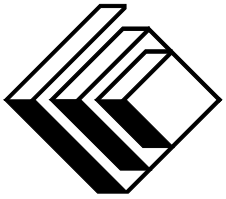


Le Centre d'éducation en mathématiques et en informatique

Concours canadien d'informatique 2008

Niveau intermédiaire

Commanditaire :

Concours canadien d'informatique

Règles et conseils à l'intention des participantes et des participants

1. Vous pouvez participer à un concours seulement. Pour participer au concours de niveau supérieur, 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 **intermédiaire**.

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

4. Vous pouvez prendre pour acquis que :
   – toutes les entrées se font par le biais du clavier ;
   – 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, `j1.pas` ou `j1.c` ou `j1.java` (ou tout autre suffixe de fichier approprié) pour le problème J1. 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. Pour certains problèmes, 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 du niveau intermédiaire 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. 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 J1 : Indice de masse corporelle

Description du problème

L'indice de masse corporelle (IMC) est un outil que les médecins utilisent pour évaluer la santé d'un adulte. Le médecin mesure la taille du patient (en mètres) et son poids (en kilogrammes). Il utilise ensuite la formule suivante pour calculer l'IMC :

$$\text{IMC} = \frac{\text{poids}}{\text{taille} \times \text{taille}}$$

Écrivez un programme qui sollicitera la taille et le poids d'un patient, calculera son IMC, puis affichera le message qui correspond au tableau suivant.

IMC	Message
Supérieur à 25	Pèse trop
De 18,5 à 25,0	Poids normal
Inférieur à 18,5	Ne pèse pas assez

Exemple de sollicitation et d'entrée 1 (entrée en *italique*)

```
Entrez le poids : 69
entrez la taille : 1.73
```

Sortie pour l'exemple 1

```
Poids normal
```

Explication de la sortie pour l'exemple 1

L'IMC est égal à $69/(1,73 \times 1,73)$, ce qui est égal à environ 23,0545. D'après le tableau, il s'agit d'un poids normal.

Exemple de sollicitation et d'entrée 2 (entrée en *italique*)

```
Entrez le poids : 84.5
Entrez la taille : 1.8
```

Sortie pour l'exemple 2

```
Pèse trop
```

Explication de la sortie pour l'exemple 2

L'IMC est égal à $84,5/(1,8 \times 1,8)$, ce qui est égal à environ 26,0802. D'après le tableau, la personne pèse trop.

Problème J2 : Dans l'ordre

Description du problème

Ces petits baladeurs numériques sont de petits ordinateurs qui gèrent et jouent des fichiers son. Le *baladeur CCI* (MPC^{2}I) est présentement en phase de développement, mais il sera bientôt en vente dans tous les bons magasins ! Dans ce problème, vous devez simuler un MPC^{2}I.

Le baladeur MPC^{2}I pourra garder 5 chansons en mémoire dont les titres seront toujours A, B, C, D et E. Le baladeur MPC^{2}I gère aussi une *liste d'écoute*, qui est un ordre particulier des chansons. Le baladeur MPC^{2}I a 4 boutons sur lesquels l'utilisateur ou l'utilisatrice appuie pour changer l'ordre de la liste d'écoute et pour jouer les chansons.

Au départ, la liste d'écoute est toujours « A, B, C, D, E ». Les 4 boutons fonctionnent comme suit :

- Bouton 1 : Place la première chanson de la liste en dernière position.
  Par exemple, « A, B, C, D, E » devient « B, C, D, E, A ».
- Bouton 2 : Place la dernière chanson de la liste en première position.
  Par exemple, « A, B, C, D, E » devient « E, A, B, C, D ».
- Bouton 3 : Alterne l'ordre des deux premières chansons de la liste.
  Par exemple, « A, B, C, D, E » devient « B, A, C, D, E ».
- Bouton 4 : Cesse de replacer les chansons et les joue selon la liste d'écoute.

Vous devez écrire un programme qui simulera un baladeur CCI. Votre programme doit solliciter sans cesse deux entiers strictement positifs, b et n. Le nombre b représente le numéro du bouton sur lequel l'utilisateur veut appuyer ($1 \leq b \leq 4$) et n et représente le nombre de fois que l'utilisateur veut appuyer sur le bouton b. Vous pouvez supposer que n vérifie toujours $1 \leq n \leq 10$.

Les entrées se termineront toujours par les nombres ($b = 4$, $n = 1$). À ce moment, le programme doit imprimer l'ordre des chansons dans la liste d'écoute et le programme doit se terminer. Vous pouvez supposer que l'utilisateur n'appuiera sur le bouton 4 qu'une seule fois.

