Concours Informatique Luxembourgeois 2012
Epreuve de Demi-Finale (22/3/2012)

Tâche I - Images Panino 30 points

Tout le monde connaît les images ou stickers Panino à collectionner : les joueurs de la coupe du monde de football, les personnages de Walt Disney, etc. Chacune de ces images porte un numéro unique par collection permettant au collectionneur de s'y retrouver, de les classer, etc.

Tâche

Soit deux collectionneurs *A* et *B* disposant respectivement de *M* et *N* images de la même collection. Écrivez un programme qui détermine le nombre minimal d'images qu'il faut enlever aux deux collections pour que les deux collections restantes de *A* et *B* soient composées exactement des mêmes images.

Exemple

```
Collection A :   4 2 3 5 2 1 1 6
Collection B :   2 2 3 3 4 1 2
```

Si on enlève à la collection *A* une image portant le numéro 1, les images 5 et 6 ainsi qu'à la collection *B* une image portant le numéro 2 et une image portant le numéro 3, alors chacun des deux collectionneurs disposera de la même collection.

```
Collection A :   4 2 3 ~~5~~ 2 ~~1~~ 1 ~~6~~
Collection B :   ~~2~~ 2 ~~3~~ 3 4 1 2
```

Restrictions

1 ≤ **M** ≤ 5000	Le nombre maximal d'images de la collection de *A*.
1 ≤ **N** ≤ 5000	Le nombre maximal d'images de la collection de *B*.
1 ≤ **I** ≤ 300	Le numéro de l'image.

Entrée et sortie du programme

Entrée

- Le programme lit d'abord à partir de la console le nom du fichier contenant les numéros des images des deux collections.
- Ce fichier contient les numéros des images des deux collections. Chacune des deux collections est composée d'une liste non nulle et non triée de nombres entiers contenant au maximum 5000 images. Chaque image est représentée par un numéro entier compris entre 1 et 300. La fin de la liste est indiquée par la valeur -1. Deux entiers consécutifs sont séparés par un seul espace. Un numéro d'image peut apparaître plusieurs fois à des endroits quelconques de la liste ; tous les numéros d'images entre 1 et 300 ne sont pas nécessairement présents.

Sortie

- Le programme doit afficher à la console le nombre minimal d'images qu'il faut enlever des deux collections pour que les deux collections soient composées exactement des mêmes images.

Exemple d'exécution

Contenu du fichier IN.TXT :

```
4 2 3 5 2 1 1 6 -1
2 2 3 3 4 1 2 -1
```

Saisie à la console :

```
IN.TXT
```

Affichage à la console :

```
5
```

Remettez le programme sous le nom PANINO.xxx, avec xxx=PAS ou C(PP). Remettez également le fichier binaire exécutable PANINO.EXE correspondant au programme.

Tâche II - Wild Wild Cards 40 points

Un *wild card* (en anglais **wild card**, ou **joker**) est un symbole générique utilisé en informatique pour la recherche d'un mot ou d'une expression incomplète.

Généralement utilisés dans les lignes de commandes, les wild cards permettent d'exprimer des noms de fichiers qui possèdent une ou plusieurs parties communes.

Les deux wild cards les plus connus sont :

- le point interrogation « **?** » qui doit être remplacé par **exactement un** caractère
- l'astérisque « ***** » qui peut être remplacé par **un nombre quelconque** de caractères (donc 0 ou plusieurs caractères)

Tâche

Écrivez un programme qui permet d'identifier tous les mots d'une liste qui correspondent à une expression contenant zéro ou plusieurs wild cards. Le programme permet d'abord d'entrer au clavier une expression **E** contenant zéro ou plusieurs wild cards. Ensuite le programme lit à partir du fichier IN.TXT une liste composé de **N** mots et écrit dans le fichier OUT.TXT tous les mots qui correspondent à l'expression entrée auparavant.

Restrictions

A = {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}

L'expression **E** est formée exclusivement des lettres minuscules de l'alphabet **A** ainsi que des symboles « **?** » et « ***** ».

$1 \leq \mathbf{N} \leq 10\,000$ Le nombre maximal de mots dans le fichier IN.TXT

Vous n'avez pas le droit d'utiliser les fonctionnalités des expressions régulières offertes par vos langages de programmation.

Entrée et sortie du programme

Entrée

- Le programme lit d'abord à partir de la console l'expression **E** qui sert à identifier les mots par la suite. L'utilisateur veille à entrer une expression valide, c.-à-d. votre programme n'a pas besoin de vérifier l'exactitude de cette expression.
- La première ligne du fichier texte IN.TXT contient un seul nombre entier positif : le nombre de mots **N**.
- Les **N** lignes suivantes contiennent les différents mots, un mot par ligne. Les mots sont tous écrits en minuscules et sont formés exclusivement des lettres de l'alphabet **A**.

Sortie

- Le programme doit créer un fichier texte OUT.TXT contenant un maximum de **N** lignes.
- Chaque ligne contient exactement un mot issu de la liste des mots IN.TXT et qui correspond à l'expression **E**.
- Dans le cas où aucun mot ne correspond à l'expression **E**, le fichier doit être vide.

