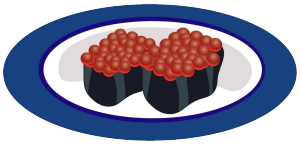
Problem J1: Conveyor Belt Sushi

Problem Description

There is a new conveyor belt sushi restaurant in town. Plates of sushi travel around the restaurant on a raised conveyor belt and customers choose what to eat by removing plates.

Each red plate of sushi costs $3, each green plate of sushi costs $4, and each blue plate of sushi costs $5.





Your job is to determine the cost of a meal, given the number of plates of each colour chosen by a customer.

Input Specification

The first line of input contains a non-negative integer, R, representing the number of red plates chosen. The second line contains a non-negative integer, G, representing the number of green plates chosen. The third line contains a non-negative integer, B, representing the number of blue plates chosen.

Output Specification

Output the non-negative integer, C, which is the cost of the meal in dollars.

Sample Input

```
0
2
4
```

Output for Sample Input

```
28
```

Explanation of Output for Sample Input

This customer chose 0 red plates, 2 green plates, and 4 blue plates. Therefore, the cost of the meal in dollars is $0 \times 3 + 2 \times 4 + 4 \times 5 = 28$.

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

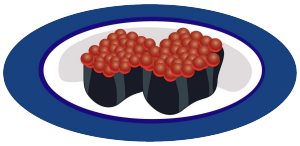
Problème J1: Sushi sur tapis roulant

Énoncé du problème

Un restaurant de sushis sur tapis roulant a récemment ouvert ses portes. Dans ce restaurant, les clients peuvent sélectionner leurs sushis en prenant des assiettes qui circulent sur un tapis roulant surélevé.

Chaque assiette rouge de sushi coûte 3 $, chaque assiette verte de sushi coûte 4 $ et chaque assiette bleue de sushi coûte 5 $.





Votre tâche consiste à déterminer le coût d'un repas en fonction du nombre d'assiettes de chaque couleur sélectionnées par un client.

Précisions par rapport aux données d'entrée

La première ligne des données d'entrée contient un entier non négatif, R, représentant le nombre d'assiettes rouges sélectionnées. La deuxième ligne contient un entier non négatif, G, représentant le nombre d'assiettes vertes sélectionnées. La troisième ligne contient un entier non négatif, B, représentant le nombre d'assiettes bleues sélectionnées.

Précisions par rapport aux données de sortie

Les données de sortie devraient afficher un seul entier non négatif C, représentant le coût du repas en dollars.

Exemple de données d'entrée

```
0
2
4
```

Exemple de données de sortie

```
28
```

Justification des données de sortie

Ce client a sélectionné 0 assiette rouge, 2 assiettes vertes et 4 assiettes bleues. Donc, le coût du repas, en dollars, est égal à $0 \times 3 + 2 \times 4 + 4 \times 5 = 28$.

English version appears before the French version

Problem J2: Dusa And The Yobis

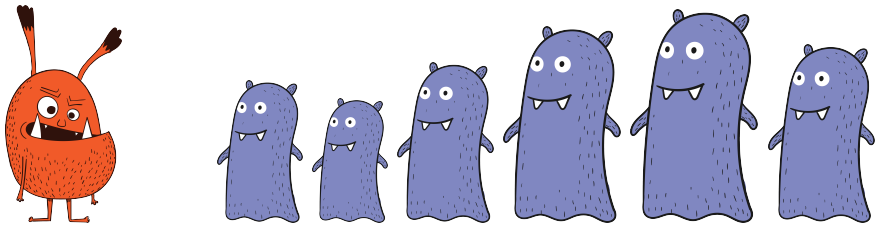
Problem Description

Dusa eats Yobis, but only Yobis of a certain size.

If Dusa encounters a Yobi that is smaller than itself, it eats the Yobi, and absorbs its size. For example, if Dusa is of size 10 and it encounters a Yobi of size 6, Dusa eats the Yobi and expands to size $10 + 6 = 16$.

If Dusa encounters a Yobi that is the same size as itself or larger, Dusa runs away without eating the Yobi.

