Le Centre d'éducation en mathématiques et en informatique

Concours canadien d'informatique 2012

Niveau supérieur

Commanditaire :

University of Waterloo

Concours canadien d'informatique — Niveau supérieur

Directives à l'intention des participantes et des participants

1. Vous pouvez participer à un concours seulement. Pour participer au concours de niveau intermédiaire, il faut choisir l'autre trousse de problèmes.

2. Sur le formulaire **Information à l'intention des élèves**, indiquez que vous participez au concours de niveau **supérieur**.

3. Vous avez trois (3) heures pour accomplir le travail.

4. Vous pouvez prendre pour acquis que :
   - toutes les entrées se trouvent dans des fichiers nommés `sX.in`, X étant le numéro du problème ($1 \leq X \leq 5$). Le fichier d'entrées du Problème S1 est donc `s1.in`.
   - toutes les sorties se font par l'écran.

   Puisque les entrées se trouvent dans des fichiers, il n'y aura aucune sollicitation. Les sorties doivent être IDENTIQUES à celles des exemples de sorties, par rapport à l'ordre, aux espaces, etc.

5. Vous devez faire votre propre travail. Les tricheurs seront punis sévèrement.

6. Il est interdit de faire appel à des caractéristiques auxquelles le juge, votre enseignant, n'a pas accès pendant l'évaluation de votre programme.

7. Vous pouvez consulter des livres et du matériel écrit. Tout matériel susceptible d'être lu électroniquement (par exemple un programme que vous avez écrit) est *interdit*. Cependant, vous pouvez faire appel aux bibliothèques reconnues pour vos langages de programmation ; par exemple, STL pour C++, java.util.*, java.io.* et d'autres pour Java, et ainsi de suite.

8. Vous devez vous limiter aux applications de programmation ordinaires (éditeurs, compilateurs, débogueurs). Toutes les autres applications sont **interdites**. Leur utilisation entraînera une disqualification.

9. Utilisez des noms de fichier qui sont propres à chaque problème : par exemple, `s1.pas` ou `s1.c` ou `s1.java` (ou tout autre suffixe de fichier approprié) pour le problème S1. Ceci facilitera la tâche de l'évaluateur.

10. Votre programme sera exécuté avec des fichiers d'essais différents de ceux qui figurent comme exemples. Assurez-vous de vérifier votre programme au moyen d'autres fichiers d'essais. Pour certains problèmes, particulièrement les problèmes 4 et 5, des solutions peu performantes peuvent faire perdre des points. Assurez-vous d'avoir un code aussi performant que possible par rapport au temps.

11. Les deux premiers participants de chaque région du pays recevront une plaque et une somme de 100 $. Leur école recevra aussi une plaque. Les régions sont :
    - L'ouest (de la C.-B. au Manitoba)
    - Le nord et l'est de l'Ontatio
    - Toronto métropolitain
    - Le centre et l'ouest de l'Ontario
    - Le Québec et les provinces de l'Atlantique

12. Si vous vous placez parmi les 20 premiers participants et participantes dans le concours du niveau supérieur, vous serez invité à participer à l'Étape 2 du CCI, qui aura lieu à l'Université de Waterloo au mois de mai 2012. Si vous vous placez parmi les premiers lors de l'Étape 2, vous serez invité à participer à l'équipe qui représentera le Canada à IOI 2012, en Italie. Notez que vous devez connaître C, C++ ou Pascal si vous êtes invité à l'Étape 2. Mais d'abord, vous devez réussir le concours d'aujourd'hui !

13. Consultez le site web du CCI à la fin du mois de mars pour connaître votre classement dans ce concours et pour connaître le nom des gagnants. Voici l'adresse :

    `www.cemc.uwaterloo.ca/ccc`

Problème S1 : Passe-moi pas le ballon !

Description du problème
Au soccer du CCI, on utilise des règlements un peu différents. Un but est marqué seulement si les quatre derniers joueurs, en ordre, qui ont touché le ballon avant qu'il ne pénètre dans le filet portent des numéros de maillot qui sont en ordre strictement croissant, le plus grand numéro étant porté par le joueur qui marque le but.

Les maillots sont numérotés de 1 à 99 (et chaque maillot est porté par un seul joueur).

Étant donné le numéro du joueur qui marque le but, vous devez déterminer combien il y a de combinaisons possibles de joueurs qui ont pu produire un but valide.

Précisions par rapport aux entrées
L'entrée sera un entier strictement positif J ($1 \leq J \leq 99$), qui représente le numéro du maillot du joueur qui a marqué le but.

Précisions par rapport aux sorties
La sortie consistera en une ligne contenant le nombre de combinaisons possibles de joueurs qui auraient pu produire un but valide, J étant le numéro du maillot du joueur qui a marqué.

