Problem S1: Positioning Peter's Paintings

Problem Description

Peter the painter just finished painting two rectangular paintings and would like to display both on a rectangular wall which has the smallest perimeter possible. The first painting has a base of length A units and a height of length B units. The second painting has a base of length X units and a height of length Y units.

Peter has a few conditions on how to arrange his paintings on the rectangular wall. The first condition is that the paintings must be upright, meaning that the bases of the paintings are parallel to the floor. The second condition is that he would like to display both paintings in full, meaning that they cannot overlap each other. Please help determine the rectangular wall of minimum perimeter such that the paintings can be displayed without violating his conditions.

Input Specification

The one line of input will consist of four space-separated positive integers, A, B, X, Y ($1 \leq A, B, X, Y \leq 10^8$).

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

Marks	Brief Description
5	Paintings are congruent squares
5	Paintings are squares
5	Paintings are rectangles (possibly squares)

Output Specification

Output a single integer representing the minimum perimeter of a rectangular wall without violating Peter's conditions.

Sample Input 1

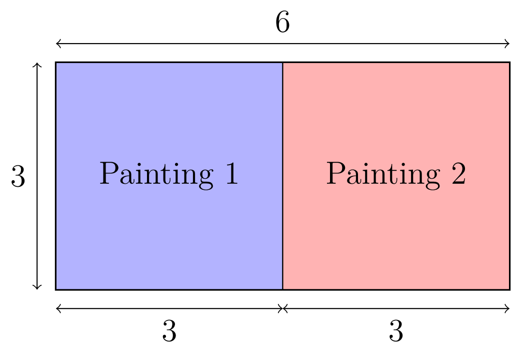
```
3 3 3 3
```

Output for Sample Input 1

```
18
```

Explanation of Output for Sample Input 1

This test case satisfies all subtasks. An optimal arrangement using a 6-by-3 wall is shown below.



Sample Input 2

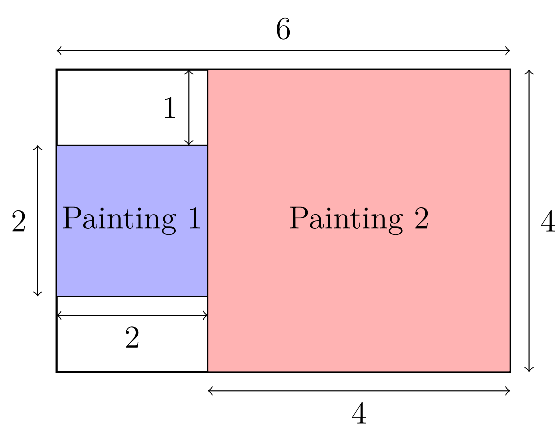
```
2 2 4 4
```

Output for Sample Input 2

```
20
```

Explanation of Output for Sample Input 2

This test case satisfies the second and third subtasks. An optimal arrangement using a 6-by-4 wall is shown below.



Sample Input 3

```
1 2 3 1
```

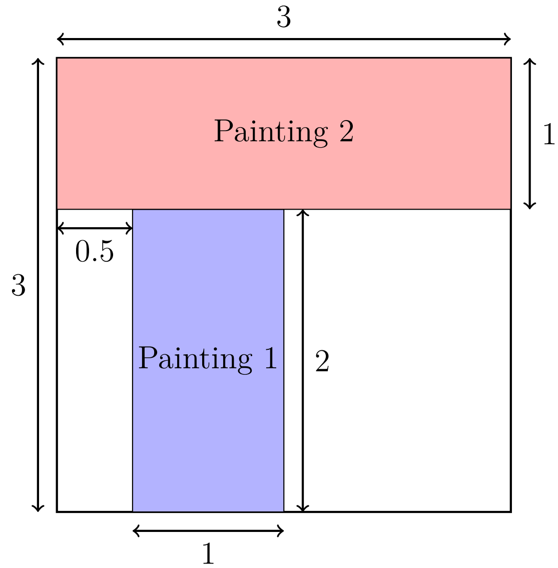
Output for Sample Input 3

```
12
```

Explanation of Output for Sample Input 3

This test case satisfies the last subtask. An optimal arrangement using a 3-by-3 wall is shown below.

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



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

Problème S1 : Positionnement des peintures de Paul

Énoncé du problème

Paul, le peintre, vient de terminer deux peintures rectangulaires et souhaite les exposer sur un mur rectangulaire au plus petit périmètre possible. La première peinture a une base de longueur A et une hauteur de longueur B. La deuxième peinture a une base de longueur X et une hauteur de longueur Y.

Paul a quelques conditions quant au positionnement de ses peintures sur le mur rectangulaire. Premièrement, il souhaite qu'elles soient positionnées à l'équerre et à l'endroit, c'est-à-dire que leur base est parallèle au sol. Il souhaite également que les deux peintures soient entièrement visibles, ce qui signifie qu'elles ne peuvent pas être superposées. Votre tâche consiste à déterminer le périmètre minimum du mur rectangulaire permettant d'exposer les peintures en respectant les conditions de Paul.

