appartient à L_j. Ce qui se fait en au plus 6 tests.