Lëtzebuerger Informatiksolympiad 2016

Hallef Finale

Tous les programmes doivent être réalisés sous forme d'applications console.

Les formats des données ainsi que des résultats sont à respecter.

TÂCHE 1 — BINAIRE

Description

Max a envie de s'investir dans le hacking et le reverse engineering. Pour y arriver, il doit entre autres, apprendre à maîtriser la représentation de données en binaire. Pour l'aider, vous lui proposez deux travaux. Le premier travail consiste tout simplement à convertir un entier positif non nul en binaire. Le deuxième travail est plus spécial. À partir d'un nombre entier positif non nul N, il faut produire deux nouveaux nombres entiers X et Y créés de la manière suivante : soit (p_1, p_2, … p_k) les positions des bits à 1 dans N, le bit de plus faible poids étant en position 0.

Alors l'entier X en binaire aura des 1 en (p_1, p_3, p_5, …) et des 0 autrement. L'entier Y en binaire aura des 1 en (p_2, p_4, p_6, …) et des 0 autrement.

Exemple : le nombre 13 a comme représentation binaire 1101.

Les positions des bits à 1 sont (p_1, p_2, p_3) = (0, 2, 3). X s'écrit alors en binaire 1001 comme les 1 seront en position (p_1, p_3), c'est-à-dire aux positions 0 et 3. Y s'écrit 100 en binaire puisqu'il ne contient qu'un bit à 1 à savoir (p_2) en position 2. En décimal X et Y s'écrivent respectivement 9 et 4.

Entrée et sortie du programme

Entrée

- N, un entier positif ou nul avec $N \leq 4294967295 = (2^{32} - 1)$. Si $N = 0$, l'exécution du programme s'arrête.

Sortie

- Pour chaque nombre $N \neq 0$ le programme affiche deux lignes. La première contient la représentation binaire de N. La deuxième ligne contient les entiers X et Y dans leur représentation décimale.

Exemples d'exécution

Entrée

```
13
121
0
```

Sortie

```
1101
9 4
1111001
81 40
```

Remettez le programme sous le nom BINAIRE.xxx, avec xxx=PAS ou C(PP) ou JAVA.

TÂCHE 2 — MOT DE PASSE

Description

Max a trouvé une façon originale de créer un mot de passe. À partir d'un mot clé qu'il connaît très bien et qu'il ne risque pas d'oublier, il choisit comme mot de passe la permutation de toutes les lettres du mot clé qui maximise le total des distances absolues entre toutes les paires de deux lettres consécutives du mot. Ce mot de passe est complété en concaténant un entier représentant cette distance. Si plusieurs mots de passe possibles ont la même distance maximale, il faut choisir le premier mot dans l'ordre lexicographique de toutes les lettres constituant le mot.
La distance entre deux caractères est définie comme étant la valeur absolue de la différence de leurs codes ASCII. Exemple : distance('a', 'a') = 0 ; distance('a', 'b') = 1 ; distance('a', 'z') = 25 .

Entrée et sortie du programme

Entrée

- Une chaîne de caractères de taille non nulle inférieure ou égale à 10 et ne contenant pas d'espace. Les caractères permis sont les lettres de 'a' à 'z' et 'A' à 'Z'.

Sortie

- Le programme affiche le mot de passe proposé.

Exemple d'exécution

Entrée

```
ouf
```

Sortie

```
ofu24
```

La liste des permutations possibles est dans l'ordre (fou, fuo, ofu, ouf, ufo, uof).
dist(fou) = dist(fo) + dist(ou) = 9 + 6 = 15
dist(fuo) = dist(fu) + dist(uo) = 15 + 6 = 21
dist(**ofu**) = dist(of) + dist(fu) = 9 + 15 = **24** c'est la distance maximale
dist(ouf) = dist(ou) + dist(uf) = 6 + 15 = 21
dist(**ufo**) = dist(uf) + dist(fo) = 15 + 9 = **24** c'est la distance maximale
dist(uof) = dist(uo) + dist(of) = 6 + 9 = 15

Remettez le programme sous le nom PASSWD.xxx, avec xxx=PAS ou C(PP) ou JAVA.