Précisions par rapport aux données d'entrée

La ligne de données d'entrée contient quatre entiers positifs, A, B, X, Y ($1 \leq A, B, X, Y \leq 10^8$), chacun étant séparé des autres par une espace.

Le tableau suivant détaille la répartition des 15 points disponibles.

Points	Brève description
5	Les peintures sont en forme de carré isométrique.
5	Les peintures sont en forme de carré.
5	Les peintures sont en forme de rectangles (et possiblement de carré).

Précisions par rapport aux données de sortie

Les données de sortie devraient afficher le périmètre minimum du mur rectangulaire en respectant les conditions de Paul.

Données d'entrée d'un 1er exemple

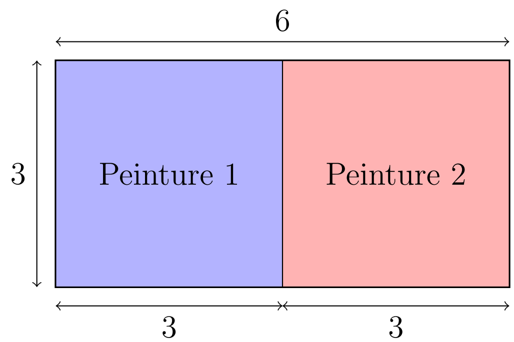
```
3 3 3 3
```

Données de sortie du 1er exemple

```
18
```

Justification des données de sortie du 1er exemple

Ce test satisfait à toutes les sous-tâches. Une disposition optimale d'un mur aux dimensions de 6×3 est illustrée ci-dessous.



Données d'entrée d'un 2^{e} exemple

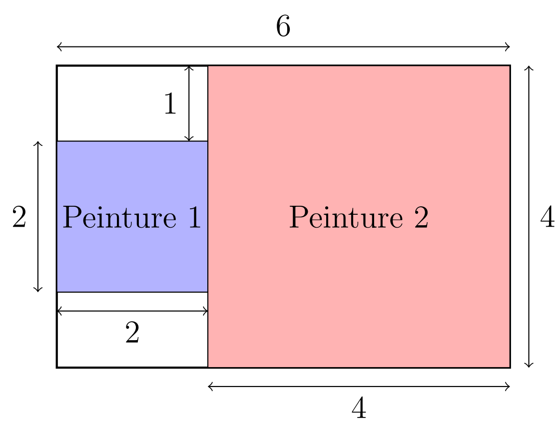
```
2 2 4 4
```

Données de sortie du 2^{e} exemple

```
20
```

Justification des données de sortie du 2^{e} exemple

Ce test satisfait à la deuxième et à la troisième sous-tâches. Une disposition optimale d'un mur aux dimensions de 6×4 est illustrée ci-dessous.



Données d'entrée d'un 3^{e} exemple

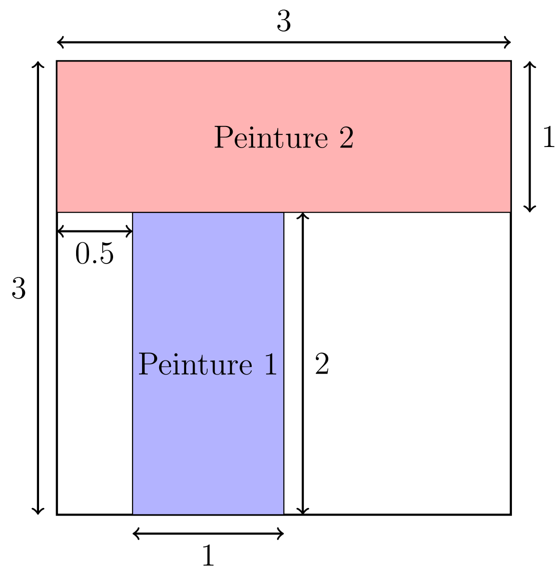
```
1 2 3 1
```

Données de sortie du 3^{e} exemple

```
12
```

Justification des données de sortie du 3^{e} exemple

Ce test satisfait à la dernière sous-tâche. Une disposition optimale d'un mur aux dimensions de 3×3 est illustrée ci-dessous.



English version appears before the French version

Problem S2: Cryptogram Cracking Club

Problem Description

Cyrene, the captain of the Cryptogram Cracking Club (CCC), came across a concerningly long cipher. Conveniently, this cipher is composed of lower-case characters (`a-z`). Comfortingly, the cipher is composed of a pattern that repeats infinitely.

Cyrene wishes to locate the c-th character of the cipher. To make your job easier, the CCC members have extracted the repeated pattern and compressed it using the Run-Length Encoding (RLE) algorithm, which replaces consecutive repeated characters with a single occurrence of the character followed by a count of how many times it was repeated. For example, for the pattern `aaaabccdddd`, the RLE algorithm outputs `a4b1c2d4`.

You are given the output of the RLE algorithm for a certain pattern. Can you determine the c-th character of the long cipher that is formed by repeating this pattern infinitely?

Input Specification

The first line of input will consist of a string S, representing a pattern produced by the RLE algorithm. The length of S will be at least 2 and at most $2 \cdot 10^5$. Additionally, all numbers appearing in S are between 1 and 10^{12}.