Exemple d'entrée 1
```
4
```

Sortie pour l'exemple d'entrée 1
```
1
```

Exemple d'entrée 2
```
2
```

Sortie pour l'exemple d'entrée 2
```
0
```

Exemple d'entrée 3
```
90
```

Sortie pour l'exemple d'entrée 3
```
113564
```

Problème S2 : Nombres aromatiques

Description du problème

Cette question porte sur le calcul de la valeur de nombres *aromatiques*, qui sont des nombres exprimés comme combinaisons de chiffres arabes et de chiffres romains.

Un nombre aromatique est de la forme $ARARAR \ldots AR$, chaque A étant un chiffre arabe et chaque R étant un chiffre romain. Chaque paire AR contribue une valeur décrite ci-dessous et en additionnant ou soustrayant les valeurs des paires, on obtient la valeur du nombre aromatique.

Un chiffre arabe A peut être égal à 0, 1, 2, 3, 4, 5, 6, 7, 8 ou 9. Un chiffre romain R peut être égal à une des sept lettres I, V, X, L, C, D ou M. Chaque chiffre romain a une valeur particulière, soit :

Chiffre	I	V	X	L	C	D	M
Valeur	1	5	10	50	100	500	1000

La valeur d'une paire AR est égale à A fois la valeur de R. Normalement, on additionne les valeurs des paires du nombre pour obtenir la valeur du nombre. Or, lorsqu'on a deux paires consécutives $ARA'R'$ où R' a une valeur *strictement plus grande* que celle de R, la valeur de la paire AR doit être *soustraite* du total au lieu d'être *ajoutée*.

Par exemple, la valeur du nombre 3M1D2C est égale à $3 * 1000 + 1 * 500 + 2 * 100$, ou 3700 et la valeur du nombre 3X2I4X est égale à $3 * 10 - 2 * 1 + 4 * 10$, ou 68.

Vous devez écrire un programme qui calcule la valeur de nombres aromatiques.

Précisions par rapport aux entrées
L'entrée est un nombre aromatique valide composé de 2 à 20 chiffres.

Précisions par rapport aux sorties
La sortie est la valeur du nombre aromatique.

Exemple d'entrée 1
```
3M1D2C
```

Sortie pour l'exemple d'entrée 1
```
3700
```

Exemple d'entrée 2
```
2I3I2X9V1X
```

Sortie pour l'exemple d'entrée 2
```
-16
```

Problème S3 : Acidité absolue

Description du problème

On veut connaître les niveaux d'acidité dans une très longue rivière pour juger de l'état de santé de la rivière. On a placé N détecteurs ($2 \leq N \leq 2\,000\,000$) dans la rivière. Chaque détecteur prélève le niveau d'acidité et donne un résultat sous forme d'un nombre entier R ($1 \leq R \leq 1\,000$). Vous rassemblez les résultats de ces prélèvements.

Vous voulez connaître la fréquence des résultats de ces prélèvements ainsi que la différence absolue entre les deux résultats les plus fréquents.

Si plus de deux résultats ont la plus grande fréquence, la différence absolue retenue doit être la *plus grande* des différences absolues entre deux résultats ayant cette fréquence. S'il y a un seul résultat ayant la plus grande fréquence, mais plus d'un résultat ayant la deuxième plus grande fréquence, la différence absolue retenue doit être la *plus grande* différence absolue entre le résultat le plus fréquent et les résultats ayant la deuxième plus grande fréquence.

Précisions par rapport aux entrées

La première ligne d'entrée contiendra un entier N ($2 \leq N \leq 2\,000\,000$), représentant le nombre de détecteurs. Chacune des N lignes suivantes contiendra le résultat obtenu par le détecteur, soit un entier R ($1 \leq R \leq 1\,000$). Vous pouvez supposer que chaque entrée comprendra au moins deux résultats.

Précisions par rapport aux sorties

La sortie sera un entier qui représente la différence absolue entre les deux résultats les plus fréquents, en tenant compte des règlements ci-dessus en cas d'égalité.

Exemple d'entrée 1

```
5
1
1
1
4
3
```

Sortie pour l'exemple d'entrée 1

```
3
```

Exemple d'entrée 2

```
4
10
6
1
8
```

Sortie pour l'exemple d'entrée 2

```
9
```

Problème S4 : Jeu de monnaie

Description du problème
Lorsqu'elle s'ennuie, Anne Coderre aime jouer au jeu suivant sur une table à l'aide de pièces de monnaie. Elle prend un ensemble de pièces distinctes de monnaie qu'elle place en ligne droite. Par exemple, supposons qu'elle a une pièce U de 1 ¢ (ou 0,01 $), une pièce C de 5 ¢ (ou 0,05 $) et une pièce D de 10 ¢ (ou 0,10 $). Elle les place en ligne droite de façon arbitraire (par exemple, D C U) et elle les déplace de manière à les replacer en ordre strictement croissant selon leur valeur, par exemple, U C D (c.-à-d. 0,01 $, 0,05 $, 0,10 $). Voici les règlements du jeu :

– L'alignement initial des pièces de monnaie définit toutes les positions dans lesquelles on peut placer les pièces. Donc, on ne peut pas ajouter des positions pendant le jeu et même si une position n'est pas occupée à un moment particulier du jeu, la position continue d'exister.

