Olympiades Régionales d'Informatique Franco-Australiennes

13 mars 2016

Durée : 4 heures

3 problèmes

Problème 1
Hooliganism

Fichier d'entrée : *entréee standard*
Fichier de sortie : *sortie standard*

Limites de temps et de mémoire : 1 seconde, 256 MB

La ville de Sydney est à la merci des « hooligans » depuis quelques temps. En effet, ces jeunes complètement hors de contrôle rôdent toute la nuit en voiture, effrayant les hippopotames et roulant sur des marguerites. Vous devez absolument trouver un moyen d'arrêter ces délinquants une fois pour toutes.

Sydney possède N intersections, numérotés de 1 à N, reliées par R routes. Chaque nuit, les routes deviennent à sens unique et vous pouvez choisir leur sens. Toutefois, afin d'éviter les bouchons, chaque intersection a une limite sur le nombre de routes qui peuvent en partir.

Vous devez choisir le sens des routes de façon à ce que le nombre de routes quittant chaque intersection ne soit pas au-delà de la limite, et que les hooligans ne puissent plus sévir. Ceci implique donc que les hooligans ne doivent pouvoir emprunter chaque intersection qu'une seule fois au maximum, ainsi il ne pourront plus faire des tours en voiture.

Entrée

- La première ligne contient deux entiers N et R, séparés par une espace, donnant respectivement le nombre d'intersections et de routes.
- Les N lignes suivantes contiennent chacune un entier a_i, le nombre maximum de routes à sens unique qui peuvent partir de l'intersection i.
- Les R lignes suivantes contiennent chacune deux entiers b et c, représentant une route entre les intersections b et c.

Sortie

Si la ville n'a pas les moyens d'enrayer le hooliganisme, votre programme doit afficher `IMPOSSIBLE` sur la sortie.

Dans le cas contraire, la sortie doit comporter R lignes, contenant chacune deux entiers b_i et c_i, signifiant que la route à sens unique va de b_i vers c_i. Toutes les routes doivent être décrites mais peuvent l'être dans n'importe quel ordre.

Exemple d'entrée 1

```
3 3
0
2
0
1 2
1 3
2 3
```

Exemple de sortie 1

```
IMPOSSIBLE
```

Explication 1

Les intersections 1 et 3 sont reliées par une route, mais aucune n'est autorisée à avoir de route qui en parte, il est donc impossible de choisir un sens pour les routes qui satisfassent ces contraintes.

Exemple d'entrée 2

```
3 3
1
1
1
1 2
1 3
2 3
```

Exemple de sortie 2

```
IMPOSSIBLE
```

Explication 2

Dans cet exemple, il est possible de choisir le sens des routes de façon à satisfaire les limites à chaque intersection. Toutefois, la seule façon de faire est de créer un cycle entre les intersections 1, 2 et 3 dans cet ordre, ou dans l'ordre inverse. Ce cycle permettrait aux hooligans de passer plus d'une fois à certaines intersections, le problème est donc insoluble.

Exemple d'entrée 3

```
3 3
1
2
0
1 2
1 3
2 3
```

Exemple de sortie 3

```
1 3
2 1
2 3
```

Explication 3

Une fois que les hooligans ont atteint l'intersection 3, ils ne peuvent plus en partir et sont donc bloqués. Or l'intersection 3 est la seule accessible depuis l'intersection 1, donc les hooligans qui atteigent l'intersection 1 seront bloqués. La situation est la même pour l'intersection 2 car les hooligans ne peuvent plus atteindre que les intersections 1 et 3 et se retrouveront donc bloqués. Les hooligans ne peuvent donc pas passer plus d'une fois à chaque intersection.

Sous-tâches & contraintes

Pour toutes les sous-tâches, $1 \leq R \leq 100\ 000$ et $2 \leq N \leq 100\ 000$. De plus, deux intersections ne sont reliées que par une route au plus, et aucune intersection n'est reliée à elle-même. Notez qu'il n'est pas toujours possible de se rendre de n'importe quelle intersection à n'importe quelle autre.