The next line of input contains a single integer c, representing the index of the character you wish to locate, starting from index 0.

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

Marks	Bounds on c	Additional Constraints
6	$0 \leq c \leq 2000$	All numbers appearing in S are between 1 and 9 (inclusive) and the length of the repeated pattern is at most 2000 characters.
3	$0 \leq c \leq 10^6$	The length of the repeated pattern is at most 10^6 characters.
3	$0 \leq c \leq 10^{12}$	The length of the repeated pattern is at most 10^6 characters.
3	$0 \leq c \leq 10^{12}$	No additional constraints.

Output Specification

Output the c-th character of the long cipher.

Sample Input 1

```
r2d2
8
```

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

Output for Sample Input 1

```
r
```

Explanation of Output for Sample Input 1

The output of the RLE algorithm `r2d2` corresponds to the pattern `rrdd`, which creates the infinitely long cipher `rrddrrdd`$\boxed{\texttt{r}}$`rddrrdd...`, where the $c = 8$th character is `r`. In this example, the $c = 8$th character is highlighted with a box around it.

Sample Input 2

```
a4b1c2d10
100
```

Output for Sample Input 2

```
d
```

Explanation of Output for Sample Input 2

The output of the RLE algorithm `a4b1c2d10` corresponds to the pattern `aaaabccdddddddddd`. When repeated infinitely, the $c = 100$th character is `d`.

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

Problème S2 : Club des cryptographes ingénieux

Énoncé du problème
Céline, la capitaine du Club des cryptographes ingénieux (CCI), doit craquer un très long cryptogramme. De façon pratique, ce cryptogramme est composé de caractères minuscules (`a` à `z`). Par ailleurs, ce cryptogramme est composé d'une séquence de caractères qui se répète à l'infini.

Céline souhaite localiser le $c^{\text{ième}}$ caractère du cryptogramme. Pour vous faciliter la tâche, les membres du CCI ont extrait la séquence répétée et l'ont compressée à l'aide de l'encodage de plage, qui remplace les caractères répétés consécutifs par une seule occurrence du caractère suivie du nombre de fois où il a été répété. Par exemple, pour la séquence `aaaabccdddd`, l'encodage de plage produit le code : `a4b1c2d4`.

Les données de sortie de l'encodage de plage pour une séquence quelconque vous sont fournies. Pouvez-vous déterminer le $c^{\text{ième}}$ caractère du long cryptogramme composé de répétitions infinies de cette séquence ?

Précisions par rapport aux données d'entrée
La première ligne de données d'entrée consistera en une chaîne de caractères S, représentant la séquence produite par l'algorithme d'encodage de plage. La longueur de S sera d'au moins 2 et d'au plus $2 \cdot 10^5$. En outre, tous les nombres figurant dans S sont compris entre 1 et 10^{12}.

La ligne de données d'entrée suivante contient un seul entier c, représentant l'indice du caractère que vous souhaitez localiser, en commençant par l'indice 0.

Le tableau suivant détaille la répartition des 15 points disponibles.

Points	**Bornes de c**	**Restrictions additionnelles**
3	$0 \leq c \leq 2000$	En outre, tous les nombres inclus dans S sont compris entre 1 et 9 (inclusivement) et la longueur de la séquence répétée est de 2000 caractères au maximum.
3	$0 \leq c \leq 10^6$	La longueur de la séquence répétée est de 10^6 caractères au maximum.
3	$0 \leq c \leq 10^{12}$	La longueur de la séquence répétée est de 10^6 caractères au maximum.
3	$0 \leq c \leq 10^{12}$	Aucune restriction additionnelle.

Précisions par rapport aux données de sortie
Les données de sortie devraient contenir le $c^{\text{ième}}$ caractère du cryptogramme long.

Données d'entrée d'un 1er exemple

```
r2d2
8
```

Données de sortie du 1er exemple

```
r
```

Justification des données de sortie du 1er exemple

Les données de sortie de l'algorithme d'encodage de plage, `r2d2`, correspondent à la séquence `rrdd` du code infini `rrddrrdd`$\boxed{\texttt{r}}$`rddrrdd`..., où le caractère $c = 8^e$ est `r`. Dans cet exemple, le caractère $c = 8^e$ est encadré.

Données d'entrée d'un 2e exemple

```
a4b1c2d10
100
```

Données de sortie du 2e exemple

```
d
```

Justification des données de sortie du 2e exemple

Les données de sortie de l'algorithme d'encodage de plage devraient contenir le code : `a4b1c2d10`, correspondant à la séquence `aaaabccdddddddddd`. Lorsqu'il est répété à l'infini, le $c = 100^e$ caractère est `d`.

Problem S3: Pretty Pens

Problem Description

You are taking an art class, and your current art assignment is very algorithmic.

You have N pens, each of which has a single colour, represented by an integer from 1 to M. Initially, the colour of the i-th pen is given by C_i. In addition, your pens are pretty, with the i-th pen having a prettiness of P_i.

For your assignment, you need to create a picture using M strokes, each from a pen of a different colour. The prettiness of your picture is the sum of the prettiness of the pens used to create the picture.