Dusa is currently facing a line of Yobis and will encounter them in order. Dusa is guaranteed to eventually encounter a Yobi that causes it to run away. Your job is to determine Dusa's size when this happens.



Input Specification

The first line of input contains a positive integer, D, representing Dusa's starting size.

The remaining lines of input contain positive integers representing the sizes of the Yobis in order.

Output Specification

Output the positive integer, R, which is Dusa's size when it eventually runs away.

Sample Input 1

```
5
3
2
9
20
22
14
```

Output for Sample Input 1

```
19
```

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

Explanation of Output for Sample Input 1
Dusa is large enough to eat the Yobi of size 3. This brings Dusa's size to 8. Dusa is large enough to eat the Yobi of size 2. This brings Dusa's size to 10. Dusa is large enough to eat the Yobi of size 9. This brings Dusa's size to 19. The Yobi of size 20 causes Dusa to run away.

Sample Input 2

```
10
10
3
5
13
```

Output for Sample Input 2

```
10
```

Explanation of Output for Sample Input 2
The Yobi of size 10 causes Dusa to run away, leaving its size unchanged.

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

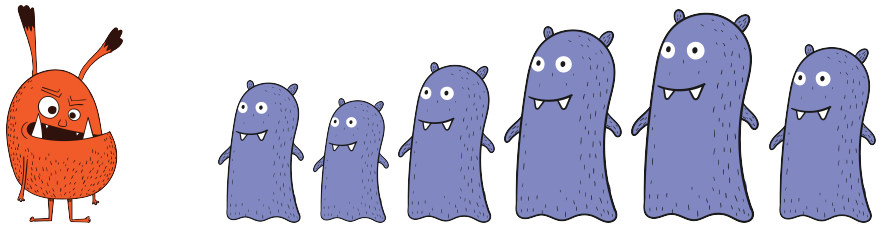
Problème J2 : Dusa et les Yobis

Problem Description
Dusa se nourrit de Yobis, mais seulement de ceux d'une taille spécifique.

Lorsque Dusa croise un Yobi plus petit que lui, il le mange et en absorbe la taille. Par exemple, si Dusa a une taille de 10 et qu'il tombe sur un Yobi de taille 6, il mange ce dernier et sa taille augmente à $10 + 6 = 16$.

Si Dusa croise un Yobi de la même taille que lui ou plus grand, Dusa s'enfuit sans manger le Yobi.

Dusa est actuellement confronté à une série de Yobis qu'il va croiser de manière successive. Il est certain qu'il finira par tomber sur un Yobi qui le poussera à la fuite. Votre tâche est de déterminer quelle sera la taille de Dusa au moment de sa fuite.



Précisions par rapport aux données d'entrée
La première ligne des données d'entrée contient un entier strictement positif D, représentant la taille de départ de Dusa.

Les lignes restantes des données d'entrée contiennent des entiers strictement positifs représentant les tailles des Yobis en ordre.

Précisions par rapport aux données de sortie
Les données de sortie devraient afficher un entier strictement positif R, représentant la taille de Dusa au moment où il prend la fuite.

Données d'entrée d'un 1er exemple
```
5
3
2
9
20
22
14
```

English version appears before the French version

Données de sortie du 1er exemple

```
19
```

Justification des données de sortie du 1er exemple

Dusa est assez grand pour manger le Yobi de taille 3, ce qui augmente sa taille à 8. Dusa est assez grand pour manger le Yobi de taille 2, ce qui augmente sa taille à 10. Dusa est assez grand pour manger le Yobi de taille 9, ce qui augmente sa taille à 19. Cependant, confronté à un Yobi de taille 20, Dusa prend la fuite.

Données d'entrée du 2e exemple

```
10
10
3
5
13
```

Données de sortie du 2e exemple

```
10
```

Justification des données de sortie du 2e exemple

Confronté à un Yobi de taille 10, Dusa s'enfuit et sa taille reste donc inchangée.

English version appears before the French version

Problem J3: Bronze Count

Problem Description

After completing a competition, you are struck with curiosity. How many participants were awarded bronze level?

Gold level is awarded to all participants who achieve the highest score. Silver level is awarded to all participants who achieve the second highest score. Bronze level is awarded to all participants who achieve the third highest score.

