Problem S1: Hat Circle

Problem Description

At a recent social gathering, N people sit around a circular table, where N is even. The seats are numbered clockwise from 1 to N. Each person is wearing a hat with a number on it. Specifically, the person at seat i is wearing a hat with the number H_i on it.

Each person looks at the person who is directly across (diametrically opposite) them in the circle.

Determine the number of people who see someone with a hat with the same number as their own.

Input Specification

The first line of input will consist of one even positive integer N, representing the number of people at the social gathering.

The next N lines each contain a single non-negative integer H_i, representing the hat number of person i.

The following table shows how the available 15 marks are distributed:

Marks	Description	Bounds on N	Bounds on H_i
2	Very small number of people; only two hat numbers	$N \leq 4$	$H_i \leq 1$
1	Only one hat number	$N \leq 100$	$H_i = 1$
2	People in even numbered seats have hat number 1; people in odd numbered seats have hat number 0	$N \leq 100$	$H_i \leq 1$
5	Medium number of people	$N \leq 2\,000$	$H_i \leq 4\,000$
5	Large number of people and hat numbers	$N \leq 1\,000\,000$	$H_i \leq 2\,000\,000$

Output Specification

Output a single integer representing the number of people who see their hat number on the person directly across from them.

Sample Input 1

```
4
0
1
0
1
```

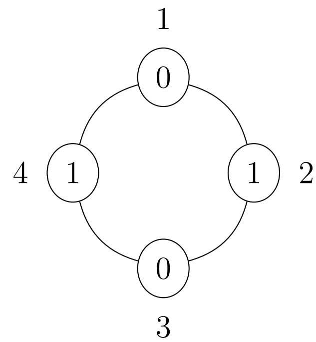
La version française figure à la suite de la version anglaise.

Output for Sample Input 1

```
4
```

Explanation of Output for Sample Input 1

The four seats around the table are shown below. Hat numbers are shown inside each seat and seat numbers are shown beside each seat. Notice that every person sees their hat number. The people in seats 1 and 3 both see hat number 0, and the people in seats 2 and 4 both see hat number 1.



Sample Input 2

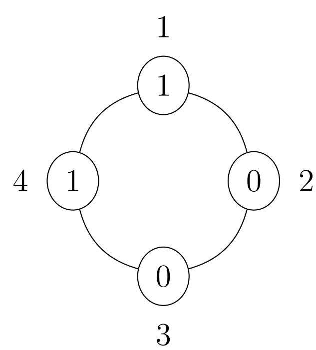
```
4
1
0
0
1
```

Output for Sample Input 2

```
0
```

Explanation of Output for Sample Input 2

The four seats around the table are shown below. Hat numbers are shown inside each seat and seat numbers are shown beside each seat. Notice that no person sees their hat number. The people in seats 1 and 4 both see hat number 0, and the people in seats 2 and 3 both see hat number 1.



La version française figure à la suite de la version anglaise.

Problème S1 : Cercle de chapeaux

Énoncé du problème

Lors d'une réunion de fraternisation, N personnes sont assises autour d'une table circulaire, N étant un entier pair strictement positif. Les sièges sont numérotés de 1 à N dans le sens des aiguilles d'une montre. Chaque personne porte un chapeau sur lequel figure un numéro. Plus précisément, la personne assise à la place i porte un chapeau sur lequel figure le numéro H_i.

Chaque personne regarde la personne qui se trouve directement en face d'elle (diamétralement opposée) dans le cercle.

Déterminer le nombre de personnes qui voient une personne avec un chapeau portant le même numéro que le leur.

Précisions par rapport aux données d'entrée

La première ligne des données d'entrée doit contenir un entier pair strictement positif N, représentant le nombre de personnes présentes à la réunion de fraternisation.

Les N lignes suivantes doivent chacune contenir un seul entier non négatif H_i, représentant le numéro qui figure sur le chapeau de la personne i.

Le tableau ci-dessous détaille la répartition des 15 points disponibles.