Your teacher has given you some room for artistic expression: before you create this pretty picture, you are allowed to change at most one pen to any other colour. After this picture is drawn, the pen will revert back to its original colour.

Your teacher will give you $\frac{1}{3}$ of the marks (5 of 15 total marks) for the art assignment based on creating the prettiest picture you can.

To push your creative limits, your teacher also has Q more pictures for you to create, which compose the remaining $\frac{2}{3}$ of the marks (10 of 15 total marks) for your art assignment. Before each picture, there will be one of two possible changes to the set of pens available:

- `1 i c` indicates that the colour of the i-th pen changes to c.
- `2 i p` indicates that the prettiness of the i-th pen changes to p.

The changes are executed sequentially (so the first change modifies the initial setup, the second change modifies the result of applying the first change, and so on).

As in the first picture you created, you are allowed to change the colour of at most one pen before the picture is created. Note that if you do change the colour of one pen, it affects only the next picture you draw, and the pen will revert to its previous colour before the next change is applied (if any).

What is the prettiness of the prettiest $Q + 1$ pictures you can create?

Input Specification

The first line of input contains three space-separated integers N, M, and Q ($1 \leq M \leq N \leq 200\ 000$, $0 \leq Q \leq 200\ 000$).

The next N lines each contain two space-separated integers, C_i and P_i ($1 \leq C_i \leq M, 1 \leq P_i \leq 10^9$), indicating the colour and prettiness of the i-th pen.

The next Q lines each contain three space-separated integers, beginning with 1 or 2. If the first integer is 1, it is followed by two integers i_j and c_j ($1 \leq i_j \leq N, 1 \leq c_j \leq M$), representing a change of the colour of the i_j-th pen to c_j. If the first integer is 2, it is followed by two integers i_j and p_j ($1 \le i_j \le N, 1 \le p_j \le 10^9$), representing a change of the prettiness of the i_j-th pen to p_j.

It is guaranteed that initially and after each change, there is at least one pen with each of the M colours.

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

Marks	**Bounds on N and M**	**Bounds on Q**	**Additional constraints**
5	$1 \le M \le N \le 200\,000$	$Q = 0$	None
2	$M = 1$; $1 \le N \le 200\,000$	$0 \le Q \le 200\,000$	None
2	$M = 2$; $2 \le N \le 200\,000$	$0 \le Q \le 200\,000$	None
2	$M \le 10$; $1 \le M \le N \le 200\,000$	$0 \le Q \le 200\,000$	None
2	$1 \le M \le N \le 200\,000$	$0 \le Q \le 200\,000$	All prettiness values are distinct
2	$1 \le M \le N \le 200\,000$	$0 \le Q \le 200\,000$	None

Output Specification

The output is $Q + 1$ lines, with the j-th line containing one integer, the largest possible prettiness value obtainable after the first $j - 1$ changes have been performed.

Sample Input 1

```
6 3 0
1 6
2 9
3 4
2 7
3 9
1 3
```

Output for Sample Input 1

```
25
```

Explanation of Output for Sample Input 1

There are six pens available in three different colours. The set of pens is:

- two pens of colour 1, with prettiness 6 and 3,
- two pens of colour 2, with prettiness 9 and 7,
- two pens of colour 3, with prettiness 4 and 9.

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

If we do not change the colour of any pens, the prettiest picture has prettiness $6+9+9=24$. However, if we change the colour of pen 4 from colour 2 to colour 1, we can create a picture with prettiness $7+9+9=25$, which is the prettiest picture that can be created.

Sample Input 2

```
3 2 2
1 20
2 30
1 10
1 3 2
2 3 25
```

Output for Sample Input 2

```
50
50
55
```

Explanation of Output for Sample Input 2

There are three pens with two different colours available.

Before the first change, using pen 1 and pen 2, with colours 1 and 2, respectively, achieves a prettiness of $20+30=50$.

After the first change, pen 3 has colour 2. There is no alteration in the maximum prettiness, even if we could switch one pen: picking pens 1 and 2 will yield the maximum prettiness.

After the second change, pen 3 will have colour 2 and prettiness 25. Before the final picture is created, we can change the colour of pen 2 to colour 1, and use pens 2 and 3 to achieve the maximum prettiness of $30+25=55$.

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

Problème S3 : Crayons colorés

Énoncé du problème
Vous suivez un cours d'art et votre projet actuel est lié à l'algorithmique.

Vous disposez de N crayons, chacun ayant une seule couleur, représentée par un entier de 1 à M. Initialement, la couleur du $i^{\text{ième}}$ crayon est donnée par C_i. Vos crayons sont jolis : le $i^{\text{ième}}$ crayon a une valeur de beauté de P_i.

Pour votre travail, vous devez créer une image en faisant M traits, chacun provenant d'un crayon d'une couleur différente. La beauté de votre œuvre est la somme des valeurs de beauté des crayons utilisés pour créer l'image.

Votre enseignant vous a laissé une certaine marge d'expression artistique : avant de créer ce beau dessin, vous avez le droit de changer au maximum un crayon pour une autre couleur. Une fois l'image dessinée, le crayon reprend sa couleur d'origine.