TÂCHE 3 — FILE D'ATTENTE

Description

Lorsqu'il ne s'adonne pas à l'informatique, Max aime passer son temps dans son parc d'attractions favori. Or ce dernier est complètement submergé par les nombreux élèves qui y passent leurs loisirs. Pour maîtriser la situation, le parc gère une file d'attente centrale. Vous avez remarqué que la file d'attente fonctionne d'une façon bien particulière et vous voudriez aider Max à en tirer profit pour pouvoir accéder aux attractions qu'il préfère.
Dans la file d'attente il y a N personnes. Dans le parc il y a T attractions différentes et exactement B jetons pour chaque attraction. La première personne dans la file reçoit un jeton pour la première attraction, la deuxième personne un jeton pour la deuxième attraction, etc. Après avoir distribué un jeton pour la dernière attraction, on recommence avec la première attraction. Une fois que la dernière personne de la file a été servie, on recommence avec la première personne, qui recevra un jeton pour l'attraction suivant celle dont un jeton

vient d'être donné à la dernière personne. On continue ainsi jusqu'à ce qu'il n'y ait plus de jetons.
Sachant quelle est l'attraction que Max préfère, mais pour lui éviter de se faire remarquer, vous appliquez la stratégie suivante pour lui proposer une place à prendre dans la file d'attente :

- Si dans la file d'attente il existe plusieurs positions qui permettent d'avoir le nombre le plus élevé de jetons pour l'attraction que Max préfère, alors vous lui recommandez celle qui est la plus proche de la fin de la file, puisqu'on arrive plus facilement à se faufiler à cet endroit.
- S'il existe une seule position donnant droit au nombre de jetons le plus élevé, vous lui suggérez d'occuper la position qui donne droit au deuxième nombre le plus élevé de jetons pour l'attraction que Max préfère. Mais si dans ce cas plusieurs positions sont possibles, vous lui suggérez de nouveau d'occuper celle qui est le plus proche de la fin de la file d'attente.

Input

La première ligne indique le nombre J de jeux de données.
Elle est suivie de J lignes contenant chacune quatre nombres entiers, N, T, B et S, sachant que: $1 \leq T, B, S \leq 100$. De plus, $2 \leq N \leq 100$ et $S \leq T$.
N est le nombre de personnes dans la file d'attente centrale.
T est le nombre d'attractions dans le parc.
B est le nombre de jetons pour chaque attraction.
S est le numéro de l'attraction que Max préfère.

Output

Pour chaque jeu de données le programme affiche deux lignes. La première ligne contient "Jeu *x*: ", *x* étant le numéro du jeu de données. Dans la deuxième ligne il affiche la position à laquelle Max devrait se trouver dans la file pour accéder à son attraction favorite.
Après chaque jeu de données, le programme affiche une ligne vide.

Exemple

Données	Résultats
2 4 6 3 5 4 3 6 1	Jeu 1: 3 Jeu 2: 4

Illustration complémentaire pour le premier jeu de données

4 6 3 5, c'est-à-dire :

- 4 personnes dans la file d'attente,
- 6 attractions,
- 3 jetons par attraction,
- Max préfère l'attraction 5.

Il y a 6 * 3 = 18 jetons à distribuer. Les personnes auront les jetons suivants :

Place dans la file	Jetons reçus pour les attractions (numéro, dans l'ordre de la réception)
1	1, 5, 3, 1, 5
2	2, 6, 4, 2, 6
3	3, 1, 5, 3
4	4, 2, 6, 4

La première place dans la file donne droit au nombre le plus élevé de jetons pour le jeu 5. Il n'existe pas d'autre position permettant d'obtenir deux jetons pour l'attraction 5. La place 3 donne droit à 1 jeton pour l'attraction 5. C'est le deuxième nombre le plus élevé. Vous lui recommandez la place 3.

Remettez le programme sous le nom FILE.xxx, avec xxx=PAS ou C(PP) ou JAVA.