Points	Description	Bornes de N	Bornes de H_i
2	Un très petit nombre de personnes ; seulement deux numéros de chapeau	$N \leq 4$	$H_i \leq 1$
1	Un seul numéro de chapeau	$N \leq 100$	$H_i = 1$
2	Les personnes occupant les places paires ont le chapeau numéro 1 ; les personnes occupant les places impaires ont le chapeau numéro 0	$N \leq 100$	$H_i \leq 1$
5	Un nombre moyen de personnes	$N \leq 2\,000$	$H_i \leq 4\,000$
5	Un grand nombre de personnes et de numéros de chapeaux	$N \leq 1\,000\,000$	$H_i \leq 2\,000\,000$

Précisions par rapport aux données de sortie

Les données de sortie devraient afficher un seul entier représentant le nombre de personnes qui voient le numéro de leur chapeau sur la personne qui se trouve juste en face d'eux.

Donnés d'entrée d'un 1^er^ exemple

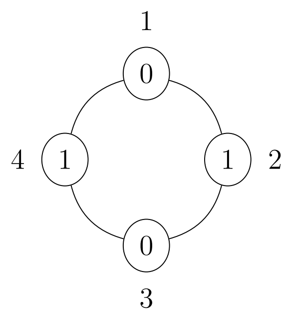
```
4
0
1
0
1
```

Donnés de sortie du 1er exemple

```
4
```

Justification des donnés de sortie du 1er exemple

Les quatre sièges autour de la table sont représentés dans la figure ci-dessous. Les numéros des chapeaux sont indiqués à l'intérieur de chaque siège tandis que les numéros des sièges sont indiqués à côté de chaque siège. Remarquons que chaque personne voit le numéro de son chapeau. Les personnes occupant les sièges 1 et 3 voient toutes deux le chapeau numéro 0 et les personnes occupant les sièges 2 et 4 voient toutes deux le chapeau numéro 1.



Donnés d'entrée d'un 2e exemple

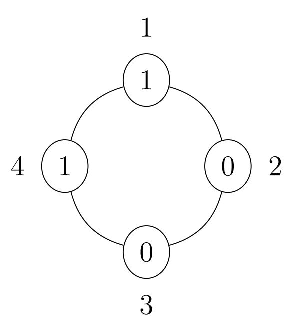
```
4
1
0
0
1
```

Donnés de sortie du 2e exemple

```
0
```

Justification des donnés de sortie du 2e exemple

Les quatre sièges autour de la table sont représentés dans la figure ci-dessous. Les numéros des chapeaux sont indiqués à l'intérieur de chaque siège tandis que les numéros des sièges sont indiqués à côté de chaque siège. Remarquons qu'aucune personne ne voit le numéro de son chapeau. Les personnes occupant les sièges 1 et 4 voient toutes deux le chapeau numéro 0 et les personnes occupant les sièges 2 et 3 voient toutes deux le chapeau numéro 1.



English version appears before the French version

Problem S2: Heavy-Light Composition

Problem Description

In a string containing only lowercase letters of the alphabet ("a" through "z"), we say a letter is *heavy* if it appears more than once in the string, and *light* otherwise.

We will be given a number of strings. For each string, we would like to determine whether the letters of the string alternate between light and heavy.

Input Specification

The first line of input will consist of two positive integers T and N, representing the number of strings and the length of each string.

The next T lines each contain a sequence of N lowercase letters of the alphabet.

The following table shows how the available 15 marks are distributed:

Marks	**Bounds on T**	**Bounds on N**	**Other Restrictions**
5	$2 \leq T \leq 4$	$2 \leq N \leq 4$	Only the letters "a" and "b" will be used
5	$2 \leq T \leq 10$	$2 \leq N \leq 30$	None
2	$2 \leq T \leq 100$	$2 \leq N \leq 100$	Only the letter "a" will be heavy; all other letters are light
3	$2 \leq T \leq 10\,000$	$2 \leq N \leq 100$	None

Output Specification

Output T lines, where each line will be either `T` or `F`. If the i-th input string does alternate between light and heavy letters, the i-th line of output should be `T`; and otherwise, the i-th line of output should be `F`.

Sample Input 1

```
3 4
abcb
bcbb
babc
```

Output for Sample Input 1

```
T
F
T
```

La version française figure à la suite de la version anglaise.

Explanation of Output for Sample Input 1

The first string is composed of a light letter, then a heavy letter, then a light letter, and then a heavy letter.

The second string ends in two consecutive heavy letters.

The third string is composed of a heavy letter, then a light letter, then a heavy letter, and then a light letter.

