East Central North America Regional Contest 2016

ECNA 2016

Practice Session

October 28

Problems

A Dinoscan
B Happy Trails
C Prime Driving Conditions
D Time is of the Essence

Do not open before the contest has started.

This page is intentionally left blank.

ACM International Collegiate Contest
2016 East Central Regional PRACTICE Contest
Grand Valley State University
University of Cincinnati
University of Windsor
Youngstown State University
October 28, 2016

Sponsored by IBM

Rules:

1. There are **four** problems to be completed in **90 minutes**.
2. All questions require you to read the test data from standard input and write results to standard output. You cannot use files for input or output. Additional input and output specifications can be found in the General Information Sheet.
3. When displaying results, follow the format in the Sample Output for each problem. Unless otherwise stated, all whitespace in the Sample Output consists of exactly one blank character.
4. The allowed programming languages are C, C++, Java, Python 2 and Python 3.
5. All programs will be re-compiled prior to testing with the judges' data.
6. Non-standard libraries cannot be used in your solutions. The Standard Template Library (STL) and C++ string libraries are allowed. The standard Java API is available, except for those packages that are deemed dangerous by contestant officials (e.g., that might generate a security violation).
7. Programs will be run against multiple input files, each file containing a single test case.
8. Programming style is not considered in this contest. You are free to code in whatever style you prefer. Documentation is not required.
9. All communication with the judges will be handled by the Kattis environment.
10. Judges' decisions are to be considered final. No cheating will be tolerated.

Problem A
Dinoscan

Sarah Tops is a paleontologist who specializes in the reconstruction of dinosaur skeletons. One problem which she faces is determining whether two bones mesh together correctly. She has taken scans of all the bones in her collection, so instead of physically trying to put bones together she plans to use the scans to determine the appropriate connections between bones. Below on the left are scans of the ends of two different bones:

1	1	1	0	0
1	0	0	0	0
1	1	1	0	0
1	0	0	0	0
1	1	1	0	0

Scan 1

0	0	1
1	1	1
0	0	1
1	1	1
0	0	1

Scan 2

1	1	1	**1**
1	**1**	**1**	**1**
1	1	1	**1**
1	**1**	**1**	**1**
1	1	1	**1**

Combined

In each scan, a 1 indicates bone and a 0 indicates empty space. Two bones mesh together if the two scans can be put together to form a solid rectangle of 1's with no overlap. Scan 1 above can mesh with Scan 2 (as shown above on the right), but Scan 1 cannot mesh with either scans 3 or 4 shown below:

0	0	0	1
0	1	1	1
0	0	0	1
1	1	1	1
0	0	0	1

Scan 3

0	0	1
1	1	1
0	1	1
1	1	1
0	0	1

Scan 4

Input

Input starts with a line containing three positive integers r c_1 c_2, indicating the number of rows in both scans and the number of columns in the left hand and right hand scans. Following this are r lines each containing c_1 characters – this is the left hand scan. All characters in the scan are either a '`0`' or a '`1`'. Following this are r lines each with c_2 characters specifying the right hand scan. The maximum value for r, c_1 and c_2 is 20. In all test cases, the first column of the left hand scan and the last column of the right hand scan contain all '`1`'s.

Output

Display `Yes` if the two scans can be meshed together, or `No` otherwise. After the scans are connected, no portion of the left hand scan should extend beyond the right hand scan, and vice-versa.

Sample Input 1

```
5 5 3
11100
10000
11100
10000
11100
001
111
001
111
001
```

Sample Output 1

```
Yes
```

Sample Input 2

```
5 5 3
11100
10000
11100
10000
11100
001
111
011
111
001
```

Sample Output 2

```
No
```

Problem B
Happy Trails

Your job as a forest ranger comes with many responsibilities, but one of the hardest tasks is planning and preparing new trails for hikers. When a new trail is opened to the public, the National Forest Service adds a page to their website for the trail which contains quite a bit of information. The web page includes things like the location where the trail begins, the difficulty of hiking the trail, and pictures taken from various scenic views along the trail. The hardest piece of data for you to gather is the difference in elevation between the start of the trail and the end of the trail.

You and your team use surveying equipment to calculate the slope and distance (along the trail) of each section of the trail. For example, a trail may begin completely flat for 500 meters, then switch to an 8° incline for 1000 meters, and then switch to a 2° decline for the final 500 meters. With so many new trails opening, manually calculating the elevation difference from the section descriptions has become too tedious. So, you've decided to write a program to make the process easier.