Votre enseignant vous attribuera $\frac{1}{3}$ des points (5 sur 15 points au total) pour votre projet artistique en fonction de sa valeur de beauté.

Pour repousser les limites de vos aptitudes créatives, votre enseignant vous propose également de créer Q images supplémentaires, qui vous attribueront $\frac{2}{3}$ des points (10 sur 15 points au total) pour votre projet artistique. Avant chaque image, vous aurez l'occasion d'effectuer l'un des deux changements possibles parmi l'ensemble de crayons disponibles.

— `1 i c` indique que la couleur du $i^{\text{ième}}$ crayon devient c.
— `2 i p` indique que la beauté du $i^{\text{ième}}$ crayon devient p.

Les changements sont exécutés de manière séquentielle, c'est-à-dire que le premier changement modifie la configuration initiale et le deuxième changement modifie le résultat du premier changement, et ainsi de suite.

Comme pour votre première œuvre, vous avez l'autorisation de changer au plus une couleur d'un crayon avant de créer votre image. Notez que si vous changez la couleur d'un crayon, cela n'affecte que l'image que vous vous apprêtez à créer. Le cas échéant, le crayon reviendra à sa couleur initiale avant que le changement suivant ne soit effectué.

Quelle est la valeur de beauté des $Q + 1$ plus belles images que vous puissiez créer ?

Précisions par rapport aux données d'entrée
La première ligne de données d'entrée contient trois entiers N, M et Q, chacun étant séparé des autres par une espace ($1 \leq M \leq N \leq 200\ 000$, $0 \leq Q \leq 200\ 000$).

Les lignes de données N suivantes contiennent chacune deux entiers, chacun étant séparé des autres par une espace, C_i et P_i ($1 \leq C_i \leq M, 1 \leq P_i \leq 10^9$), indiquant la couleur et la beauté du $i^{\text{ième}}$ crayon.

Les lignes de données Q suivantes contiennent chacune trois nombres entiers, chacun étant séparé des autres par une espace, commençant par 1 ou 2. Si le premier entier est 1, il est suivi de deux entiers i_j et c_j ($1 \leq i_j \leq N, 1 \leq c_j \leq M$), représentant un changement de la couleur du $i_j$$^{\text{ième}}$ crayon en c_j. Si le premier entier est 2, il est suivi de deux entiers i_j et p_j ($1 \leq i_j \leq N, 1 \leq p_j \leq 10^9$), représentant un changement de la valeur de beauté du $i_j$$^{\text{ième}}$ crayon en p_j.

Il est garanti qu'au début de la création et qu'après chaque changement, il y aura au moins un crayon de chacune des couleurs M.

Le tableau suivant détaille la répartition des 15 points disponibles.

Points	**Bornes des données N et M**	**Bornes des données Q**	**Restrictions additionnelles**
5	$1 \leq M \leq N \leq 200\,000$	$Q = 0$	Aucune
2	$M = 1\,;\, 1 \leq N \leq 200\,000$	$0 \leq Q \leq 200\,000$	Aucune
2	$M = 2\,;\, 2 \leq N \leq 200\,000$	$0 \leq Q \leq 200\,000$	Aucune
2	$M \leq 10\,;\, 1 \leq M \leq N \leq 200\,000$	$0 \leq Q \leq 200\,000$	Aucune
2	$1 \leq M \leq N \leq 200\,000$	$0 \leq Q \leq 200\,000$	Toutes les valeurs de beauté sont distinctes.
2	$1 \leq M \leq N \leq 200\,000$	$0 \leq Q \leq 200\,000$	Aucune

Précisions par rapport aux données de sortie

Les données de sortie devraient contenir des lignes de données $Q+1$, la $j^{\text{ième}}$ ligne contenant un entier, la plus grande valeur de beauté possible pouvant être obtenue après que les premiers changements $j-1$ aient été effectués.

Données d'entrée d'un 1er exemple

```
6 3 0
1 6
2 9
3 4
2 7
3 9
1 3
```

Données de sortie du 1er exemple

```
25
```

Justification des données de sortie du 1^er^ exemple
Six crayons sont disponibles en trois couleurs différentes. Voici la description de l'ensemble de crayons.

— Deux crayons de couleur 1, avec une valeur de beauté de 6 et 3.
— Deux crayons de couleur 2, avec une valeur de beauté de 9 et 7.
— Deux crayons de couleur 3, avec une valeur de beauté de 4 et 9.

Sans changer la couleur d'aucun crayon, la plus belle image aura une valeur de beauté de $6+9+9 = 24$. Cependant, en changeant la couleur du crayon 4 de la couleur 2 à la couleur 1, il est possible de créer une image d'une valeur de beauté de $7 + 9 + 9 = 25$, ce qui est la plus belle image pouvant être créé.

Données d'entrée d'un 2^e^ exemple

```
3 2 2
1 20
2 30
1 10
1 3 2
2 3 25
```

Données de sortie du 2^e^ exemple

```
50
50
55
```

Justification des données de sortie du 2^e^ exemple
Trois crayons sont disponibles en deux couleurs différentes.