– Le jeu consiste en une séquence de déplacements et à chaque déplacement, une pièce est déplacée à une position *adjacente*.

– Les pièces de monnaie peuvent être empilées les unes sur les autres et lors d'un déplacement, Anne prend toujours une pièce sur le dessus d'une pile pour la placer sur le dessus d'une pile adjacente.

– Lorsqu'elle place une pièce dans une pile, Anne ne peut pas la placer sur une pièce de valeur inférieure.

Pour simplifier, on donnera aux pièces de monnaie des valeurs qui sont des entiers consécutifs (p. ex., la pièce de 1 ¢ vaut 1, la pièce de 5 ¢ vaut 2 et la pièce de 10 ¢ vaut 3). Dans l'exemple précédent, Anne réussirait en 20 déplacements. Le tableau suivant indique les déplacements (XY signifie que la pièce X est par-dessus la pièce Y) :

Dépl.	Position 1	Position 2	Position 3
Départ	3	2	1
1	3	12	
2	13	2	
3	13		2
4	3	1	2
5	3		12
6		3	12
7		13	2
8	1	3	2
9	1	23	
10		123	

Dépl.	Position 1	Position 2	Position 3
11		23	1
12	2	3	1
13	2	13	
14	12	3	
15	12		3
16	2	1	3
17	2		13
18		2	13
19		12	3
20	1	2	3

À noter que pour certains alignements initiaux, il est impossible de réussir à replacer les pièces en ordre strictement croissant.

Précisions par rapport aux entrées
L'entrée est constituée d'un certain nombre de jeux d'essais. Un jeu d'essais contient deux lignes. La première ligne contient un entier n strictement positif ($n < 8$), ce qui correspond au nombre de pièces de monnaie. On peut supposer que les pièces auront pour valeurs 1, 2, 3, ..., n. La deuxième ligne contient les entiers de 1 à n placés dans un ordre quelconque, ce qui constitue la position de départ. Dans l'exemple précédent, l'entrée serait :

```
3
3 2 1
```

La fin des jeux d'essais est indiquée par un 0 seul sur une ligne.

Précisions par rapport aux sorties
Pour chaque jeu d'essais, la sortie sera une ligne qui indique soit le *nombre minimal* de déplacements qu'Anne peut faire pour réussir à placer les pièces en ordre ou, si c'est impossible de réussir, la ligne indiquera le mot `IMPOSSIBLE`.

Exemple d'entrée

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

Sortie pour l'exemple d'entrée

```
20
IMPOSSIBLE
```

Problème S5 : Trajet de souris

Description du problème
Vous êtes une souris qui vit dans une cage, dans un grand laboratoire. Le laboratoire est composé d'un réseau rectangulaire de cages placées en R rangées et C colonnes ($1 \leq R, C \leq 25$).

Pour vous laisser faire votre marche santé, on vous permet d'aller d'une cage à une autre. Vous pouvez aller d'une cage à une autre en allant vers la droite à une cage adjacente de la même rangée ou en allant vers le bas à une cage adjacente de la même colonne. Il est interdit d'aller en diagonale, vers la gauche ou vers le haut.

Votre cage est située dans un coin du laboratoire, dans la cage $(1, 1)$ (ce qui indique la 1re rangée du haut, 1re colonne à gauche). Vous voulez rendre visite à votre frère qui est dans la cage (R, C) (dernière rangée en bas, dernière colonne à droite), située dans le coin diagonalement opposé. Or, il vous faut éviter certaines cages qui contiennent des chats.

Votre frère, qui aime les nombres, aimerait connaître combien il y a de chemins, de votre cage à la sienne, qui ne passent pas par les cages contenant des chats. Vous devez écrire un programme qui calculera le nombre de ces chemins qui évitent les chats.

Précisions par rapport aux entrées
La première ligne d'entrée contiendra deux entiers, R et C, séparés d'une espace. Ces entiers représentent respectivement le nombre de rangées et le nombre de colonnes dans le laboratoire. La deuxième ligne d'entrée contient un entier K qui indique le nombre de cages contenant un chat. Chacune des K lignes suivantes donne, dans l'ordre, le numéro de la rangée et le numéro de la colonne d'une cage qui contient un chat. Aucune de ces K cages n'est répétée et chaque position donnée est valide. À noter que les chats ne seront pas dans les cages $(1, 1)$ ou (R, C).

Précisions par rapport aux sorties
La sortie sera un entier non négatif qui indique le nombre de chemins, de votre cage en position $(1, 1)$ à celle de votre frère en position (R, C), qui évitent les chats. Vous pouvez supposer que ce nombre sera inférieur à $1\ 000\ 000\ 000$.

Exemple d'entrée 1
```
2 3
1
2 1
```

Sortie pour l'exemple d'entrée 1
```
2
```

Exemple d'entrée 2
```
3 4
3
```

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

Sortie pour l'exemple d'entrée 2

```
1
```