- Pour la sous-tâche 1 (20 points), $a_i = N$, c'est-à-dire que le nombre maximum de routes qui peuvent partir d'une intersection est le même que le nombre d'intersections.
- Pour la sous-tâche 2 (20 points), chaque intersection est reliée à exactement deux autres intersections.
- Pour la sous-tâche 3 (30 points), $1 \leq R \leq 100\ 000$ and $2 \leq N \leq 1\ 000$.
- Pour la sous-tâche 4 (30 points), aucune autre contrainte.

Calcul du score

Le score pour chaque test d'entrée sera de 100 % si une bonne réponse est écrite sur la sortie, ou de 0 % sinon.

Problème 2
No Ball

Fichier d'entrée : *entrée standard*
Fichier de sortie : *sortie standard*

Limites de temps et de mémoire : 2 secondes, 512 MB

Aujourd'hui vous partez en retraite, et comme toutes les employés de *Graphland*, vous allez vous installer dans la région de *Flatland*. Plein de nostalgie, vous vous rappelez la belle époque où vous jouiez au cricket avec vos amis. Vous vous souvenez des discussions sans fin pour décider de qui devait apporter le jeu et où vous alliez jouer. Il ne vous en faut pas plus pour décider de construire une maison qui vous permettra, vous et vos amis, d'avoir un lieu de rencontre et un bon terrain de jeu.

Flatland est représenté par une grille numérotée. Le coin supérieur gauche a pour coordonnées $(0, 0)$, et ces coordonnées augmentent lorsque l'on se déplace vers la droite ou vers le bas. Chacun de vos N amis habite sur une case différente. Malheureusement, *Flatland* est immense et il est impossible de contacter tous vos amis. Plus précisément, vous avez le choix entre Q rectangles de communication et vous ne pourrez contacter que les amis situés dans le rectangle que vous choisirez. Un rectangle de communication dont les coins supérieur gauche et inférieur droit sont (R_1, C_1) et (R_2, C_2) contient une case (r, c) si $R_1 \leq r \leq R_2$ et $C_1 \leq c \leq C_2$.

Vous décidez donc de trouver un meilleur emplacement pour construire votre maison pour chaque rectangle. Cet emplacement est celui qui minimise la somme des distances de Manhattan à vos amis qui sont dans le rectangle. En cas d'égalité, prenez celle qui a d'abord le plus petit numéro de ligne, puis le plus petit numéro de colonne.

Notez que bien que vos amis habitent tous à des emplacements différents, vous êtez autorisé à construire une maison au même emplacement que l'un d'eux [1].

Puisque vous êtes cet ami qui « passe son temps à coder », il vous semble naturel d'écrire un programme pour trouver les meilleurs emplacements pour votre maison.

Entrée

- La première ligne de l'entrée contient deux entiers N et Q, séparés par une espace, représentant respectivement le nombre de vos amis et de rectangles de communications.
- Les N lignes suivants contiennent chacune deux entiers r_i et c_i donnant le numéro de la ligne et de la colonne où habite votre i^{eme} ami.
- Les Q lignes suivants contiennent chacune quatres entiers R_1, C_1, R_2, C_2 donnant le coin supérieur gauche et inférieur droit d'un rectangle de communication.

Sortie

Votre programme doit afficher Q lignes contenant chacune deux entiers : la ligne et la colonne du meilleur emplacement pour construire votre maison pour le rectangle de communication.