Sample Input 2

```
2 3
abc
bcb
```

Output for Sample Input 2

```
F
T
```

Explanation of Output for Sample Input 2

The first string is composed of all light letters.

The second string is composed of a heavy letter, then a light letter, and then a heavy letter.

La version française figure à la suite de la version anglaise.

Problème S2 : Composition lourde-légère

Énoncé du problème

Dans une chaîne ne contenant que des lettres minuscules de l'alphabet (de « a » à « z »), on dit qu'une lettre est *lourde* si elle parait plus d'une fois dans la chaîne et *légère* dans le cas contraire.

Étant donné un certain nombre de chaînes de lettres, déterminer si les lettres de chaque chaîne alternent entre légères et lourdes.

Précisions par rapport aux données d'entrée

La première ligne des données d'entrée doit contenir deux entiers strictement positifs T et N, représentant respectivement le nombre de chaînes et la longueur de chaque chaîne.

Chacune des T lignes suivantes doit contenir une chaîne de N lettres minuscules de l'alphabet.

Le tableau ci-dessous détaille la répartition des 15 points disponibles.

Points	**Bornes de T**	**Bornes de N**	**Autres restrictions**
5	$2 \leq T \leq 4$	$2 \leq N \leq 4$	Seules les lettres « a » et « b » seront utilisées
5	$2 \leq T \leq 10$	$2 \leq N \leq 30$	Aucune
2	$2 \leq T \leq 100$	$2 \leq N \leq 100$	Seule la lettre « a » sera lourde ; toutes les autres lettres sont légères
3	$2 \leq T \leq 10\ 000$	$2 \leq N \leq 100$	Aucune

Précisions par rapport aux données de sortie

Les données de sortie devraient contenir T lignes, chacune étant soit `T`, soit `F`. Si la $i^{\text{ième}}$ chaîne d'entrée alterne entre lettres légères et lettres lourdes, alors la $i^{\text{ième}}$ ligne des données de sortie devrait être `T` ; sinon, la $i^{\text{ième}}$ ligne des données de sortie devrait être `F`.

Donnés d'entrée d'un 1er exemple

```
3 4
abcb
bcbb
babc
```

Donnés de sortie du 1er exemple

```
T
F
T
```

English version appears before the French version

Justification des donnés de sortie du 1^{er} exemple
La première chaîne est composée d'une lettre légère, puis d'une lettre lourde, puis d'une lettre légère, puis d'une lettre lourde.

La deuxième chaîne se termine par deux lettres lourdes consécutives.

La troisième chaîne est composée d'une lettre lourde, puis d'une lettre légère, puis d'une lettre lourde, puis d'une lettre légère.

Donnés d'entrée d'un 2^e exemple
```
2 3
abc
bcb
```

Donnés de sortie du 2^e exemple
```
F
T
```

Justification des donnés de sortie du 2^e exemple
La première chaîne est composée uniquement de lettres légères.

La deuxième chaîne est composée d'une lettre lourde, puis d'une lettre légère, puis d'une lettre lourde.

Problem S3: Swipe

Problem Description

Swipe is a new mobile game that has recently exploded in popularity. In each level of Swipe, you are given 2 rows of integers that can be represented as arrays A and B of size N. The objective of Swipe is to beat each level by turning array A into array B.

There are two swipe operations you can perform on array A.

- Swipe right: Select the subarray $[\ell, r]$ and set $A_i = A_\ell$ for all $\ell \leq i \leq r$.
- Swipe left: Select the subarray $[\ell, r]$ and set $A_i = A_r$ for all $\ell \leq i \leq r$.

For example, starting with array $A = [0, 1, 2, 3, 4, 5]$, if we swipe right on $[2, 4]$, we would obtain the array $[0, 1, 2, 2, 2, 5]$. If instead, we started with the same array A, and swiped left on $[3, 5]$, we would obtain the array $[0, 1, 2, 5, 5, 5]$. Note that these arrays are 0-indexed.

Unfortunately, the game is bugged and contains levels that are impossible to beat. Determine if it is possible to transform array A into array B. If it is possible, determine a sequence of swipe operations that transforms array A into array B.

Input Specification

The first line of input will consist of one positive integer N, representing the length of each of the two arrays of integers.

The second line of input contains N space separated integers contained in array A.