Given a list of all the scores, your job is to determine the score required for bronze level and how many participants achieved this score.

Input Specification

The first line of input contains a positive integer, N, representing the number of participants.

Each of the next N lines of input contain a single integer representing a participant's score.

Each score is between 0 and 75 (inclusive) and there will be at least three distinct scores.

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

Marks	Description	Bound
6	The scores are distinct and the number of participants is small.	$N \leq 50$
7	The scores might not be distinct and the number of participants is small.	$N \leq 50$
2	The scores might not be distinct and the number of participants could be large.	$N \leq 250\,000$

Output Specification

Output a non-negative integer, S, and a positive integer, P, separated by a single space, where S is the score required for bronze level and P is how many participants achieved this score.

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

Sample Input 1

```
4
70
62
58
73
```

Output for Sample Input 1

```
62 1
```

Explanation of Output for Sample Input 1

The score required for bronze level is 62 and one participant achieved this score.

Sample Input 2

```
8
75
70
60
70
70
60
75
70
```

Output for Sample Input 2

```
60 2
```

Explanation of Output for Sample Input 2

The score required for bronze level is 60 and two participants achieved this score.

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

Problème J3 : Compte de bronze

Énoncé du problème

Après la fin d'une compétition, une question vous intrigue : combien de personnes participantes se sont vu attribuer le niveau bronze ?

Le niveau or est décerné à toutes les personnes participantes qui ont obtenu le score le plus élevé. Le niveau argent est décerné à toutes les personnes participantes qui ont obtenu le deuxième meilleur score. Le niveau bronze est attribué à toutes les personnes participantes qui obtiennent le troisième meilleur score.

Étant donné une liste de tous les scores, votre tâche consiste à déterminer le score nécessaire pour atteindre le niveau bronze et le nombre de personnes participantes qui ont réussi à l'obtenir.

Précisions par rapport aux données d'entrée

La première ligne des données d'entrée contient un entier strictement positif N, représentant le nombre de personnes participantes.

Chacune des N lignes suivantes d'entrée contient un seul entier représentant le score d'une personne participante.

Chaque score est un entier de 0 à 75 (inclusivement) et il doit y avoir au moins trois scores distincts.

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

Points	**Description**	**Bornes**
6	Les scores sont distincts et il y a peu de personnes participantes.	$N \leq 50$
7	Les scores ne sont pas nécessairement distincts et il y a peu de personnes participantes.	$N \leq 50$
2	Les scores ne sont pas nécessairement distincts et le nombre de personnes participantes peut être élevé.	$N \leq 250\,000$

English version appears before the French version

Précisions par rapport aux données de sortie
Les données de sortie devraient afficher un entier non négatif S (le score requis pour atteindre le niveau bronze) et un entier strictement positif P (le nombre de personnes participantes ayant obtenu ce score), les deux étant séparés par un espace.

Données d'entrée d'un 1er exemple
```
4
70
62
58
73
```

Données de sortie du 1er exemple
```
62 1
```

Justification des données de sortie du 1er exemple
Le score requis pour atteindre le niveau bronze est 62 et une seule personne participante a obtenu ce score.

Données d'entrée d'un 2e exemple
```
8
75
70
60
70
70
60
75
70
```

Données de sortie du 2e exemple
```
60 2
```

Justification des données de sortie du 2e exemple
Le score requis pour atteindre le niveau bronze est 60 et deux personnes participantes ont obtenu ce score.

English version appears before the French version

Problem J4: Troublesome Keys

Problem Description
As Alex is typing, their keyboard is acting strangely. Two letter keys are causing trouble:

- One letter key displays the same wrong letter each time it is pressed. Alex calls this key the *silly key*. Oddly, Alex never actually tries to type the wrong letter displayed by the silly key.
- Another letter key doesn't display anything when it is pressed. Alex calls this key the *quiet key*.

Alex presses the silly key at least once but they don't necessarily press the quiet key.

Your job is to determine the troublesome keys and the wrong letter that is displayed. Luckily, this is possible because Alex never presses the silly key immediately after pressing the quiet key and Alex never presses the quiet key immediately after pressing the silly key.