Exemple d'entrée

```
4 3
2 0
0 1
3 2
1 3
0 0 2 3
0 0 3 3
1 2 3 3
```

1. Flatland a en effet au moins trois dimensions, malgré ce que pourrait laisser penser son nom.

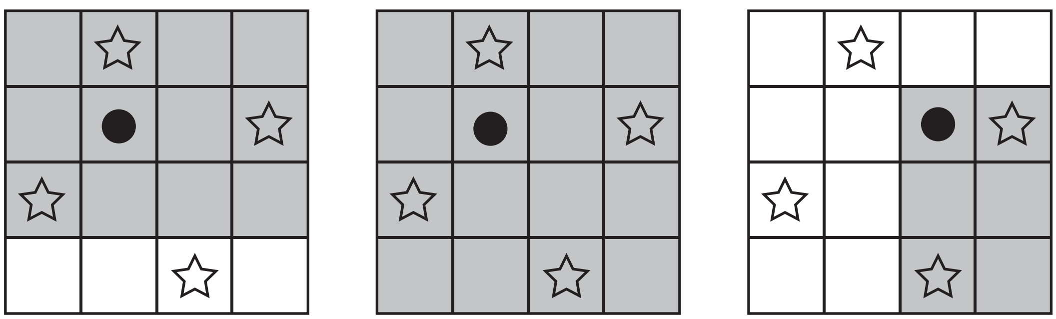
Exemple de sortie

```
1 1
1 1
1 2
```

Explication

Les dessins ci-dessous illustrent la situation : les maisons de vos amis sont représentées par des étoiles, le rectangle de communication par un fond gris et l'emplacement idéal par un cercle noir.

Dans le dessin de gauche, changer d'emplacement ne ferait qu'augmenter la somme des distances de Manhattan à vos amis. Dans le dessin du milieu, les 4 cases du centre ont la même somme de Manhattan à vos amis. Toutefois vous devez choisir celle la plus haute puis la plus à gauche. Dans le dessin de droite, toutes les cas ont la même somme de Manhattan, mais là encore vous devez choisir la case en haut à gauche.



Sous-tâches & contraintes

Pour toutes les sous-tâches, $1 \leq N \leq 100\,000$, $1 \leq Q \leq 10\,000$, $0 \leq r_i, c_i, R_1, C_1, R_2, C_2 \leq 1\,000\,000\,000$, $R_1 \leq R_2$ and $C_1 \leq C_2$. Vous êtes aussi assuré qu'il y a toujours au moins un ami dans chaque rectangle de communication.

- Pour la sous-tâche 1 (5 points), $1 \leq N, Q \leq 1\,000$, et $0 \leq r_i, c_i \leq 100\,000$.
- Pour la sous-tâche 2 (10 points), $0 \leq c_i \leq 100\,000$, et $r_i = 0$ pour tous vos amis.
- Pour la sous-tâche 3 (10 points), $0 \leq c_i \leq 1\,000\,000\,000$, et $r_i = 0$ pour tous vos amis.
- Pour la sous-tâche 4 (35 points), $0 \leq r_i, c_i \leq 100\,000$ pour tous vos amis, et $R_1 = C_1 = 0$ pour tous les rectangles de communication.
- Pour la sous-tâche 5 (30 points), $0 \leq r_i, c_i \leq 100\,000$.
- Pour la sous-tâche 6 (10 points), il n'y aucune autre contrainte.

Calcul du score

Le score pour chaque test d'entrée sera de 100 % si une bonne réponse est écrite sur la sortie, ou de 0 % sinon.

Problème 3
Signaux Lumineux

Fichier d'entrée : *entrée standard*
Fichier de sortie : *sortie standard*

Limites de temps et de mémoire : 1 seconde, 256 MB

Avec l'aide de l'excentrique Dr. Oui, vous avez réussi à sauver une partie du pays de la menace du *Virus*[2]. Malheureusement, le *Virus* s'est propagé à d'autres pays avant que n'ayez pu mettre en place le vaccin du Dr. Oui dans le reste du monde. Désormais, le Dr. Oui est certain que le seul espoir pour l'humanité est d'arriver à attirer l'attention des extraterrestres.