Exemple d'exécution

Contenu du fichier IN.TXT **:**

```
5
take
take1
taken
take12
tak123
```

Saisie à la console :

```
take?
```

Contenu du fichier OUT.TXT :

```
take1
taken
```

Autres exemples

Expression	Mot	Correspondance
*ake	cheesecake	oui
t	alphabet	oui
*t?	alphabet	non
?????	cat	non
???	dog	oui
a*bc	aabbccbc	oui
?bc?	abcd	oui
colo?r	color	non
colo?r	colour	oui

Remettez le programme sous le nom WILDCARDS.xxx, avec xxx=PAS ou C(PP). Remettez également le fichier binaire exécutable WILDCARDS.EXE correspondant au programme.

Tâche III – Horaire 30 points

Chaque élève l'utilise régulièrement : l'horaire qui lui montre les jours, les plages horaires et ses cours. Voici un horaire typique tel qu'il est utilisé dans beaucoup d'écoles allemandes sous forme d'un tableau :

<table>
<tr><th>Uhrzeit</th><th>Montag</th><th>Dienstag</th><th>Mittwoch</th><th>Donnerstag</th><th>Freitag</th></tr>
<tr><td>8:00</td><td>Deutsch</td><td>Französisch</td><td>Biologie</td><td>Religion</td><td rowspan="2">Mathe</td></tr>
<tr><td>9:00</td><td>Englisch</td><td>Musik</td><td>Geschichte</td><td>Physik</td></tr>
<tr><td>10:00</td><td rowspan="2">Sport</td><td>Geschichte</td><td>Mathe</td><td rowspan="2">Französisch</td><td>Englisch</td></tr>
<tr><td>11:00</td><td>Mathe</td><td>Englisch</td><td>Chemie</td></tr>
<tr><td>12:00</td><td>Religion</td><td>Physik</td><td>Erdkunde</td><td>Erdkunde</td><td>Musik</td></tr>
<tr><td>13:00</td><td>Sozialkunde</td><td>Chemie</td><td>Deutsch</td><td>Deutsch</td><td>Bio</td></tr>
</table>

En HTML (langage qui décrit la structure d'une page web), un tel tableau n'est cependant pas défini colonne par colonne mais ligne par ligne. De plus, HTML offre la possibilité de « fusionner » plusieurs cellules consécutives ensemble comme par exemple les cours « Sport », « Französisch » et « Mathe ».

Tâche

Écrivez un programme qui permet d'interroger un horaire afin d'informer l'utilisateur du cours défini à la plage horaire précisée. Le programme doit d'abord lire l'horaire depuis le fichier IN.TXT et répondre ensuite correctement à une ou plusieurs requêtes. Une requête est toujours de la forme « **jour heure** ». La réponse correcte à cette requête est le nom du cours que l'élève suit à cette heure pendant le jour précisé.

Restrictions

jour = {lundi, mardi, mercredi, jeudi, vendredi}

heure = {8, 9, 10, 11, 12, 13, 14, 15, 16}

- L'horaire a exactement 5 jours.
- Chaque jour a exactement 9 heures.
- Il n'y pas de trou dans la définition de l'horaire.

Entrée et sortie du programme

- Le fichier texte IN.TXT contient une séquence de cours. Chaque cours nécessite deux lignes :
  - la première ligne précise le nom du cours et peut contenir n'importe quelle chaîne de caractères ;
  - la deuxième ligne précise la durée du cours et est un nombre entier positif.
- L'horaire du fichier IN.TXT est défini ligne par ligne : le premier cours est donc lundi à 8 heures, le deuxième cours est mardi à 8 heures ...
- Ensuite votre programme est interactif : l'utilisateur peut entrer indéfiniment une requête de la forme « **jour heure** » et le programme doit répondre correctement à cette requête en affichant à l'écran le nom du cours comme il est défini dans le fichier IN.TXT. L'utilisateur veille à respecter le bon format pour chaque requête. Au lieu d'une requête, l'utilisateur peut entrer « **-1** » pour terminer le programme.

Exemples de définition d'horaires

Pour des raisons d'espace, les horaires suivants contiennent uniquement deux jours avec 2 heures par jour. **Les horaires de votre programme contiennent tous 5 jours avec 9 heures par jour !**

Exemple 1

Horaire :

MATHE	LATIN
ALLEM	FRANC

Contenu du fichier IN.TXT :

```
MATHE
1
LATIN
1
ALLEM
1
FRANC
1
```

Exemple 2

Horaire :

MATHE	LATIN
	FRANC

Contenu du fichier IN.TXT :

```
MATHE
2
LATIN
1
FRANC
1
```

Exemple 3

Horaire :

MATHE	LATIN
FRANC	

Contenu du fichier IN.TXT :

```
MATHE
1
LATIN
2
FRANC
1
```

Exemple 4

Horaire :

MATHE	LATIN

Contenu du fichier IN.TXT :

```
MATHE
2
LATIN
2
```

Exemple d'exécution

Pour des raisons d'espace, l'exemple d'exécution est basé sur l'horaire réduit de l'exemple 2 (illustré encore une fois ci-dessous). Supposons que l'horaire est composé des deux jours *lundi* et *mardi* et que les cours ont lieu à 8 heures et à 9 heures.

Contenu du fichier IN.TXT :

```
MATHE
2
LATIN
1
FRANC
1
```

Exemple d'exécution :

```
lundi 8
MATHE
mardi 8
LATIN
mardi 9
FRANC
lundi 9
MATHE
-1
```

Remettez le programme sous le nom HORAIRE.xxx, avec xxx=PAS ou C(PP). Remettez également le fichier binaire exécutable HORAIRE.EXE correspondant au programme.