Concours canadien d'informatique

Règles à l'intention des participantes et des participants du niveau supérieur

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. Les noms des fichiers sont `s`N**.in**, N étant le numéro du problème (c.-à-d. que le fichier d'entrées du Problème S1 est `s1.in`).
   – toutes les sorties se font par l'écran.

   Dans certains problèmes, on peut vous demander de fournir une sollicitation pour l'utilisateur. Si aucune sollicitation n'est requise, il n'est pas nécessaire d'en fournir une. 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.*, etc. 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'essai différents de ceux qui figurent comme exemples. Assurez-vous de vérifier votre programme au moyen d'autres fichiers d'essai.
11. Pour certains problèmes, comme le problème 5, il faudra soumettre une solution efficace pour se mériter le maximum des points.
12. 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

13. 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 début du mois de mai 2006. Si vous vous placez parmi les 4 premiers lors de l’Étape 2, vous serez invité à participer à l’équipe qui représentera le Canada à IOI 2006, au Mexique. 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 !

14. Consultez le site web du CCI à la fin du mois de mars pour connaître votre classement dans ce concours, pour voir comment on pouvait résoudre les problèmes et pour connaître le nom des gagnants. Voici l’adresse :

`www.cemc.uwaterloo.ca/ccc`

Problème S1 : Maternité

Description du problème

Alice et Bob sont des mouches des cerises. Ils sont les fiers parents d'une jolie petite mouche. Malheureusement, à cause d'une petite erreur, les infirmières de la salle de maternité ne savent pas trop laquelle des mouches est leur fille. Heureusement que vous êtes là pour les aider. Si vous connaissez le profil génétique des parents, pouvez-vous déterminer laquelle des mouches est leur fille ?

Vous avez appris, en biologie, que les attributs (la couleur des yeux, la couleur des cheveux, etc.) sont transmis d'une génération à une autre. Un *gène* particulier contrôle chaque attribut. Chaque gène est représenté par deux *allèles*, qui sont responsables des variations de ces attributs. Par exemple, il y a deux versions du gène qui contrôle la couleur des yeux, l'un pour les yeux bruns et l'autre pour les yeux bleus. Pour chaque attribut d'une mouche des cerises, il y a deux allèles, l'un provenant d'un parent et le deuxième provenant de l'autre.

Si deux allèles sont différents, alors l'un d'eux *l'allèle dominant*, influencera l'apparence de la mouche. L'autre allèle *l'allèle récessif*, n'a aucune influence sur l'apparence de la mouche. Habituellement, l'allèle dominant est représenté par une lettre majuscule et l'allèle récessif est représenté par une lettre minuscule. Par exemple, pour la couleur des yeux, l'allèle des yeux bruns est représenté par B (dominant) et l'allèle des yeux bleus est représenté par b (récessif). Si la mouche reçoît BB ou Bb, elle aura les yeux bruns ; si elle reçoît bb, elle aura les yeux bleus.

Lors de la reproduction, chaque parent transmet un allèle par gène à son enfant. On peut utiliser une *grille de Punnett* pour représente les combinaisons possibles d'allèles Par exemple, voici la *grille de Punnett* pour des enfants d'Alice (Bb) et de Bob (Bb) :

		Bob	
		B	**b**
Alice	**B**	BB	Bb
	b	Bb	bb

Malheureusement, les profils génétiques complets des bébés ne sont pas disponibles, puisqu'il faut quelques semaines pour les obtenir. Nous n'avons que les attributs des enfants – la couleur des cheveux, des yeux, etc. Pouvez-vous utiliser ces renseignements, de même que le profil génétique des parents, pour déterminer les bébés qui ne peuvent pas être les leurs ?

Précisions par rapport aux entrées

Vous avez de la chance. Nos mouches des cerises n'ont que cinq gènes, notés de A à E. Les deux premières lignes d'entrées décrivent respectivement la mère et le père. Chacune de ces lignes est composée de cinq paires de lettres, soit une paire par gène. Chaque paire décrit les deux allèles du parent pour le gène en question. Les allèles se rapportent, dans l'ordre, aux gènes de A à E.

La troisième ligne présente un nombre X, $(X \leq 10)$, soit le nombre de bébés qu'il faut vérifier. Les X lignes suivantes décrivent les *attributs* des X bébés. Chaque ligne comprend cinq lettres (de A à E). Une lettre majuscule indique que le bébé affiche l'attribut de l'allèle dominant, tandis qu'une lettre minuscule indique que le bébé affiche l'attribut de l'allèle récessif. Par exemple, si le bébé a les yeux bruns, la lettre B sera utilisée ; si le bébé a les yeux bleus, la lettre b sera utilisée.