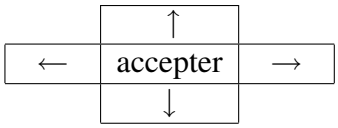
Exemple de sollicitations et d'entrées (entrées en *italique*)	**Explications**
	(La liste d'écoute initiale est « A, B, C, D, E »)
`Numéro du bouton :` *2*	
`Nombre de fois :` *1*	($b = 2$, $n = 1$; donc « A, B, C, D, E » devient « E, A, B, C, D »)
`Numéro du bouton :` *3*	
`Nombre de fois :` *1*	($b = 3$, $n = 1$; donc « E, A, B, C, D » devient « A, E, B, C, D »)
`Numéro du bouton :` *2*	
`Nombre de fois :` *3*	($b = 2$, $n = 3$; donc « A, E, B, C, D » devient « B, C, D, A, E »)
`Numéro du bouton :` *4*	
`Nombre de fois :` *1*	($b = 4$, $n = 1$) Avec $b = 4$, vous devez imprimer la liste d'écoute.

Sortie pour l'exemple

```
B C D A E
```

Problème J3 : Système GPS

Description du problème
Pour son anniversaire de naissance, Sylvie a reçu un appareil récepteur GPS (système de positionnement mondial) qu'elle utilise pour faire le tracking des sentiers de randonnée de la région. Le long de ses trajets, elle inscrit dans son appareil GPS des points de cheminement qu'elle transcrira sur une carte après son retour à la maison. Or, son appareil n'est pas muni d'un clavier. Il est muni d'un pavé de quatre touches qui déplacent un curseur vers la gauche, vers le haut, vers la droite et vers le bas, ainsi qu'une touche pour accepter la lettre choisie. Le pavé de boutons curseurs ressemble à ce qui suit :



L'écran affiche un tableau de lettres et de symboles qui servent à inscrire les descriptions. Voici un aperçu de la grille :

A	B	C	D	E	F
G	H	I	J	K	L
M	N	O	P	Q	R
S	T	U	V	W	X
Y	Z	espace	-	.	entrée

Lorsqu'on veut inscrire le nom d'un point de cheminement, le curseur se place d'abord sur la lettre A. Il faut déplacer le curseur à l'emplacement de la lettre ou du symbole voulu et le faire accepter en appuyant sur la touche « accepter ». Le curseur peut se déplacer à la verticale ou à l'horizontale, une case à la fois. Il ne peut se déplacer en diagonale. Lorsque toutes les lettres d'une description ont été acceptées, il faut placer le curseur sur la case « entrée » et actionner la touche « accepter », ce qui fait accepter l'expression au complet.

Vous devez écrire un programme qui calcule le nombre de mouvements du curseur qui sont nécessaires pour inscrire une expression. Par exemple, pour écrire l'expression « GPS », à partir de la position initiale A, il faut faire bouger le curseur d'une position vers le bas pour choisir le G, le faire bouger de 3 positions vers la droite et 1 position vers le bas pour choisir le P, le faire bouger de 1 vers le bas et 3 vers la gauche pour choisir le S, puis le faire bouger de 1 vers le bas et 5 vers la droite pour choisir la case « entrée ». Le curseur a fait un total de 15 mouvements. On remarque que le nombre de mouvements du curseur ne change pas si on le fait bouger horizontalement d'abord, puis verticalement, ou verticalement d'abord, puis horizontalement. De plus, il n'est pas permis de faire bouger le curseur en bouclage, c'est-à-dire en le faisant déborder de la grille pour qu'il paraisse de l'autre côté.

Précisions par rapport aux entrées
L'entrée consistera en une chaîne d'au plus 40 caractères. Vous pouvez supposer que tous les caractères de la chaîne paraissent dans la grille.

Précisions par rapport aux sorties
La sortie sera un entier qui représente le nombre total de mouvements du curseur qu'il faut pour inscrire la chaîne de caractères selon la grille donnée.

Exemple d'entrée 1
```
GPS
```

Sortie pour l'exemple 1
```
15
```

Exemple d'entrée 2
```
BOUT DU LAC
```

Sortie pour l'exemple 2
```
32
```

Problème J4 : De préfixée à suffixée

Description du problème

La *notation préfixée* est une notation non conventionnelle pour des expressions arithmétiques. Avec la notation conventionnelle pour des expressions arithmétiques, appelée *notation infixée*, on place un opérateur binaire entre les opérandes (p. ex., $3 + 4$), alors qu'avec la notation préfixée, l'opérateur est placé avant les opérandes (p. ex., $+3\ 4$). De même, la notation préfixée pour l'expression $5-2$ est $-5\ 2$. La notation préfixée a pour avantage de ne pas avoir besoin de parenthèses. Par exemple, l'expression $5-(4-2)$ a pour notation préfixée $-5-4\ 2$; l'expression $(5-4)-2$ a pour notation préfixée $--5\ 4\ 2$. La notation préfixée est aussi connue sous le vocable de *notation polonaise*, à cause de Jan Łukasiewicz, un logicien polonais qui l'a créée vers les 1920.

De même, dans la *notation suffixée*, ou *notation polonaise inversée*, l'opérateur est placé après les opérandes. Par exemple, l'expression $(5-4)-2$, présentée en notation infixée, a pour notation suffixée $5\ 4-2-$.