Pour cela, le Dr. Oui a construit $2^N - 1$ signaux lumineux spéciaux dans son jardin (qui se trouve être plat et très grand). Les signaux sont numérotés de 1 à $N - 1$, et le signal n° i se trouve aux coordonnéés (x_i, y_i). Ils ont tous des positions distinctes, et trois signaux distincts ne sont jamais alignés. Il ne reste plus qu'à les connecter.

Pour cela, le Dr. Oui à besoin de vous : vous devez choisir un certain nombre de paires de ces signaux pour les relier par des fils électriques rectilignes. Pour attirer l'attention des extraterrestres, il faut démontrer l'intelligence des humains, et le Dr. Oui est arrivé à la conclusion que les signaux doivent être connectés de façon à former un *arbre binaire complet*, où les signaux sont les nœuds et les fils sont les arêtes.

Précisément, vous devez placer $2^N - 2$ fils rectilignes joignant deux signaux tels que :

- deux fils ne s'intersectent pas en dehors des extrémités connectées à un même signal ;
- un unique signal est connecté à deux autres (la « racine » de l'arbre) ;
- 2^{N-1} signaux exactement ne sont connectés qu'à un autre signal, et le plus court chemin vers la racine est constitué d'exactement $N - 1$ fils ; ce sont les « feuilles » de l'arbre ;
- les $2^{N-1} - 2$ autres signaux, qui ne sont ni la racine, ni des feuilles, sont connectés à exactement trois autres signaux ;
- la longueur totale de fil est minimale.

La longueur de fil entre deux signaux aux points (x_i, y_i) et (x_j, y_j) est $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ (la distance euclidienne entre les deux signaux). La longueur totale de fil utilisée est la somme des longueurs de tous les fils placés entre les paires de signaux choisis.

Il ne vous est pas demandé de donner la meilleure solution possible. Vous devez simplement utiliser la longueur totale de fil la plus petite que vous pouvez. Votre solution sera comparée à celle des juges, et les meilleures solutions obtiendront plus de points. Voir la section *Score* pour plus de détails.

Entrée

- La première ligne contiendra un unique entier N, le nombre de signaux étant $2^N - 1$.
- Les $2^N - 1$ lignes suivantes contiendront chacune deux entiers séparés par une espace. La i-ème ligne contiendra x_i y_i, les coordonnées du signal n° i.

Sortie

Votre programme doit afficher $2^N - 2$ lignes sur la sortie, chacune contenant deux entiers a_i b_i, indiquant qu'il y a un fil rectiligne entre le signal n° a_i et le signal n° b_i. Ces lignes peuvent être dans un ordre quelconque, et l'ordre des entiers sur chaque ligne n'a pas d'importance.

2. cf. FARIO 2011, « Le Virus »