Avant le premier changement, l'utilisation du crayon 1 et du crayon 2, avec les couleurs 1 et 2, respectivement, permet d'obtenir une beauté de $20 + 30 = 50$.

Après le premier changement, le crayon 3 a la couleur 2. La valeur de beauté maximale ne change pas, même si l'on change de crayon : les crayons 1 et 2 permettent d'obtenir la valeur de beauté maximale.

Après le deuxième changement, le crayon 3 aura la couleur 2, permettant d'obtenir une valeur de beauté de 25. Avant de créer l'image finale, il est possible de remplacer la couleur du crayon 2 par la couleur 1 et d'utiliser les crayons 2 et 3 pour obtenir la valeur de beauté maximale de $30 + 25 = 55$.

Problem S4: Floor is Lava

Problem Description

You're trapped in a scorching dungeon with N rooms numbered from 1 to N connected by M tunnels. The i-th tunnel connects rooms a_i and b_i in both directions, but the floor of the tunnel is covered in lava with temperature c_i.

To help you navigate the lavatic tunnels, you are wearing a pair of heat-resistant boots that initially have a chilling level of 0. In order to step through lava with temperature ℓ, your boots must have the same chilling level ℓ; if the chilling level is too low then the lava will melt your boots, and if it's too high then your feet will freeze as you cross the tunnel.

Luckily, when you're standing in a room, you can increase or decrease the chilling level of your boots by d for a cost of d coins. You start in room 1 and would like to reach the exit which you know is located in room N. What is the minimum cost to do so?

Input Specification

The first line of input contains two integers N and M $(1 \leq N, M \leq 200\,000)$.

The next M lines each contain three integers a_i, b_i, and c_i $(1 \leq a_i, b_i \leq N, a_i \neq b_i, 1 \leq c_i \leq 10^9)$, describing the i-th tunnel.

There is at most one tunnel connecting any pair of rooms, and it is possible to reach all other rooms from room 1.

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

Marks	Additional constraints
2	$M = N - 1$
4	For all tunnels, $1 \leq c_i \leq 10$
4	Each room has at most 5 outgoing tunnels
5	None

Output Specification

Output the minimum cost (in coins) to reach room N from room 1.

Sample Input

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

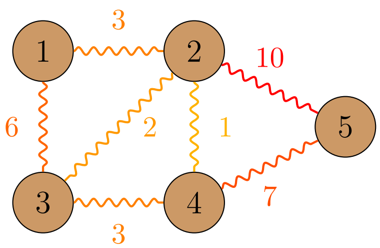
La version française figure à la suite de la version anglaise.

```
2 4 1
2 5 10
```

Output for Sample Input

```
9
```

Explanation of Output for Sample Input



A diagram of the dungeon is shown above. The optimal escape strategy is as follows:

1. Increase the chilling level to 3 for a cost of 3 coins.
2. Walk through the tunnel to room 2.
3. Decrease the chilling level to 2 for a cost of 1 coin.
4. Walk through the tunnel to room 3.
5. Increase the chilling level to 3 for a cost of 1 coin.
6. Walk through the tunnel to room 4.
7. Increase the chilling level to 7 for a cost of 4 coins.
8. Walk through the tunnel to room 5 and escape.

This has a total cost of 9 coins, and it can be shown that no cheaper route exists.

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

Problème S4 : Tunnels de lave

Énoncé du problème

Vous êtes piégé dans un donjon suffocant composé de salles N numérotées de 1 à N et reliées par des tunnels M. Le $i^{\text{ième}}$ tunnel relie les salles a_i et b_i dans les deux sens, mais le sol du tunnel est recouvert de lave à la température c_i.

Pour vous aider à naviguer dans les tunnels de lave, vous portez une paire de bottes résistantes à la chaleur, dont le niveau de refroidissement initial est de 0. Pour traverser la lave dont la température équivaut à ℓ, vos bottes doivent avoir le même niveau de refroidissement ℓ. Si le niveau de refroidissement est trop bas, la lave fera fondre vos bottes. S'il est trop élevé, vos pieds gèleront lorsque vous traverserez le tunnel.

Heureusement, lorsque vous vous trouvez dans une salle, vous pouvez augmenter ou diminuer le niveau de refroidissement de vos bottes de d pour un coût de d cristaux. Vous commencez dans la salle 1 et souhaitez atteindre la sortie qui se trouve dans la salle N. Quel est le coût minimum pour y parvenir ?

Précisions par rapport aux données d'entrée

La première ligne de données d'entrée contient deux entiers N et M $(1 \leq N, M \leq 200\,000)$.

Les lignes de données M suivantes contiennent chacune trois entiers, a_i, b_i et c_i ($1 \leq a_i, bi \leq N$, $a_i \neq b_i$, $1 \leq c_i \leq 10^9$), décrivant le $i^{\text{ième}}$ tunnel.

Il existe au maximum un tunnel reliant chaque paire de salles, et il est possible d'atteindre toutes les autres salles à partir de la salle 1.

Le tableau suivant détaille la répartition des 15 points disponibles.