The third line of input contains N space separated integers contained in array B.

The following table shows how the available 15 marks are distributed:

Marks	Bounds on N	Bounds on A_i and B_i
2	$N = 2$	$1 \leq A_i, B_i \leq 3$
4	$1 \leq N \leq 8$	$1 \leq A_i, B_i \leq 8$
4	$1 \leq N \leq 500$	$1 \leq A_i, B_i \leq 3000$
5	$1 \leq N \leq 300\,000$	$1 \leq A_i, B_i \leq 300\,000$

Note that for a subtask worth M marks, you will receive $\left\lfloor \frac{M}{2} \right\rfloor$ marks for a solution that only correctly outputs the first line of output.

Output Specification

The first line of output will contain `YES` if there is a sequence of swipes that can transform array A into array B; otherwise, the first line of output will contain `NO`.

La version française figure à la suite de la version anglaise.

If the first line of output is `YES`, the next line contains a non-negative integer K ($K \leq N$), indicating the number of swipes.

Each of the next K lines contain three space-separated values: D_j, ℓ_j, and r_j. The value D_j will be either `R` or `L`, indicating that the jth swipe is either a right or left swipe, respectively. The values ℓ_j and r_j indicate the left-end and right-end of the swipe where $0 \leq \ell_j \leq r_j < N$.

Sample Input 1

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

Output for Sample Input 1

```
YES
1
R 1 2
```

Sample Input 2

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

Output for Sample Input 2

```
NO
```

Sample Input 3

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

Output for Sample Input 3

```
YES
0
```

La version française figure à la suite de la version anglaise.

Problème S3 : Swipe

Énoncé du problème

Le jeu *Swipe* est un nouveau jeu mobile qui connaît un grand succès. Dans chaque niveau du jeu *Swipe*, on a 2 rangées d'entiers que l'on peut représenter à l'aide de tableaux A et B de taille N. L'objectif du jeu est de battre chaque niveau en transformant le tableau A en tableau B.

Il existe deux opérations de glissement que l'on peut effectuer sur le tableau A.

— Glissement vers la droite : On sélectionne le sous-tableau $[\ell, r]$ et on définit $A_i = A_\ell$ pour tout $\ell \leq i \leq r$.

— Glissement vers la gauche : On sélectionne le sous-tableau $[\ell, r]$ et on définit $A_i = A_r$ pour tout $\ell \leq i \leq r$.

À titre d'exemple, considérons le tableau $A = [0, 1, 2, 3, 4, 5]$. En appliquant un glissement vers la droite sur $[2, 4]$, on obtient le tableau $[0, 1, 2, 2, 2, 5]$. Cependant, si l'on part du même tableau A et que l'on applique un glissement vers la gauche sur $[3, 5]$, on obtiendrait le tableau $[0, 1, 2, 5, 5, 5]$. Remarquons que ces tableaux sont indexés à partir de 0.

Malheureusement, le jeu est buggé et contient des niveaux qui sont impossibles à battre. Déterminer s'il est possible de transformer le tableau A en tableau B. Si cela est possible, déterminer une séquence d'opérations de glissement qui permettrait de transformer le tableau A en tableau B.

Précisions par rapport aux données d'entrée

La première ligne des données d'entrée droit contenir un entier strictement positif N, représentant la longueur de chacun des deux tableaux d'entiers.

La deuxième ligne des données d'entrée doit contenir les N entiers qui figurent dans le tableau A, chacun des entiers étant séparé des autres par un espace simple.

La troisième ligne des données d'entrée doit contenir les N entiers qui figurent dans le tableau B, chacun des entiers étant séparé des autres par un espace simple.

Le tableau ci-dessous détaille la répartition des 15 points disponibles.

Points	**Bornes de N**	**Bornes de A_i et B_i**
2	$N = 2$	$1 \leq A_i, B_i \leq 3$
4	$1 \leq N \leq 8$	$1 \leq A_i, B_i \leq 8$
4	$1 \leq N \leq 500$	$1 \leq A_i, B_i \leq 3000$
5	$1 \leq N \leq 300\,000$	$1 \leq A_i, B_i \leq 300\,000$

Notez que pour une sous-tâche valant M points, $\left\lfloor \frac{M}{2} \right\rfloor$ points seront attribués pour une solution qui ne produit correctement que la première ligne de sortie.

