## A) Great Circle Routes

### Problem Description

It is well known that the shortest path between two points on the surface of the earth (assumed to be a perfect sphere of radius 6400 km) is a great circle route. To get this, draw the circle (on the surface of the earth) that goes through the two points which is centred at the centre of the earth. The shorter arc of the circle between the two points is the shortest distance. In the figure below, the solid circle represents the earth (assumed to be a sphere), with centre O. There are two points X and Y on the surface (whose positions can be specified by a latitude and longitude). The dotted line represents a circle on the surface of the earth, with centre O. The length of the arc of the circle between X and Y is the great circle distance, which is the shortest distance between the two points while travelling on the surface.

A traveler wishes to accomplish a complex itinerary, going from point to point. Each point is specified by its latitude in degrees north or south, and its longitude in degrees East or West. The objective is to calculate the total distance travelled by the traveler as he goes from point to point in order, if he travels by the shortest distance between any two points.

### Constraints

N<=10

### Input Format

Line 1 contains N, the number of points the traveler wishes to travel between.

The next N lines each contain two comma separated strings specifying the latitude and longitude of the point in degrees. The latitude string consists of a non-negative integer (representing the latitude of the point) followed by N or S to indicate north or south. (A point on the equator will be represented by 0N). The longitude string will consist of a non-negative integer representing the longitude of the point in degrees, followed by E or W to specify East or West. For longitudes where it does not matter (such as 0 and 180) the longitude will always be specified as E. The north and south poles will be deemed to have 0 longitude

### Output

The output should be one line that specifies the distance travelled by the traveler to the nearest integer. The traveler will travel from one point to the next in order.

### Explanation

**Example 1**

Input

3

0N,1W

0N,179E

90N,0E

Output

30159

Explanation

There are 3 points. The first point is one on the equator, and 1 degree West. The second point is also on the equator, and 179 degrees East. The third point is the north pole.

The great circle passing through the first two points is the equator (as its centre is the centre of the earth). As the difference in longitudes (taking the fact that one is E and one is W) is 180 degrees. Hence this distance is half the circumference of the earth, or 0.5 \* 2\* π \*6400=6400 π.

To find the distance between the second and the third point, we note that the longitude 179 degrees East passes through the north pole and is a great circle. The angle subtended at the centre of the earth by the two points is 90 degrees, and hence the distance is (1/4) of the circumference of the circle, or (0.25)\*2\*π\*6400, or 3200π.

The total distance travelled is 9600π or 30159 (to the nearest integer). This is the output.

**Example 2**

Input

3

0N,47E

45N,47E

45S,47E

Output

15080

Explanation

As the first and second points are on the same longitude, the longitude is the great circle that passes through them. The angle subtended at the centre of the earth is 45 degrees, as that is the difference in the latitudes. Hence the great circle distance is (1/8) of the circumference of the circle, or (0.125\*2\*π\*6400)=1600π

The second and third points are also on the same longitude, and the angle subtended at the centre of the earth is 90 degrees. Hence the distance between the points is (1/4) of the circumference of the circle. The distance is (0.25)\*2\*π\*6400, or 3200π.

The total distance travelled is 4800π, or 15080. This is the output.

**B) Count Bits**

**Problem Description**

Given a sequence of distinct numbers a1, a2, ….. an, an inversion occurs if there are indices i<j such that ai > aj .

For example, in the sequence 2 1 4 3 there are 2 inversions (2 1) and (4 3).

The input will be a main sequence of N positive integers. From this sequence, a Derived derived sequence will be obtained using the following rule. The output is the number of inversions in the derived sequence.

**Rule for forming derived sequence**

The derived sequence is formed by counting the number of 1s bits in the binary representation of the corresponding number in the input sequence.

Thus, if the input is 3,4,8, the binary representation of the numbers are 11,100 and 1000. The derived sequence is the number of 1sin the binary representation of the numbers in the input sequence, and is 2,1,1

**Constraints**

N <= 50

Integers in sequence <= 107

**Input Format**

The first line of the input will have a single integer, which will give N.

The next line will consist of a comma separated string of N integers, which is the main sequence

**Output**

The number of inversions in the derived sequence formed from the main sequence.

**Explanation**

**Example 1**

Input

5

55, 53, 88, 27, 33

Output

8

Explanation

The number of integers is 5, as specified in the first line. The given sequence is 55, 53, 88, 27, 33.

The binary representation is 110111, 110101, 1011000, 11011, 100001and 100001 . The derived sequence is 5,4,3,4,2, 4,3,4,2 (corresponding to the number of 1s bits). The number of inversions in this is 8, namely (5,4),(5,3),(5,4),(5,2),(4,3),(4,2),(3,2),(4,2). Hence the output is 8.