Précisions par rapport aux sorties

Pour chaque bébé, imprimez la ligne « Enfant possible. » si le bébé peut être le leur ou la ligne « Pas leur enfant ! » si le bébé ne peut pas être le leur.

Exemple d'entrée

```
AABbCcddEe
AaBBccddee
5
ABCdE
aBcdE
ABcdE
ABCde
ABcDe
```

Sortie pour l'exemple

```
Enfant possible.
Pas leur enfant !
Enfant possible.
Enfant possible.
Pas leur enfant !
```

Explications par rapport à la sortie

Le 2^e bébé ne peut être leur enfant. Le bébé affiche l'attribut *a*, qui est récessif ; l'enfant doit donc avoir les allèles *aa*. Or, le premier parent a *AA* ; il ne peut donc pas avoir transmis un *a*. Le 5^e bébé ne peut être leur enfant. Le bébé affiche l'attribut *D*. Or, les deux parents ont les allèles *dd* ; il ne peuvent pas avoir transmis un *D*. Les autres bébés pourraient être les leurs !

Problème S2 : L’attaque des textes chiffrés

Description du problème

Rima est déchiffreure. Elle sait que des personnes méchantes (Monsieur X et Monsieur Z) se transmettent des messages secrets au sujet de choses très vilaines.

Rima a intercepté un message *en clair*, ainsi que le message *chiffré* correspondant. Un texte en clair est un texte qui n’a pas été chiffré (encodé), c’est-à-dire un texte que l’on peut comprendre en français ou en anglais, tandis qu’un texte chiffré est un message qui a été encodé et qui semble être du charabia. Pour encoder un message, chaque lettre est remplacée par une autre lettre, ce qui rend le message inintelligible.

Or, Rima est une déchiffreure hors pair. Elle connaît l’algorithme qu’utilisent Monsieur X et Monsieur Z. Elle sait que pour encoder leurs messages, ils font correspondre chaque lettre à une autre (qui peut à l’occasion être la même). Évidemment, cette correspondance doit être biunivoque, ce qui veut dire que chaque lettre du texte en clair doit correspondre à exactement une lettre du message chiffré et que chaque lettre du message chiffré doit correspondre à exactement une lettre du texte en clair.

Votre tâche est d’automatiser le décodage de Rima et ainsi sauver l’univers.

Précisions par rapport aux entrées

L’entrée est composée de 3 chaînes, une chaîne par ligne. La première chaîne est le texte en clair que Rima connaît. La deuxième chaîne est le texte chiffré correspondant. La troisième chaîne est un autre texte chiffré. Vous pouvez prendre pour acquis que chaque chaîne est composée d’au moins 1 caractère et d’un maximum de 80 caractères. Vous pouvez aussi prendre pour acquis qu’il y a exactement 27 caractères valides, soit les lettres majuscules (de A à Z) et le caractère qui représente une espace vide (‘ ’). Il n’y aura donc aucune ponctuation, ni lettres minuscules, ni caractères spéciaux (comme ‘ !’ ou ‘@’) dans les messages en clair ou dans les messages chiffrés.

Précisions par rapport aux sorties

La sortie est une chaîne de texte en clair qui correspond au deuxième texte chiffré de l’entrée. Or, il se peut qu’il soit impossible de déterminer la lettre au clair qui correspond à une lettre du deuxième texte chiffré. Dans ce cas, la lettre qui manque doit être indiquée par un point (‘.’).

Exemple d’entrée 1

```
PORTEZ CE VIEUX WHISKY AU JUGE BLOND QUI FUME
QPSUF ADFAWJFVYAXIJTLZABVAKVHFACMPOEARVJAGVNF
OPVTATPNNFTANFDIBOUTAFUAWJMBJOTAIBAIBAIB
```

Sortie pour l’exemple 1

```
NOUS SOMMES MECHANTS ET VILAINS HA HA HA
```

Explications par rapport à la sortie

Remarquez que chaque caractère du texte en clair est dans le premier message, ce qui permet le décodage.