English version appears before the French version

Précisions par rapport aux données de sortie
La première ligne des données de sortie devrait afficher `YES` s'il existe une séquence de glissements qui peut transformer le tableau A en tableau B. Sinon, la première ligne des données de sortie devrait afficher `NO`.

Si la première ligne des données de sortie affiche `YES`, la ligne suivante doit contenir un entier non négatif K ($K \leq N$), représentant le nombre de glissements.

Chacune des K lignes suivantes contient trois valeurs, D_j, ℓ_j, et r_j, chacune étant séparée des autres par un espace simple. La valeur D_j est soit `R`, soit `L`, indiquant respectivement que le $j^{\text{ième}}$ glissement est soit un glissement vers la droite, soit un glissement vers la gauche. Les valeurs ℓ_j et r_j représentent l'extrémité gauche et l'extrémité droite du glissement ($0 \leq \ell_j \leq r_j < N$).

Donnés d'entrée d'un 1$^{\text{er}}$ exemple
```
3
3 1 2
3 1 1
```

Donnés de sortie du 1$^{\text{er}}$ exemple
```
YES
1
R 1 2
```

Donnés d'entrée d'un 2$^{\text{e}}$ exemple
```
4
1 2 4 3
1 4 2 3
```

Donnés de sortie du 2$^{\text{e}}$ exemple
```
NO
```

Donnés d'entrée d'un 3$^{\text{e}}$ exemple
```
4
2 1 4 3
2 1 4 3
```

Donnés de sortie du 3$^{\text{e}}$ exemple
```
YES
0
```

Problem S4: Painting Roads

Problem Description

Alanna, the mayor of Kitchener, has successfully improved the city's road plan. However, a travelling salesperson from the city of RedBlue complained that the roads are not colourful enough. Alanna's second job is to paint some of the roads.

Kitchener's road plan can be represented as a collection of N intersections with M roads, where the i-th road connects intersections u_i and v_i. All roads are initially grey. Alanna would like to paint some of the roads in red or blue such that the following condition is satisfied:

- Whenever there is a grey road that connects u_i and v_i, there is also a path of roads from u_i to v_i such that the roads on the path alternate between red and blue, without any of the roads on this path being grey.

To lower the city's annual spending, Alanna would like to minimize the number of painted roads. Can you help Alanna design a plan that meets all the requirements?

Input Specification

The first line contains two integers N and M ($1 \leq N, M \leq 2 \cdot 10^5$).

The i-th of the next M lines contains two integers u_i and v_i, meaning that there exists a road from intersection u_i to intersection v_i ($1 \leq u_i, v_i \leq N, u_i \neq v_i$).

There is at most one road between any unordered pair of intersections.

The following table shows how the available 15 marks are distributed:

Marks	Additional Constraints
2	There is a road connecting intersection i with intersection $i+1$ for all $1 \leq i < N$ (and possibly other roads).
3	We can reach any intersection from any other intersection, and $N = M$.
3	No road belongs to two or more simple cycles (see Definition below).
7	None

Definition: if we denote a road between intersections u and v as $u \leftrightarrow v$, then a *simple cycle* is a sequence $w_1 \leftrightarrow w_2 \leftrightarrow \ldots \leftrightarrow w_k \leftrightarrow w_1$ where $k \geq 3$ and all w_i are distinct.

Output Specification

Output a string of M characters, representing the paint plan. The i-th character should be `R` if the i-th road is to be painted red, `B` if i-th road is to be painted blue, or `G` (for "grey") if the i-th road is to be left unpainted.

La version française figure à la suite de la version anglaise.

Remember that you must minimize the number of painted roads while satisfying the condition. If there are multiple possible such plans, output any of them.