**Example 2**

Input

8

120,21,47,64,72,35,18,98

Output

15

Explanation

The number of integers is 8. The given sequence is 120,21,47,64,72,35,18,98. The corresponding binary sequence is 1111000,10101,101111,1000000,1001000,100011, 10010,1100010. The derived sequence (number of 1s) is 4,3,5,1,2,3,2,3. The number of inversions is 15, namely (4,3),(4,1),(4,2),(4,3),(4,2),(4,3),(3,1),(3,2),(3,2),(5,1),(5,2),(5,3), (5,2), (5,3),(3,2). Hence the output is 15.

## C) Finding Sum

### Problem Description

You are given a set of N positive integers and another integer P, where P is a small prime. You need to write a program to count the number of subsets of size (cardinality) 3, the sum of whose elements is divisible by P. Since the number K of such sets can be huge, output K modulo 10007 1009 (the remainder when K is divided by 1009)

### Constraints

N <= 500

P<50

All integers <=1000

### Input Format

First line two comma separated integers N, P

The next line contains N comma separated integers

### Output

One integer giving the number of subsets the sum of whose elements is divisible by P. Give result modulo 1009

### Explanation

**Example 1**

Input

4,5

5,10,15,20

Output

4

Explanation

Every non empty subset of the given numbers has sum of its elements a multiple of 5. Since there are 4 subsets of size 3, the output is 4.

**Example 2**

Input

5,5

3,7,12,13,15

Output

4

Explanation

There are 4 subsets of size 3 with sum a multiple of 5: {3, 7, 15}, {12, 13, 15}, {7, 13, 15}, {3, 12, 15}, Hence the output is 4.

## D) Bug Crawl

A cube has its 6 faces numbered from 1-6 (three of its faces are shown in the diagram). There are six possible orientations on the cube (U,D,N,E,W,S) as shown in the diagram, of which only 4 are possible on any one face (for example, on face 1, N and S are impossible orientations)

An electronic remote control bug is on a face of a cube at some orientation. On your command it can crawl around the cube. The possible commands are F,B,L,R.

A cube has its 6 faces numbered from 1-6 (three of its faces are shown in the diagram). There are six possible orientations on the cube (U,D,N,E,W,S) as shown in the diagram, of which only 4 are possible on any one face (for example, on face 1, N and S are impossible orientations)

An electronic remote control bug is on a face of a cube at some orientation. On your command it can crawl around the cube. The possible commands are F,B,L,R.

Command F : it moves to the adjacent face in the same direction it is facing

.Command B : it turns around by 180° and moves forward by one face.

Command L: it turns left and moves forward by one face

Command R: it turns right and moves forward by one face.

The faces 4,5,6 are opposite to faces 1,2,3 respectively.

For example, if it has orientation U on face 1, on command L, it goes to face 5 and has orientation N, and from there on command F , it will move to face 4 and will have orientation E.

Given a sequence of commands, and the final position (face and orientation) of the bug (after executing the commands in the sequence), we need to determine the initial position (face and orientation) of the bug.

### Constraints

The length of the string of command letters <=50

### Input Format

One string of 2 characters giving the face and orientation of the bug after it executes the instructions. The first character is a number between 1 and 6, denoting the face, and the second character is the orientation (from the set {U, D, N, S, E, W}) of the bug after executing the commands

One string of command letters. This is a sequence of letters from the set of valid commands {F, B, L, R}

### Output

One string of two characters denoting the position of the bug before it executes the instructions. The first character gives the face number (1,2,3,...,6) and the second giving the orientation (E,W,U,D,N,S ) the bug was facing before it executed the instructions

### Explanation

**Example 1**

Input

1U

FFF

Output

3N

Explanation

If the bug starts at 3N, it will move to 4D, 6S and 1U if a command F is given at each position. Hence, if it starts at 3N, after 3 consecutive F commands, it will be at 1U, which is the given final position. Hence the output is 3N.

**Example 2**

Input

4W

LRB

Output

3E

Explanation

If the bug starts at 3E, after the L command, it goes to 4D, and after the R command, it goes to 2S. From there, on the B command, it goes to 4W, which is the end position. Hence the output is 3E.

Command F : it moves to the adjacent face in the same direction it is facing

.Command B : it turns around by 180° and moves forward by one face.

Command L: it turns left and moves forward by one face

Command R: it turns right and moves forward by one face.

The faces 4,5,6 are opposite to faces 1,2,3 respectively.

For example, if it has orientation U on face 1, on command L, it goes to face 5 and has orientation N, and from there on command F , it will move to face 4 and will have orientation E.

Given a sequence of commands, and the final position (face and orientation) of the bug (after executing the commands in the sequence), we need to determine the initial position (face and orientation) of the bug.