Vous devez écrire un programme qui traduit une expression de la notation préfixée à la notation suffixée.

Précisions par rapport aux entrées

Chaque ligne contient une expression arithmétique en notation préfixée. Les opérateurs sont + et − et les nombres sont tous des entiers non négatifs d'un chiffre. Les opérateurs et les nombres sont séparés par une seule espace et il n'y a aucune espace en début de ligne. La dernière entrée est indiquée par un 0 seul sur une ligne. Vous pouvez supposer que chaque ligne d'entrée contient une expression valide de moins de 20 opérateurs en notation préfixée.

Précisions par rapport aux sorties

Vous devez traduire chaque expression en notation suffixée et la produire sur une ligne séparée. Les nombres et les opérateurs sont séparés par au moins une espace. Le 0 final n'est pas traduit.

Exemples d'entrées et de sorties

Exemple d'entrée	Exemple de sortie
`1`	`1`
`+ 1 2`	`1 2 +`
`- 2 2`	`2 2 -`
`+ 2 - 2 1`	`2 2 1 - +`
`- - 3 + 2 1 9`	`3 2 1 + - 9 -`
`0`	

Problème J5 : Réaction

Description du problème
Les deux plus grands physiciens nucléaires canadiens, Patrick et Roland, viennent de compléter la construction du premier réacteur de fission nucléaire au monde. Ils doivent maintenant s'asseoir et faire fonctionner le réacteur toute la journée, chaque jour. Naturellement, après quelque temps ils se sont ennuyés. Pour se distraire, ils ont appris à contrôler les réactions individuelles à l'intérieur du réacteur et ils ont inventé un jeu appelé Réaction.

Dans le jeu Réaction, un certain nombre de particules sont placées dans le réacteur au départ. Les joueurs jouent à tour de rôle, mais Patrick joue toujours le premier. Lorsque c'est à son tour de jouer, un joueur doit choisir un nombre des particules qui restent de manière à former une des réactions possibles. Ces particules sont alors détruites. Éventuellement, il reste tellement peu de particules qu'il devient impossible de former une autre réaction nucléaire ; le joueur dont c'est le tour et qui ne peut jouer perd alors la partie.

Dans notre univers, vous pouvez supposer qu'il n'y a que quatre sortes de particules, soit A, B, C et D. Chaque réaction correspond à une liste de particules qui peuvent être détruites dans un même tour. Voici les cinq réations possibles :

1. AABDD
2. ABCD
3. CCD
4. BBB
5. AD

Par exemple, la première réaction, soit « AABDD », indique qu'il est possible de détruire deux particules A, une particule B et deux particules D dans un même tour.

Il ressort que peu importe le nombre et la sorte de particules qui sont déposées dans le réacteur, exactement l'un des deux joueurs peut avoir une *stratégie gagnante parfaite*. On dit que le *joueur X a une stratégie gagnante parfaite* si, peu importe les réactions choisies par son adversaire, X peut toujours gagner en choisissant ses réactions avec soin. Par exemple, si on a placé une particule A, cinq particules B et trois particules D dans le réacteur, alors Roland a une stratégie gagnante parfaite, soit : « Si Patrick choisit la réaction BBB au départ, Roland choisit la réaction AD en réponse ; si Patrick choisit la réaction AD au départ, Roland choisit la réaction BBB en réponse. » (Cette stratégie est gagnante parce que dans les deux cas, Patrick ne peut jouer à son second tour, puisqu'il ne reste pas assez de particules pour former une des cinq réactions possibles.)

Étant donné un nombre de particules de chaque sorte qui sont déposées dans le réacteur, pouvez-vous déterminer lequel des deux a une stratégie gagnante parfaite ?

Précisions par rapport aux données
La première ligne d'entrée contient le nombre n ($1 \leq n < 100$), soit le nombre de scénarios d'essais. Chaque scénario d'essai est formé de 4 entiers séparés par une espace, tous sur une ligne ; ils représentent le nombre initial respectif de particules A, B, C et D qui sont déposées dans le réacteur. Vous pouvez supposer que chaque nombre initial de particules est un entier de 0 à 8.

Précisions par rapport aux sorties
Pour chaque scénario d'essai, imprimez le nom du joueur qui a une stratégie gagnante parfaite, soit Patrick ou Roland.

Exemple d'entrées

```
6
0 2 0 2
1 3 1 3
1 5 0 3
3 3 3 3
8 8 6 7
8 8 8 8
```

Sorties pour l'exemple

```
Roland
Patrick
Roland
Roland
Roland
Patrick
```

Explication partielle des sorties pour l'exemple
Dans le premier scénario, Roland gagne, puisque Patrick ne peut former *aucune* réaction. (La stratégie de Roland, c'est de ne rien faire.)

Dans le deuxième scénario, Patrick a une stratégie gagnante, soit « former la réaction ABCD », qui a pour effet de faire perdre Roland à son premier tour.

L'issue du troisième scénario est expliquée dans l'énoncé du problème.