Exemple d’entrée 2

```
IL N Y A PAS ASSEZ DE LETTRES
BTSNSDSESFEMSEMMAQSGASTAXXZAM
BCFHMMBITASGASTASGAKHGAZ
```

Sortie pour l’exemple 2

```
I.P.SS..LE DE LE DE..DER
```

Explications par rapport à la sortie

Remarquez que les lettres chiffrées `C, H, I, K` ne peuvent être décodées, car elles ne paraissent pas dans le premier message chiffré. Le dernier message en clair est `IMPOSSIBLE DE LE DECODER`. Les autres lettres utilisées devraient correspondre dans les deux messages.

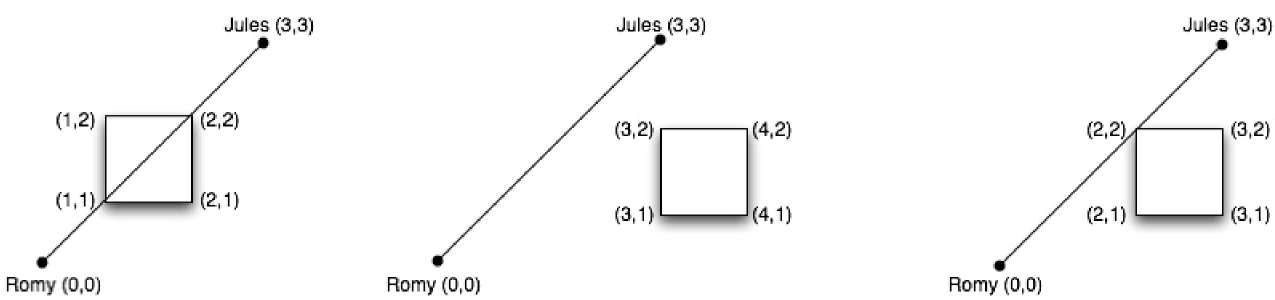
Problème S3 : Téléphone à boîtes de conserve

Description du problème

Romy et Jules ont pris l'habitude de se parler au moyen de leur téléphone cellulaire. Malheureusement, leurs parents ne s'aiment pas et ils ont interdit aux enfants de se parler. Les parents leur ont même enlevé leur cellulaire. Romy et Jules doivent donc trouver un autre moyen de communication. Après une recherche sur Internet, ils ont décidé de fabriquer un téléphone à boîtes de conserve.

Un téléphone à boîtes de conserve est formé de deux boîtes de soupe vides reliées par une ficelle. Pour l'utiliser, il faut que la ficelle soit bien tendue. Une personne parle et l'autre écoute. Il est important de ne pas toucher la ficelle, de manière qu'elle puisse vibrer et transmettre le son d'une boîtes de conserve à l'autre.

Pour que le téléphone fonctionne, il faut qu'il y ait une ligne directe entre leurs chambres. Pour déterminer s'il est possible de tirer une ficelle entre les chambres, Romy et Jules consultent une carte qui utilise des coordonnées. Chaque coordonnée est un entier. Observez les trois situations suivantes.



1. Vue bloquée 2. Vue libre 3. Vue bloquée (Se touchent)

Dans ces figures, « Romy » représente la fenêtre de Romy, au point (0,0), tandis que « Jules » représente la fenêtre de Jules, au point (3,3). Dans la première figure, un édifice bloque la ligne directe entre les fenêtres. Dans la deuxième figure, il n'y a rien qui bloque la ligne directe et ils peuvent installer un téléphone à boîtes de conserve. Dans la troisième figure, une ligne droite entre les fenêtres toucherait le coin d'un édifice. Puisque la ficelle ne peut être touchée, le téléphone à boîtes de conserve ne peut être installé. On considère que la ligne directe est bloquée.

Précisions par rapport aux entrées

La première ligne contient quatre entiers représentant les coordonnées de la fenêtre de Romy et de celle de Jules. Ainsi l'entrée x_R y_R x_J y_J représente les coordonnées (x_R, y_R) de la fenêtre de Romy et les coordonnées (x_J, y_J) de la fenêtre de Jules. Vous pouvez prendre pour acquis que $-1000 \leq x_R, x_J \leq 1000$ et $-1000 \leq y_R, y_J \leq 1000$. La deuxième ligne contient un seul entier n, $(0 \leq n \leq 100)$, qui indique le nombre d'édifices qu'il y aura dans les n lignes suivantes. Chaque édifice est spécifié sur une ligne distincte qui contient d'abord un entier indiquant le nombre de coins qui forment l'édifice. Cet entier est suivi des coordonnées entières des coins de l'édifice, dans le sens des aiguilles d'une montre ou dans le sens contraire. Aucun édifice n'a plus de 32 coins. Le premier exemple d'entrée et de sortie correspond à la première figure ci-haut.