Input Specification
There will be two lines of input. The first line of input represents the N keys Alex presses on the keyboard. The second line of input represents the letters displayed on the screen.

Both lines of input will only contain lowercase letters of the alphabet.

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

Marks	Description	Bound
3	The quiet key is not pressed. A small number of keys are pressed.	$N \leq 50$
3	The first troublesome key pressed is the silly key. A small number of keys are pressed.	$N \leq 50$
5	The first troublesome key pressed may be the silly key or the quiet key. A small number of keys are pressed.	$N \leq 50$
4	The first troublesome key pressed may be the silly key or the quiet key. A large number of keys are pressed.	$N \leq 500\,000$

Output Specification
There will be two lines of output.

On the first line, output the letter corresponding to the silly key and the wrong letter displayed on the screen when it is pressed, separated by a single space.

On the second line, output the letter corresponding to the quiet key if it is pressed. Output the dash character (`-`) if the quiet key is not pressed.

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

Sample Input 1
```
forloops
fxrlxxps
```

Output for Sample Input 1
```
o x
-
```

Explanation of Sample Output 1
The letter corresponding to the silly key was the letter `o`. Each time it was pressed, the wrong letter `x` was displayed. The quiet key was not pressed.

Sample Input 2
```
forloops
fxrlxxp
```

Output for Sample Input 2
```
o x
s
```

Explanation of Sample Output 2
The letter corresponding to the silly key was the letter `o`. Each time it was pressed, the wrong letter `x` was displayed. The quiet key corresponds to the letter `s` which was not displayed.

Sample Input 3
```
forloops
frlpz
```

Output for Sample Input 3
```
s z
o
```

Explanation of Sample Output 3
The letter corresponding to the silly key was the letter `s`. Each time it was pressed, the wrong letter `z` was displayed. The quiet key corresponds to the letter `o` which was not displayed.

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

Problème J4 : Touches problématiques

Énoncé du problème

Alors qu'Alex tape sur son clavier d'ordinateur, son clavier se comporte étrangement. Deux touches de lettres posent problème :

— Une touche affiche la même lettre erronée à chaque fois qu'elle est frappée. Alex a surnommé cette touche la *touche absurde*. Curieusement, Alex ne cherche jamais à taper la lettre erronée affichée par la touche absurde.
— Une autre touche n'affiche rien lorsqu'elle est frappée. Alex a surnommé cette touche la *touche silencieuse*.

Alex frappe la touche absurde au moins une fois, mais ne frappe pas nécessairement la touche silencieuse.

Votre tâche consiste à déterminer quelles sont les touches problématiques et la lettre erronée qu'affiche la touche absurde. Heureusement, cela est réalisable car Alex ne frappe jamais la touche absurde immédiatement après avoir frappé la touche silencieuse et ne frappe jamais la touche silencieuse immédiatement après avoir frappé la touche absurde.

Précisions par rapport aux données d'entrée

Les données d'entrée contiennent deux lignes. La première ligne des données d'entrée contient les N touches qu'Alex frappe sur le clavier. La seconde ligne des données d'entrée contient les lettres qui s'affichent à l'écran.

Les deux lignes des données d'entrée ne doivent contenir que des lettres minuscules de l'alphabet.

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

Points	**Description**	**Bornes**
3	La touche silencieuse n'est pas frappée. Un petit nombre de touches sont frappées.	$N \leq 50$
3	La première touche problématique frappée est la touche absurde. Un petit nombre de touches sont frappées.	$N \leq 50$
5	La première touche problématique frappée peut être la touche absurde ou la touche silencieuse. Un petit nombre de touches sont frappées.	$N \leq 50$
4	La première touche problématique frappée peut être la touche absurde ou la touche silencieuse. Un grand nombre de touches sont frappées.	$N \leq 500\,000$

English version appears before the French version

Précisions par rapport aux données de sortie
Les données de sortie devraient contenir deux lignes.

La première ligne des données de sortie devrait afficher la lettre correspondant à la touche absurde et la lettre erronée qui s'affiche à l'écran lorsqu'elle est frappée, ces deux lettres étant séparées par un espace.