Input

Input begins with a positive integer $n \leq 100$ indicating the number of trail sections. The following n lines describe the trail sections from the beginning to the end of the trail. Each trail section is described by two integers a d ($-50 \leq a \leq 50$, $1 \leq d \leq 10000$) indicating its angle of elevation in degrees and the distance in meters along that section of the trail, respectively.

Output

Display the difference in elevation between the start of the trail and the end of the trail in meters. The difference in elevation will always be greater than 0 meters. Round answers to the hundredths place. Always print answers to two decimal places and include the leading 0 on answers between 0 and 1.

Sample Input 1	Sample Output 1
3 0 500 8 1000 -2 500	121.72

Sample Input 2	Sample Output 2
3 12 1000 -11 1200 5 250	0.73

Sample Input 3	Sample Output 3
2 -7 2000 9 4000	382.00

Problem C
Prime Driving Conditions

Meredith has just bought a new car and needs license plates. In Pennsylvania, license plates consist of three uppercase letters followed by four digits. License plates are given out in *lexicographic order*, meaning that plate AAA 0000 is first, followed by AAA 0001, AAA 0002, etc,. After AAA 9999 comes plates AAB 0000, AAB 0001, etc., and after plate AZZ 9999 comes plate BAA 0000. The very last plate is ZZZ 9999.

For most people, any old license plate will do, but not for Meredith. She insists that the 4 digit number on the plate must be a prime number. Recall that a prime number is a number > 1 whose only factors are 1 and the number itself. The first few prime numbers are

$$2, 3, 5, 7, 11, 13, 17, 19, 23, \ldots$$

So if Meredith goes to the DMV to get a set of plates, and the next available plate is JPB 1285, she'll wait until plate JPB 1289 becomes available, since 1289 is the first prime greater than or equal to 1285. If the next available plate had been JPB 9999, she would have waited for the plate JPC 0002.

Perhaps you see what we're driving at here. Starting at a given license plate, find the first plate lexicographically equal or greater that contains a prime number.

As a side note, this past January the largest prime number known to date was discovered. It is equal to $2^{74,207,281} - 1$ and contains 22,338,618 digits. A little too large to fit on a license plate.

Input

The input file contains multiple test cases. Each test case is on a single line and contains a three character string followed by a 4 digit number. The three character string consists solely of uppercase letters other than the string "ZZZ". A line containing END 0000 terminates input.

Output

For each test case, display the license plate Meredith will accept.

Sample Input 1	Sample Output 1
JPB 1285 JPB 9999 END 0000	JPB 1289 JPC 0002

This page is intentionally left blank.

Problem D
Time is of the Essence

Do you know how long one million seconds is? How about one billion seconds? Well, the answer to the first one is

11 days, 13 hours, 46 minutes and 40 seconds

Accurate, but a little over-precise. A more reasonable answer might be “11 days, 14 hours" or just “12 days" (after rounding). As another example, if I asked how long three million inches were, you could approximate it by saying either “47 miles" or “47 miles, 613 yards". This sort of imprecision is what we’re looking for in this problem!

Input

Input consists of two lines. The first line has the form

$$n\ name_1\ c_1\ name_2\ c_2\ \ldots\ name_{n-1}\ c_{n-1}\ name_n$$

Each $name_i$ is a unit of measurement and each c_i is a conversion rate, telling you how many units of type $name_{i+1}$ are in a unit of type $name_i$. For example, in the first sample input, this line tells you that there are 24 hours in one day, 60 minutes in each hour and 60 seconds in each minute. The value n indicates the total number of unit names and satisfies $2 \leq n \leq 10$. All names are alphabetic strings with no blanks and always specify the plural form of the unit (which you should use for output no matter the value is associated with the unit). All conversion values are positive integers. Following this line is a single positive integer $m \leq 2,000,000,000$ indicating an amount of the last unit (i.e., $name_n$).

Output

Display two lines. The first line should give the closest conversion of the input amount to units specified by $name_1$, rounded to the nearest integer. The second line should do the same, but be a bit more precise using the two largest units – $name_1$ and $name_2$. When rounding, always round up when the fraction is ≥ 0.5. Note that there are no commas in the output.

Sample Input 1

```
4 days 24 hours 60 minutes 60 seconds
1000000
```

Sample Output 1

```
12 days
11 days 14 hours
```

Sample Input 2

```
5 years 365 days 24 hours 60 minutes 60 seconds
1000000000
```

Sample Output 2

```
32 years
31 years 259 days
```

Sample Input 3

```
2 feet 12 inches
11
```

Sample Output 3

```
1 feet
0 feet 11 inches
```

Sample Input 4

```
4 jiffys 2 lambtailshakes 11 flashes 3 twinklings
20902
```

Sample Output 4

```
317 jiffys
316 jiffys 1 lambtailshakes
```