Précisions par rapport aux sorties

La sortie est un nombre qui indique le nombre d'édifices qui bloquent ou qui touchent la distance à vue.

Exemple d'entrée

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

Sortie pour l'exemple

```
1
```

Problème S4 : Groupes

Description du problème

En mathématiques, un *groupe*, G, est une structure formée d'éléments et d'une opération (que l'on appellera $\times$) de manière que si x et y sont des éléments de G, $x \times y$ est aussi un élément de G. De plus, l'opération satisfait aux propriétés suivantes :

- Associativité : Pour tous les éléments x, y et z de G, $x \times (y \times z) = (x \times y) \times z$.
- Identité : Le groupe contient un élément appelé *identité* (on le nommera i) de manière que pour tout élément x de G, $x \times i = x$ et $i \times x = x$.
- Inverse : Pour chaque élément x du groupe, il y a un élément *inverse* (on le nommera x^{-1}), de manière que $x \times x^{-1} = i$ et $x^{-1} \times x = i$.

Les groupes sont utilisés dans une grande variété de situations. Par exemple, on s'en sert pour modéliser les états quantiques d'un atome et pour décrire les mouvements utilisés dans la solution du cube Rubik. Les entiers et leur addition forment un groupe. L'identité est 0 et l'inverse de x est $-x$. Ce groupe est infini, alors que dans ce problème, nous ne traiterons que de groupes finis.

Les entiers modulo 10, avec l'opération d'addition, forment un groupe simple. Ce groupe contient les entiers $0, 1, \ldots, 9$ et l'opération consiste en une addition pour laquelle on ne garde que le chiffre des unités dans la réponse. L'identité est 0. Ce groupe satisfait aussi à la propriété suivante : $x \times y = y \times x$. Ce n'est pas le cas de tous les groupes. Par exemple, le groupe suivant n'y satisfait pas. Il est formé des éléments a, b, c, d, e et i. L'opération est définie par la table de multiplication suivante. On remarque que le groupe satisfait aux trois propriétés de base, soit l'associativité et l'existence d'une identité et d'un inverse. Or $c \times d \neq d \times c$, car $c \times d = a$ et $d \times c = b$.

	i	a	b	c	d	e
i	i	a	b	c	d	e
a	a	i	d	e	b	c
b	b	e	i	d	c	a
c	c	d	e	i	a	b
d	d	c	a	b	e	i
e	e	b	c	a	i	d

Vous devez écrire un programme qui lira une suite de tables de multiplications et qui déterminera si la structure est un groupe.

Précisions par rapport aux entrées

L'entrée présente un nombre de scénarios d'essais. Chaque scénario d'essai commence par un entier n, ($0 \leq n \leq 100$). Si le scénario d'essai commence par $n = 0$, le programme doit terminer. Pour simplifier l'entrée, nous présenterons les entiers $1, \ldots, n$ pour représenter les n éléments de la structure à évaluer ; l'identité peut être n'importe quel de ces éléments, pas nécessairement l'élément 1. À la suite de cette ligne, il y a n lignes d'entrées contenant chacune n entiers de l'intervalle $[1, \ldots, n]$. Le $q^{\text{ième}}$ entier de la $p^{\text{ième}}$ ligne de cette suite représente la valeur de $p \times q$.

Précisions par rapport aux sorties

Si la structure est un groupe, imprimez `oui` (sur sa propre ligne) ; autrement, imprimez `non` (sur sa propre ligne). Vous ne devez rien imprimer pour le scénario d'essai où $n = 0$.

Exemple d'entrée

```
2
1 2
2 1
6
1 2 3 4 5 6
2 1 5 6 3 4
3 6 1 5 4 2
4 5 6 1 2 3
5 4 2 3 6 1
6 3 4 2 1 5
7
1 2 3 4 5 6 7
2 1 1 1 1 1 1
3 1 1 1 1 1 1
4 1 1 1 1 1 1
5 1 1 1 1 1 1
6 1 1 1 1 1 1
7 1 1 1 1 1 1
3
1 2 3
3 1 2
3 1 2
0
```

Sortie pour l'exemple

```
oui
oui
non
non
```

Explications par rapport à la sortie

Les deux premières collections d'éléments sont des groupes, car elles satisfont aux trois propriétés. La troisième collection ne forme pas un groupe, car $3 \times (2 \times 2) = 3 \times 1 = 3$, mais $(3 \times 2) \times 2 = 1 \times 2 = 2$. Dans la quatrième collection, il n'y a aucune identité. En effet, 1 n'est pas une identité, car $2 \times 1 \neq 2$; 2 n'est pas une identité, car $2 \times 1 \neq 1$; 3 n'est pas une identité, car $1 \times 3 \neq 1$.