Sample Input 1

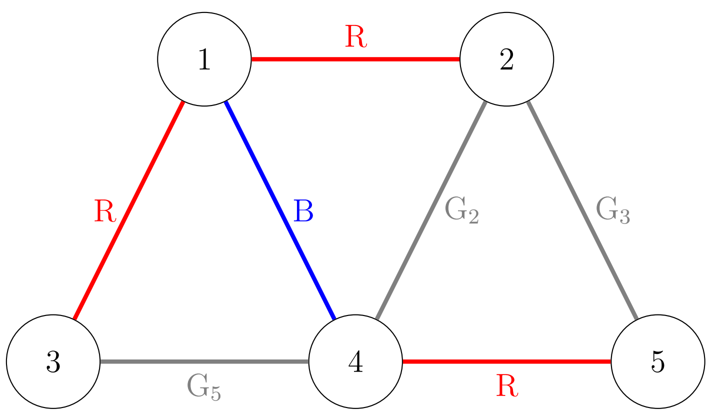
```
5 7
1 2
2 4
5 2
4 5
4 3
1 3
1 4
```

Output for Sample Input 1

```
RGGRGRB
```

Explanation of Output for Sample Input 1

A diagram of the intersections along with a valid paint plan that minimizes the number of painted roads is shown below. Note that the colours are shown on each road as R (red), B (blue), or G (grey).



All the unpainted roads satisfy the condition:

- The 2nd road, labelled G_2, connects intersection 2 with intersection 4. The path through intersections $2, 1, 4$ alternates red, blue.
- The 3rd road, labelled G_3, connects intersection 5 with intersection 2. The path through intersections $5, 4, 1, 2$ alternates red, blue, red.
- The 5th road, labelled G_5, connects intersection 4 with intersection 3. The path through intersections $4, 1, 3$ alternates blue, red.

La version française figure à la suite de la version anglaise.

Sample Input 2

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

Output for Sample Input 2

```
BB
```

Explanation of Output for Sample Input 2

Note that it is possible for Kitchener to be disconnected.

La version française figure à la suite de la version anglaise.

Problème S4 : Peindre les routes

Énoncé du problème

Alanna, la mairesse de Kitchener, a réussi à améliorer le plan routier de la ville. Cependant, un vendeur itinérant de la ville de RougeBleu s'est plaint que les routes manquaient de couleur. Par conséquent, la nouvelle mission d'Alanna consiste à peindre certaines des routes.

Le plan routier de Kitchener est composé de N intersections avec M routes, où la $i^{\text{ième}}$ route relie les intersections u_i et v_i. Initialement, toutes les routes sont grises. Alanna aimerait peindre certaines routes en rouge ou en bleu de manière que la condition suivante soit remplie :

— Pour toute route grise reliant u_i à v_i, il doit exister un itinéraire de u_i à v_i composé de routes dont les couleurs alternent entre rouge et bleu, sans qu'aucune route de cet itinéraire ne soit grise.

Dans l'optique de limiter les dépenses annuelles de la ville, Alanna souhaite minimiser le nombre de routes à peindre. Pouvez-vous aider Alanna à concevoir un plan qui répond à toutes ces exigences ?

Précisions par rapport aux données d'entrée

La première ligne des données d'entrée doit contenir deux entiers N et M ($1 \leq N, M \leq 2 \cdot 10^5$).

La $i^{\text{ième}}$ ligne des M lignes suivantes doit contenir deux entiers u_i et v_i, indiquant qu'il existe une route reliant l'intersection u_i à l'intersection v_i ($1 \leq u_i, v_i \leq N, u_i \neq v_i$).

Il existe au maximum une route entre chaque paire non ordonnée d'intersections.

Le tableau ci-dessous détaille la répartition des 15 points disponibles.

Points	Contraintes additionnelles
2	Il existe une route reliant l'intersection i à l'intersection $i+1$ pour tout $1 \leq i < N$ (et possiblement d'autres routes).
3	Il est possible de se rendre à n'importe quelle intersection depuis une autre et $N = M$.
3	Aucune route n'appartient à deux ou plus cycles simples (voir la définition ci-dessous).
7	Aucune

Définition : soit $u \leftrightarrow v$ une route qui relie les intersections u et v. Un *cycle simple* est une suite $w_1 \leftrightarrow w_2 \leftrightarrow \ldots \leftrightarrow w_k \leftrightarrow w_1$, w_i étant tous distincts et $k \geq 3$.