Points	Restrictions additionnelles
2	$M = N - 1$
4	Pour tous les tunnels, $1 \leq c_i \leq 10$
4	Chaque salle possède au maximum 5 tunnels sortants.
5	Aucune

Précisions par rapport aux données de sortie

Les données de sortie devraient indiquer le coût minimum (en cristaux) pour atteindre la salle N à partir de la salle 1.

Exemple de données d'entrée

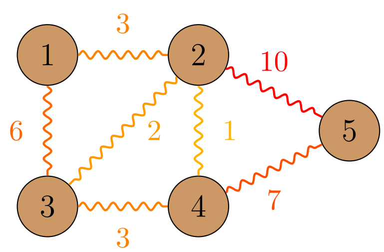
```
5 7
1 2 3
2 3 2
```

```
1 3 6
3 4 3
4 5 7
2 4 1
2 5 10
```

Exemple de données de sortie

```
9
```

Justification des données de sortie



Un schéma du donjon figure ci-dessus. La stratégie d'évasion optimale est la suivante.

1. Augmenter le niveau de refroidissement à 3 pour un coût de 3 cristaux.
2. Traverser le tunnel jusqu'à la salle 2.
3. Diminuer le niveau de refroidissement à 2 pour un coût de 1 cristal.
4. Traverser le tunnel jusqu'à la salle 3.
5. Augmenter le niveau de refroidissement à 3 pour un coût de 1 cristal.
6. Traverser le tunnel jusqu'à la salle 4.
7. Augmenter le niveau de refroidissement à 7 pour un coût de 4 cristaux.
8. Traverser le tunnel jusqu'à la salle 5 et s'échapper.

Le coût total est de 9 cristaux et il est possible de démontrer qu'il n'existe pas d'itinéraire moins coûteux.

Problem S5: To-Do List

Problem Description

Wow, your to-do list is empty...but not for long! Over the next few seconds, you'll have to handle Q updates to your to-do list.

For the first type of update, you will have to add a new homework assignment to your to-do list. This assignment will be released at the beginning of second s, and will take t seconds to complete ($1 \leq s, t \leq 10^6$). For the second type of update, you will have to remove the i-th homework assignment that was added to your to-do list.

After each update, you wonder: what's the earliest time you can finish all of the homework assignments in your to-do list? You can only work on one assignment at a time, and you must finish a homework assignment once you start it without switching to another assignment.

Input Specification

The first line of input contains an integer Q.

The next Q lines each contain a line starting with a character `A` or `D`. A line starting with `A` represents the first type of update and ends with two space-separated **encrypted*** integers s' and t'. A line starting with `D` represents the second type of update and ends with an encrypted integer i'. It is guaranteed that there have been at least i assignments added and that the i-th assignment to be added has not been removed yet.

It is guaranteed that there is at least one homework assignment on your to-do list after every update.

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

Marks	Bounds on Q	Additional constraints
2	$1 \leq Q \leq 3\,000$	None
6	$1 \leq Q \leq 10^6$	Only updates of the first type
7	$1 \leq Q \leq 10^6$	None

*Note that the input for this problem is encrypted. To decrypt and obtain the actual values of s, t, and i, you may use the following formulas:

$$s = (s' + \text{ans}) \bmod (10^6 + 3) \quad t = (t' + \text{ans}) \bmod (10^6 + 3) \quad i = (i' + \text{ans}) \bmod (10^6 + 3)$$

Here, ans represents the answer after the previous update and is initially 0 before any updates. It may also be useful to note that mod corresponds to the `%` operator in most programming languages, indicating the remainder after division. For example, $5 \bmod 3 = 2$ and $17 \bmod 4 = 1$.

Output Specification

Output Q lines, where the i-th line contains the earliest time (in seconds) you can finish all of the homework assignments in your to-do list after the i-th update.

Sample Input 1

```
6
A 3 3
A 2 0
A 999996 999995
D 999991
A 1000000 999994
D 999992
```

Sample Input 1 (Unencrypted)

```
6
A 3 3
A 7 5
A 4 3
D 1
A 8 2
D 2
```

Output for Sample Input 1

```
5
11
13
11
13
9
```

Explanation of Output for Sample Input 1

The unencrypted sample input is provided for ease of reference.

After the first update, we can start the first assignment at the beginning of second 3 and finish at the end of second 5 (interval $[3, 5]$).

After the second update, we can do the first assignment over the interval $[3, 5]$ and the second assignment over the interval $[7, 11]$.

After the third update, we can do the first assignment over the interval $[3, 5]$, the third assignment over the interval $[6, 8]$, and then the second assignment over the interval $[9, 13]$.

After the fourth update, we can do the third assignment over the interval $[4, 6]$ and the second assignment over the interval $[7, 11]$.

After the fifth update, we can do the third assignment over the interval $[4, 6]$, the second assignment over the interval $[7, 11]$, and the fourth assignment over the interval $[12, 13]$.

After the sixth update, we can do the third assignment over the interval $[4, 6]$ and the fourth assignment over the interval $[8, 9]$.