### Constraints

The length of the string of command letters <=50

### Input Format

One string of 2 characters giving the face and orientation of the bug after it executes the instructions. The first character is a number between 1 and 6, denoting the face, and the second character is the orientation (from the set {U, D, N, S, E, W}) of the bug after executing the commands

One string of command letters. This is a sequence of letters from the set of valid commands {F, B, L, R}

### Output

One string of two characters denoting the position of the bug before it executes the instructions. The first character gives the face number (1,2,3,...,6) and the second giving the orientation (E,W,U,D,N,S ) the bug was facing before it executed the instructions

### Explanation

**Example 1**

Input

1U

FFF

Output

3N

Explanation

If the bug starts at 3N, it will move to 4D, 6S and 1U if a command F is given at each position. Hence, if it starts at 3N, after 3 consecutive F commands, it will be at 1U, which is the given final position. Hence the output is 3N.

**Example 2**

Input

4W

LRB

Output

3E

Explanation

If the bug starts at 3E, after the L command, it goes to 4D, and after the R command, it goes to 2S. From there, on the B command, it goes to 4W, which is the end position. Hence the output is 3E.

## E) Building Blocks

**Problem Description**

The Mathematics teacher wanted to introduce a new competition to the students to sharpen their skills in optimization. He drew a rectangular M ×N grid on the ground and filled it with some non-negative integers on each cell of the grid. The cell named (i,j) is at the intersection of i th row and j th column. He gave the following challenge to the students:

1. On each cell of the grid, you can pile any number of cube blocks.

2. Each layer must be rectangular (with no gaps) and rest p supported completely by the immediate below layer.

3. The number of blocks on each cell should not exceed the number written on the cell on the ground.

4. In each layer, the cell above (1,1) must be covered.

The challenge is to pile up the maximum number of blocks subject to the above conditions.

For example if the bottom grid was as follows:

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 0 |

the maximum number of blocks you can pile is 6 with one layer covering all the cells from (1,1) to (3,2).

**Constraints**

1 <= M,N <= 50

Maximum value in each cell is 50

**Input Format**

The first line will contain two comma separated integers M,N giving the size of the grid, where M is the number of rows and N is the number of columns.

The next M lines will each contain comma separated N non-negative integers giving the numbers in the grid cells.

**Output**

One line containing the number of blocks that can be piled according to the rules.

**Explanation**

**Example 1:**

Input:

3,4

5,4,9,3

4,3,5,6

2,2,1,1

Output:

32

Explanation:

One example of the maximum number of blocks that could be piled on the grid is shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| 5 | 4 | 4 | 3 |
| 3 | 3 | 3 | 3 |
| 1 | 1 | 1 | 1 |

The total number of blocks is 32. Hence the output is 32.

**Example 2:**

Input

4,7

27,26,28,14,15,38,0

38,40,35,2,20,43,39

18,48,43,2,47,18,26

38,2,29,23,14,31,32

Output

242

Explanation

The number of rows is 4. The number of columns is 7. The values in the cells of the grid are

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 28 | 14 | 15 | 38 | 0 |
| 38 | 40 | 35 | 2 | 20 | 43 | 39 |
| 18 | 48 | 43 | 2 | 47 | 18 | 26 |
| 38 | 2 | 29 | 23 | 14 | 31 | 32 |

One possible maximal placing of the blocks is

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 26 | 2 | 2 | 2 | 0 |
| 27 | 26 | 26 | 2 | 2 | 2 | 0 |
| 18 | 18 | 18 | 2 | 2 | 2 | 0 |
| 2 | 2 | 2 | 2 | 2 | 2 | 0 |

As there are 242 blocks in this maximal placing, the output is 242.

## F) Array Product

### Problem Description

You are given a list of N integers and another positive integer K. Write a program to compute the number of ways in which the product P of the given N integers can be expressed as a product of K positive integers (not necessarily distinct). The order of the factors in the expression is not important. For example, 1 x 2 x 3 and 2 x 3 x 1 are not counted as different ways of expressing 6 as a product of three integers.

### Constraints

The product of the N integers <= 10^9

Each of the N integers <=5000

### Input Format

First line contains two space separated integers, N and K

The next line contains N space separated integers

### Output

One line containing the number of ways in which the product of the N integers can be expressed as a product of K positive integers

### Explanation

**Example 1**

Input

2 4

2 3

Output

2

Explanation

The product of the given integers is 6. This can be expressed as a product of 4 integers in 2 ways: 1x1x1x6, 1x1x2x3

**Example 2**

Input

2 3

4 16

Output

7

Explanation

The product is 64. This can be expressed as a product of three integers in the following ways:

1 x 1 x 64

1 x 2 x 32

1 x 4 x 16

1 x 8 x 8

2 x 2 x 16

2 x 4 x 8

4 x 4 x 4