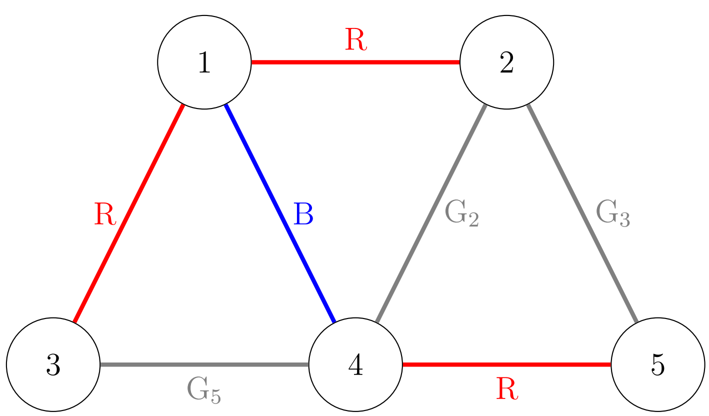
Précisions par rapport aux données de sortie
Les données de sortie devraient afficher une chaîne de M caractères, représentant le plan de peinture. Le $i^{\text{ième}}$ caractère devrait être `R` si la $i^{\text{ième}}$ route doit être peinte en rouge, `B` si la $i^{\text{ième}}$ route doit être peinte en bleu ou `G` (pour « gris ») si la $i^{\text{ième}}$ route ne doit pas être peinte.

Il est impératif de minimiser le nombre de routes à peindre tout en remplissant la condition établie. S'il existe plusieurs plans possibles, les données de sortie peuvent en afficher un quelconque.

Donnés d'entrée d'un 1$^{\text{er}}$ exemple
```
5 7
1 2
2 4
5 2
4 5
4 3
1 3
1 4
```

Donnés de sortie du 1$^{\text{er}}$ exemple
```
RGGRGRB
```

Justification des donnés de sortie du 1$^{\text{er}}$ exemple
La figure ci-dessous illustre les intersections ainsi qu'un plan de peinture qui minimise le nombre de routes à peindre. Les couleurs des routes sont représentées par les lettres R (rouge), B (bleu) ou G (gris).



Toutes les routes non peintes remplissent la condition :

- La 2e route, soit la route G_2, relie l'intersection 2 à l'intersection 4. Les couleurs du chemin passant par les intersections $2, 1, 4$ alternent de la manière suivante : rouge, bleu.
- La 3e route, soit la route G_3, relie l'intersection 5 à l'intersection 2. Les couleurs du chemin passant par les intersections $5, 4, 1, 2$ alternent de la manière suivante : rouge, bleu, rouge.
- La 5e route, soit la route G_5, relie l'intersection 4 à l'intersection 3. Les couleurs du chemin passant par les intersections $4, 1, 3$ alternent de la manière suivante : bleu, rouge.

Donnés d'entrée d'un 2e exemple

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

Donnés de sortie du 2e exemple

```
BB
```

Justification des donnés de sortie du 2e exemple

Remarquons qu'il est possible que Kitchener soit déconnecté.

Problem S5: Chocolate Bar Partition

Problem Description

Maxwell has a chocolate bar that he wants to share with his friends. The chocolate bar can be represented as a 2 by N array of integers $T_{i,j}$, the tastiness of each square. Maxwell would like to split the entire chocolate bar into connected parts such that the average (mean) tastiness of the chocolate bar is the same for each part. Maxwell would like to know what is the maximum number of connected parts he can split his chocolate bar into as described above.

A part is considered connected if you can visit every cell by moving up, down, left or right.

Input Specification

The first line of input will consist of one positive integer N, representing the length of the chocolate bar.

The second line of input contains N spaced integers representing the top row of the chocolate bar where the j-th integer from the left represents $T_{1,j}$.

Similarly, the third line of input contains N spaced integers representing the bottom row of the chocolate bar where the j-th integer from the left represents $T_{2,j}$.

The following table shows how the available 15 marks are distributed:

Marks	**Bounds on N**	**Bounds on $T_{i,j}$**
2	$N = 2$	$0 \leq T_{i,j} \leq 5$
2	$1 \leq N \leq 8$	$0 \leq T_{i,j} \leq 20$
1	$1 \leq N \leq 20$	$0 \leq T_{i,j} \leq 20$
2	$1 \leq N \leq 100$	$0 \leq T_{i,j} \leq 20$
2	$1 \leq N \leq 1\ 000$	$0 \leq T_{i,j} \leq 100$
3	$1 \leq N \leq 2\ 000$	$0 \leq T_{i,j} \leq 100\ 000$
3	$1 \leq N \leq 200\ 000$	$0 \leq T_{i,j} \leq 100\ 000\ 000$

Output Specification

Output a single integer, representing the maximum number of connected parts Maxwell can split his chocolate bar into.