La seconde ligne des données de sortie devrait afficher la lettre correspondant à la touche silencieuse si elle a été frappée. Si la touche silencieuse n'a pas été frappée, alors cette ligne devrait afficher un tiret (`-`).

Données d'entrée d'un 1er exemple
```
forloops
fxrlxxps
```

Données de sortie du 1er exemple
```
o x
-
```

Justification des données de sortie du 1er exemple
La lettre correspondant à la touche absurde était la lettre `o`. Chaque fois qu'elle a été frappée, la lettre erronée `x` s'affichait. La touche silencieuse n'a pas été frappée.

Données d'entrée d'un 2e exemple
```
forloops
fxrlxxp
```

Données de sortie du 2e exemple
```
o x
s
```

Justification des données de sortie du 2e exemple
La lettre correspondant à la touche absurde était la lettre `o`. Chaque fois qu'elle a été frappée, la lettre erronée `x` s'affichait. La touche silencieuse correspond à la lettre `s`, qui n'a pas été affichée.

Données d'entrée d'un 3e exemple
```
forloops
frlpz
```

Données de sortie du 3e exemple
```
s z
o
```

English version appears before the French version

Justification des données de sortie du 3^{e} exemple
La lettre correspondant à la touche absurde était la lettre `s`. Chaque fois qu'elle a été frappée, la lettre erronée `z` s'affichait. La touche silencieuse correspond à la lettre `o`, qui n'a pas été affichée.

English version appears before the French version

Problem J5: Harvest Waterloo

Problem Description

There is a wildly popular new harvest simulation game called *Harvest Waterloo*. The game is played on a rectangular pumpkin patch which contains bales of hay and pumpkins of different sizes. To begin the game, a farmer is placed at the location of a pumpkin.

The farmer harvests all pumpkins they can reach by moving left, right, up, and down throughout the patch. The farmer cannot move diagonally. The farmer can also not move through a bale of hay nor move outside of the patch.

Your job is to determine the total value of all the pumpkins harvested by the farmer. A small pumpkin is worth \$1, a medium pumpkin is worth \$5, and a large pumpkin is worth \$10 dollars.

Input Specification

The first line of input is an integer $R > 0$ which is the number of rows within the patch.

The second line of input is an integer $C > 0$ which is the number of columns within the patch.

The next R lines describe the patch. Each line will contain C characters and each character will either represent a pumpkin size or a bale of hay: `S` for a small pumpkin, `M` for a medium pumpkin, `L` for a large pumpkin, or `*` for a bale of hay.

The next line of input is an integer A where $0 \leq A < R$, and the last line of input is an integer B where $0 \leq B < C$. Row A and column B is the starting location of the farmer and the top-left corner of the patch is row 0 and column 0.

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

Marks	Description	Bound
1	The patch is small and there are no bales of hay.	$R \times C \leq 100$
4	The patch is small and the bales of hay divide the entire patch into rectangular areas.	$R \times C \leq 100$
5	The patch is small and the bales of hay can be anywhere.	$R \times C \leq 100$
5	The patch is large and the bales of hay can be anywhere.	$R \times C \leq 100\,000$

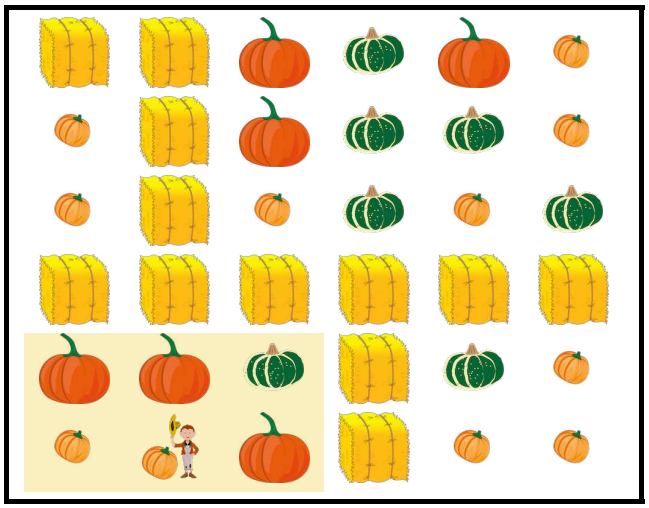
Output Specification

Output the integer, V, which is the total value in dollars of all the pumpkins harvested by the farmer.