Sample Input 2

```
2
A 1000000 1000000
A 4 4
```

Sample Input 2 (Unencrypted)

```
2
A 1000000 1000000
A 1000000 1000000
```

Output for Sample Input 2

```
1999999
2999999
```

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

Problème S5 : Liste de tâches

Problem Description
Votre liste de tâches est vide... mais pas pour longtemps ! Au cours des prochaines secondes, vous devrez gérer les mises à jour Q de votre liste de tâches.

Pour le premier type de mise à jour, vous devrez ajouter une nouvelle tâche à votre liste. Cette tâche vous sera attribuée au début de la seconde s et prendra t secondes à effectuer ($1 \leq s, t \leq 10^6$). Pour le deuxième type de mise à jour, vous devrez retirer la $i^{\text{ième}}$ tâche qui a été ajoutée à votre liste.

Après chaque mise à jour, vous vous demandez dans quel délai vous pourrez terminer les tâches de votre liste. Vous ne pouvez travailler que sur une seule tâche à la fois et vous devez la terminer avant de passer à une autre tâche.

Précisions par rapport aux données d'entrée
La première ligne de données d'entrée contient un entier, Q.

Les lignes de données Q suivantes contiennent chacune une ligne commençant par un caractère `A` ou `D`. Une ligne commençant par un caractère `A` représente le premier type de mise à jour et se termine par deux entiers **chiffrés***, chacun étant séparé par des espaces : s' et t'. Une ligne commençant par un caractère `D` représente le deuxième type de mise à jour et se termine par un entier chiffré : i'. Il est garanti qu'au moins i tâches ont été ajoutées et que la $i^{\text{ième}}$ tâche ajoutée n'a pas encore été supprimée.

Il est certain qu'après chaque mise à jour, au moins une tâche figure sur votre liste.

Le tableau suivant détaille la répartition des 15 points disponibles.

Points	Bornes des données Q	Restrictions additionnelles
2	$1 \leq Q \leq 3\,000$	Aucune
6	$1 \leq Q \leq 10^6$	Seules les mises à jour du premier type sont effectuées.
7	$1 \leq Q \leq 10^6$	Aucune

*Notez que les données d'entrée de ce problème sont chiffrées. Pour les déchiffrer et obtenir les valeurs réelles des données s, t et i, vous pouvez utiliser les formules suivantes.

$$s = (s' + \text{ans}) \bmod (10^6 + 3) \quad t = (t' + \text{ans}) \bmod (10^6 + 3) \quad i = (i' + \text{ans}) \bmod (10^6 + 3)$$

Ici, ans représente la réponse après la mise à jour précédente et équivaut à 0 avant toute mise à jour. Il peut également être utile de noter que mod correspond à l'opérateur modulo, `%`, dans la plupart des langages de programmation, indiquant le résultat après la division. Par exemple : 5 mod 3 = 2 et 17 mod 4 = 1.

Précisions par rapport aux données de sortie
Les données de sortie devraient contenir les lignes Q, la $i^{\text{ième}}$ ligne contenant le délai le plus court (en secondes) dans lequel vous pouvez terminer toutes les tâches de votre liste après la $i^{\text{ième}}$ mise à jour.

Données d'entrée d'un 1er exemple

```
6
A 3 3
A 2 0
A 999996 999995
D 999991
A 1000000 999994
D 999992
```

Données d'entrée d'un 1er exemple (non chiffrées)

```
6
A 3 3
A 7 5
A 4 3
D 1
A 8 2
D 2
```

Données de sortie du 1er exemple

```
5
11
13
11
13
9
```

Justification des données de sortie du 1er exemple
Cet exemple de données d'entrée non chiffrées est fourni à titre de référence.

Après la première mise à jour, il est possible de commencer la première tâche au début de la seconde 3 et de la terminer à la fin de la seconde 5 (intervalle $[3, 5]$).

Après la deuxième mise à jour, il est possible d'effectuer la première tâche au cours de l'intervalle $[3, 5]$ et la deuxième tâche au cours de l'intervalle $[7, 11]$.

Après la troisième mise à jour, il est possible d'effectuer la première tâche au cours de l'intervalle $[3, 5]$, la troisième tâche au cours de l'intervalle $[6, 8]$, puis la deuxième tâche au cours de l'intervalle $[9, 13]$.

Après la quatrième mise à jour, il est possible d'effectuer la troisième tâche au cours de l'intervalle $[4, 6]$ et la deuxième tâche au cours de l'intervalle $[7, 11]$.

Après la cinquième mise à jour, il est possible d'effectuer la troisième tâche au cours de l'intervalle $[4, 6]$, la deuxième tâche au cours de l'intervalle $[7, 11]$, puis la quatrième tâche au cours de l'intervalle $[12, 13]$.

Après la sixième mise à jour, il est possible d'effectuer la troisième tâche au cours de l'intervalle $[4, 6]$ et la quatrième tâche au cours de l'intervalle $[8, 9]$.

Données d'entrée d'un 2ᵉ exemple

```
2
A 1000000 1000000
A 4 4
```

Données d'entrée d'un 2ᵉ exemple (non chiffrées)

```
2
A 1000000 1000000
A 1000000 1000000
```

Données de sortie du 2ᵉ exemple

```
1999999
2999999
```