Sample Input 1

```
2
5 4
6 5
```

La version française figure à la suite de la version anglaise.

Output for Sample Input 1
```
2
```

Explanation of Output for Sample Input 1
An example of how to split this chocolate bar optimally into 2 parts is to have the bottom right corner as its own part and the rest of the chocolate as the second part, as shown below.

5	4
6	5

Each piece will have an average tastiness of 5.

Sample Input 2
```
5
1 0 1 2 0
0 2 0 3 1
```

Output for Sample Input 2
```
5
```

Explanation of Output for Sample Input 2
One way to get the optimal split is shown in the following picture:

1	0	1	2	0
0	2	0	3	1

Note that each piece has an average tastiness of 1.

La version française figure à la suite de la version anglaise.

Problème S5 : Diviser une barre de chocolat

Énoncé du problème

Maxwell a une barre de chocolat qu'il veut partager avec ses amis. La barre de chocolat peut être représentée par un tableau $2 \times N$ d'entiers $T_{i,j}$, $T_{i,j}$ représentant le niveau de saveur de chaque carré. Maxwell aimerait diviser la barre de chocolat en morceaux connectés de sorte que la saveur moyenne de la barre de chocolat soit la même pour chaque morceau. Maxwell aimerait savoir quel est le nombre maximum de morceaux connectés en lesquels il pourrait diviser sa barre de chocolat, conformément au critère décrit précédemment.

On considère qu'un morceau est « connecté » si l'on peut visiter chaque carré en se déplaçant vers le haut, le bas, la gauche ou la droite.

Précisions par rapport aux données d'entrée

La première ligne des données d'entrée doit contenir un entier strictement positif N, représentant la longueur de la barre de chocolat.

La deuxième ligne des données d'entrée contient N entiers, chacun étant séparé des autres par un espace simple, représentant la rangée supérieure de la barre de chocolat où le $j^{\text{ième}}$ entier à partir de la gauche représente $T_{1,j}$.

De même, la troisième ligne des données d'entrée contient N entiers, chacun étant séparé des autres par un espace simple, représentant la rangée inférieure de la barre de chocolat où le $j^{\text{ième}}$ entier à partir de la gauche représente $T_{2,j}$.

Le tableau ci-dessous détaille la répartition des 15 points disponibles.

Points	**Bornes de N**	**Bornes de $T_{i,j}$**
2	$N = 2$	$0 \leq T_{i,j} \leq 5$
2	$1 \leq N \leq 8$	$0 \leq T_{i,j} \leq 20$
1	$1 \leq N \leq 20$	$0 \leq T_{i,j} \leq 20$
2	$1 \leq N \leq 100$	$0 \leq T_{i,j} \leq 20$
2	$1 \leq N \leq 1\ 000$	$0 \leq T_{i,j} \leq 100$
3	$1 \leq N \leq 2\ 000$	$0 \leq T_{i,j} \leq 100\ 000$
3	$1 \leq N \leq 200\ 000$	$0 \leq T_{i,j} \leq 100\ 000\ 000$

Précisions par rapport aux données de sortie

Les données de sortie devraient afficher le nombre maximum de morceaux connectés en lesquels Maxwell pourrait diviser sa barre de chocolat.

Donnés d'entrée d'un 1er exemple

```
2
```

```
5 4
6 5
```

Donnés de sortie du 1^er^ exemple

```
2
```

Justification des donnés de sortie du 1^er^ exemple

Pour diviser cette barre de chocolat en deux morceaux de manière optimale, on peut par exemple considérer le coin inférieur droit comme étant un morceau et le reste du chocolat comme étant le deuxième morceau, comme dans la figure ci-dessous.

5	4
6	5

Chaque morceau a une saveur moyenne de 5.

Donnés d'entrée d'un 2^e^ exemple

```
5
1 0 1 2 0
0 2 0 3 1
```

Donnés de sortie du 2^e^ exemple

```
5
```

Justification des donnés de sortie du 2^e^ exemple

Voici une manière optimale dont on pourrait diviser la barre de chocolat :

1	0	1	2	0
0	2	0	3	1

Chaque morceau a une saveur moyenne de 1.