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

Sample Input 1

```
6
6
**LMLS
S*LMMS
S*SMSM
******
LLM*MS
SSL*SS
5
1
```



Output for Sample Input 1

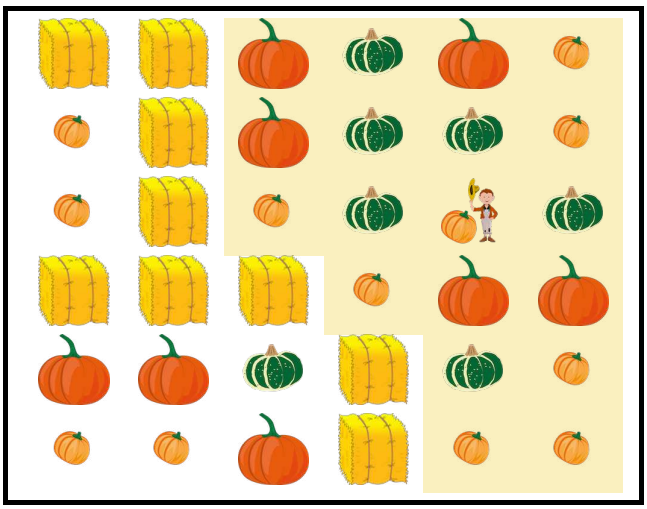
```
37
```

Explanation of Output for Sample Input 1

Starting at row 5 and column 1, the farmer can reach the 6 pumpkins in the highlighted area. They harvest 2 small pumpkins, 1 medium pumpkin, and 3 large pumpkins. The total value in dollars of this harvest is $2 \times 1 + 1 \times 5 + 3 \times 10 = 37$.

Sample Input 2

```
6
6
**LMLS
S*LMMS
S*SMSM
***SLL
LLM*MS
SSL*SS
2
4
```



Output for Sample Input 2

```
88
```

Explanation of Output for Sample Input 2

Starting at row 2 and column 4, the farmer can reach the 19 pumpkins in the highlighted area. They harvest 8 small pumpkins, 6 medium pumpkin, and 5 large pumpkins. The total value in dollars of this harvest is $8 \times 1 + 6 \times 5 + 5 \times 10 = 88$.

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

Problème J5 : Récolte Waterloo

Énoncé du problème
Le jeu *Récolte Waterloo* est un nouveau jeu de simulation très populaire. Le jeu se déroule dans un champ rectangulaire qui contient des bottes de foin et des citrouilles de différentes tailles. Au début du jeu, un fermier est positionné là où se trouve une citrouille.

Le fermier récolte toutes les citrouilles qu'il peut atteindre en se déplaçant vers la gauche, la droite, en haut ou en bas dans le champ. Il ne peut se déplacer en diagonale, traverser une botte de foin, ni sortir des limites du champ.

Votre tâche consiste à déterminer la valeur totale des citrouilles que le fermier a récoltées. Une petite citrouille vaut 1 $, une citrouille moyenne vaut 5 $ et une grande citrouille vaut 10 $.

Précisions par rapport aux données d'entrée
La première ligne des données d'entrée contient un entier $R > 0$, représentant le nombre de rangées dans le champ.

La deuxième ligne des données d'entrée contient un entier $C > 0$, représentant le nombre de colonnes dans le champ.

Les R lignes suivantes décrivent le champ. Chacune de ces lignes contient C caractères et chaque caractère représente soit une taille de citrouille, soit une botte de foin : `S` pour une petite citrouille, `M` pour une citrouille moyenne, `L` pour une grande citrouille ou `*` pour une botte de foin.

La ligne suivante des données d'entrée est un entier A ($0 \leq A < R$) et la dernière ligne des données d'entrée est un entier B ($0 \leq B < C$). La rangée A et la colonne B représentent l'emplacement initial du fermier ; le coin supérieur gauche du champ correspond à la rangée 0 et à la colonne 0.