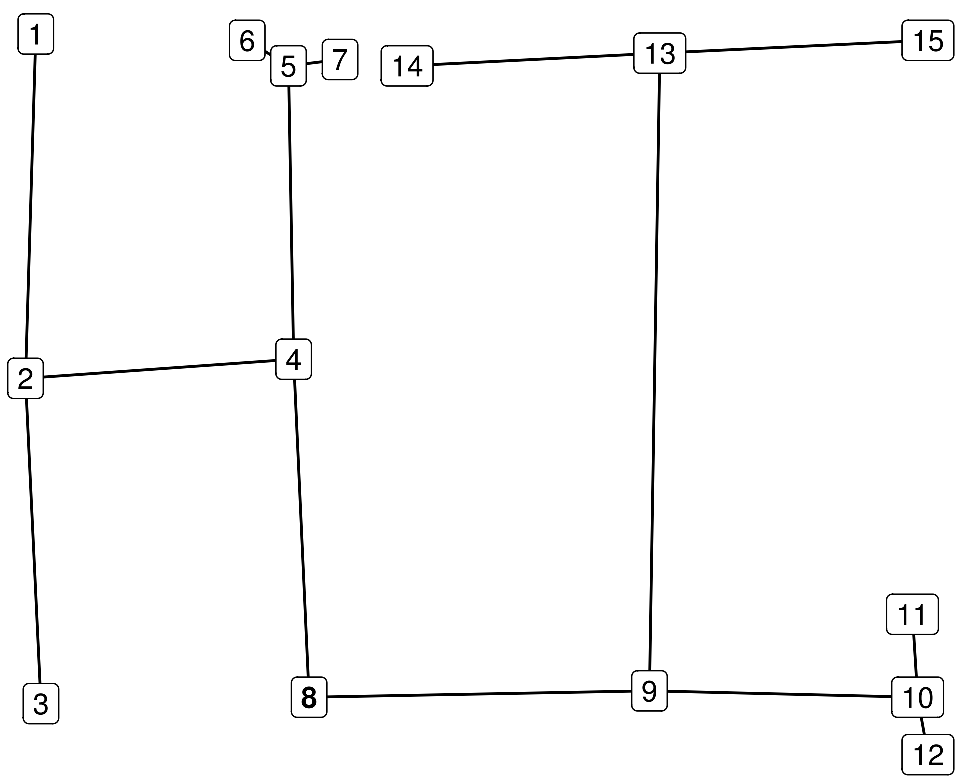
Exemple d'entrée

```
4
2 0
0 54
3 105
52 51
51 5
43 1
61 4
55 104
121 103
173 104
172 91
175 113
123 3
74 5
175 1
```

Exemple de sortie

```
8 4
8 9
4 2
4 5
9 13
9 10
2 3
2 1
5 6
5 7
13 14
13 15
10 11
10 12
```

Explication



On peut voir que le signal lumineux n° 8 est la racine de l'arbre, puisque c'est le seul nœud connecté à deux autres. Il a pour descendants les signaux n° 4 et 9 au niveau suivant. Le signal n° 4 a pour descendants 2 et 5, puis le n° 2 a pour descendants 3 et 1, qui sont des feuilles. Ainsi les signaux n° 1 et 3 sont à une profondeur 3 par rapport à la racine 8, et tous les nœuds internes ont exactement deux descendants. De même, toutes les autres feuilles $(6, 7, 14, 15, 11, 12)$ sont à une profondeur 3 par rapport à la racine. Cet arbre est donc un arbre binaire complet. C'est de plus un arbre valide puisque les fils ne s'intersectent pas.

En sommant les longueurs des fils, on obtient un total de $616,6759$

Score

Pour chaque fichier test, le score de votre solution sera déterminé comme suit :

- si votre solution ne décrit pas un arbre binaire complet sans intersections, elle obtiendra un score de 0 ;
- sinon, soit P la distance moyenne entre deux signaux multipliée par $2^N - 2$, et Q la longueur totale de fil minimale obtenue par une des solutions des juges. Votre score sera déterminé sur une échelle linéaire, P donnant 0% et Q donnant 90%, avec un minimum de 30% et un maximum de 100%. Autrement dit votre score sera de $\min(100, \max(30, 90 \times \frac{P-X}{P-Q}))$, où X est la longueur totale de fil utilisée par votre solution. On garantit que $P > Q$.

Pour chaque sous-tâche, votre solution recevra le score minimum obtenu parmi tous les fichers tests de la sous-tâche.

Sous-tâches et contraintes

Pour toutes les sous-tâches, $0 \leq x_i, y_i \leq 1\ 000\ 000\ 000$.

- Pour la sous-tâche 1 (15 points), $N = 2$
- Pour la sous-tâche 2 (15 points), $N = 3$
- Pour la sous-tâche 3 (10 points), $N = 4$
- Pour la sous-tâche 4 (10 points), $N = 5$
- Pour la sous-tâche 5 (10 points), $N = 6$
- Pour la sous-tâche 6 (10 points), $N = 7$
- Pour la sous-tâche 7 (10 points), $N = 8$
- Pour la sous-tâche 8 (10 points), $N = 9$
- Pour la sous-tâche 9 (10 points), $N = 10$