Problème S5 : L'origine de la vie

Description du problème

Le *jeu de la vie* de John H. Conway n'est pas un jeu. Il s'agit plutôt d'un *automate cellulaire* — un ensemble de règles qui décrivent des interactions entre les cellules adjacentes d'une grille. Dans notre jeu, il y a une grille rectangulaire de n sur m cellules identifiées par des coordonnées (x, y), x et y étant des entiers.

Le jeu progresse d'étape en étape ; à chaque étape, une nouvelle *génération* est calculée à partir de la *génération* actuelle. Le jeu commence par la *génération initiale*. Dans n'importe quelle génération, que nous appellerons la génération actuelle, chaque cellule est *vivante* ou *morte*. Dans la génération suivante, le statut d'une cellule peut changer, selon le statut de ses voisins dans la génération actuelle. Deux cellules distinctes, (x_1, y_1) et (x_2, y_2), sont des voisins si elles sont adjacentes à l'horizontale, à la verticale ou en diagonale, c'est-à-dire si $|x_1 - x_2| \leq 1$ et $|y_1 - y_2| \leq 1$. Une cellule qui n'est pas sur la frontière de la grille rectangulaire a donc huit voisins.

Trois paramètres (a, b, c), a, b et c étant des entiers, ont un effet sur le jeu. Voici les règles du jeu :

- Si une cellule vivante a moins de a voisins vivants dans la génération actuelle, elle meurt d'ennui et elle sera donc morte dans la génération suivante.
- Si une cellule vivante a plus de b voisins vivants dans la génération actuelle, elle meurt de surpopulation et elle sera donc morte dans la génération suivante.
- Si une cellule morte a plus de c voisins vivants dans la génération actuelle, elle revient à la vie et sera vivante dans la génération suivante.
- Dans les autres cas, le statut d'une cellule ne change pas de la génération actuelle à la génération suivante.

Les règles de passage d'une génération à la suivante sont appliquées indéfiniment. Il se peut qu'une génération soit répétée. Dans ce cas, la vie continue à l'infini. Il se peut aussi que toutes les cellules meurent. De plus, si on explore des générations antérieures qui auraient pu aboutir à la génération actuelle, il se peut que l'on découvre une génération qui doit être une première génération ; il s'agirait d'une génération qui n'aurait pas pu être créée à partir d'une génération précédente selon les règles. Une telle génération est appelée un *jardin d'Éden*.

Étant donné les paramètres du jeu, ainsi que la génération actuelle, votre tâche est de déterminer si le jeu aurait pu commencer par un jardin d'Éden. Le cas échéant, il faut indiquer le nombre d'étapes qu'il a fallu pour partir du jardin d'Éden et d'en arriver à la génération actuelle. Si plusieurs réponses sont possibles, il faut donner la plus petite. Si la génération actuelle ne provient pas d'un jardin d'Éden, imprimer -1 comme sortie.

Précisions par rapport aux entrées

Il y a un total de $m + 1$ lignes d'entrée. La première ligne contient les paramètres, soit les entiers m, n, a, b, c séparés par une espace. Voici les contraintes : $1 \leq m \leq 4$, $1 \leq n \leq 5$, $1 \leq a < b \leq 8$, $1 \leq c \leq 8$. Chacune des m lignes suivantes contient une chaîne de n caractères représentant une ligne de la génération actuelle sur la grille. Dans une chaîne, l'astérisque (“`*`”) indique une cellule *vivante*, tandis qu'un point (“`.`”) indique une cellule *morte*.

Exemple d'entrée

```
4 5 2 3 2
.****
.****
.****
.****
```

Sortie pour l'exemple

```
2
```

Explications par rapport à la sortie

Supposons que l'entrée est la génération actuelle. Une génération précédente pourrait être :

```
**.**
..*.*
....*
*****
```

La génération ci-dessus pourrait provenir de :

```
.****
**.*.
*****
*..*.
```

Cette génération ne peut provenir d'une autre génération selon les règles. De plus, il n'existe aucune série plus courte de générations qui aboutissent à la génération de l'entrée. La génération actuelle de l'entrée provient donc d'un jardin d'Eden après 2 étapes.