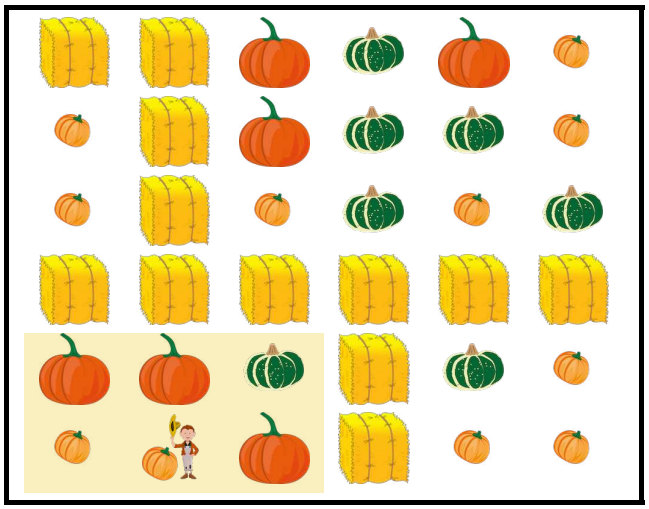
Le tableau ci-dessous détaille la répartition des 15 points disponibles.

Points	**Description**	**Bornes**
1	Le champ est petit et il n'y a pas de bottes de foin.	$R \times C \leq 100$
4	Le champ est petit et les bottes de foin divisent le champ en régions rectangulaires.	$R \times C \leq 100$
5	Le champ est petit et les bottes de foin peuvent se trouver n'importe où.	$R \times C \leq 100$
5	Le champ est grand et les bottes de foin peuvent se trouver n'importe où.	$R \times C \leq 100\,000$

English version appears before the French version

Précisions par rapport aux données de sortie
Les données de sortie devraient afficher un entier V, représentant la valeur totale, en dollars, des citrouilles que le fermier a recoltées.

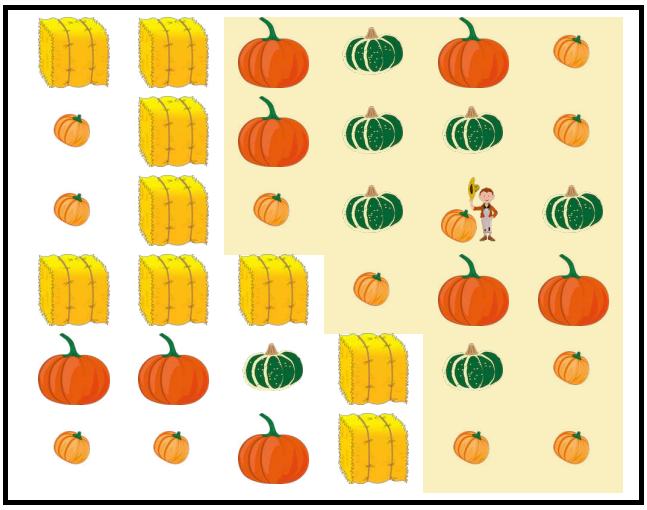
Données d'entrée d'un 1er exemple
```
6
6
**LMLS
S*LMMS
S*SMSM
******
LLM*MS
SSL*SS
5
1
```



Données de sortie du 1er exemple
```
37
```

Justification des données de sortie du 1er exemple
En commençant à la rangée 5 et à la colonne 1, le fermier peut atteindre les 6 citrouilles dans la région surlignée. Il récolte 2 petites citrouilles, 1 citrouille moyenne et 3 grandes citrouilles. La valeur totale, en dollars, de cette récolte est égale à $2 \times 1 + 1 \times 5 + 3 \times 10 = 37$.

Données d'entrée d'un 2e exemple
```
6
6
**LMLS
S*LMMS
S*SMSM
***SLL
LLM*MS
SSL*SS
2
4
```



Données de sortie du 2e exemple
```
88
```

Justification des données de sortie du 2e exemple
En commençant à la rangée 2 et à la colonne 4, le fermier peut atteindre les 19 citrouilles dans la région surlignée. Il récolte 8 petites citrouilles, 6 citrouilles moyennes et 5 grandes citrouilles. La valeur totale, en dollars, de cette récolte est égale à $8 \times 1 + 6 \times 5 + 5 \times 10 = 88$.

English version appears before the French version