

A+ Computer Science Computer Science Competition Hands-On Programming Set

I. General Notes

1. Do the problems in any order you like. They do not have to be done in order from 1 to 12.
2. All problems have a value of 60 points.
3. There is no extraneous input. All input is exactly as specified in the problem. Unless specified by the problem, integer inputs will not have leading zeros. Unless otherwise specified, your program should read to the end of file.
4. Your program should not print extraneous output. Follow the form exactly as given in the problem.
5. A penalty of 5 points will be assessed each time that an incorrect solution is submitted. This penalty will only be assessed if a solution is ultimately judged as correct.

II. Point Values and Names of Problems

Number	Name
Problem 1	Howl
Problem 2	Copier
Problem 3	Pups
Problem 4	Date
Problem 5	Math
Problem 6	Word
Problem 7	Bomb
Problem 8	Rain
Problem 9	Repeat
Problem 10	Pegs
Problem 11	Treasure
Problem 12	Rat

**For more Computer Science practice tests and materials, go to
www.apluscompsci.com**

A+ Computer Science – 2018-2019 PACKET 4

1. Howl

Program Name: howl.java

Input File: none

On a full moon, one might be able to hear a wolf howl in the distance. You manage to come across a wolf howling at the moon, and you want to capture the beautiful image. Unfortunately, you left your camera at home, so you decide to just create the picture on your computer! Create the image of a wolf howling as shown below.

Input

None.

Output

The wolf art, exactly as shown below. There is an extra row of numbers added for your convenience, but the row of numbers should NOT be included in your output.

Output to Screen

A+ Computer Science – 2018-2019 PACKET 4

2. Copier

Program Name: copier.java

Input File: copier.dat

Your copier broke down last week, and you need to copy a list of numbers for a class project due tomorrow! Luckily, you can use your computer to copy the numbers for you. Given a list of numbers, each on their own row, print out the number, a space, and then another copy of the number.

Input

The first line will contain a single integer n that indicates the number of numbers to follow, each on their own row. The next n rows will each contain a single number.

Output

For each of the n rows, print out the original number and a copy of the number, with one space of separation.

Example Input File

```
3
7
3
10
```

Example Output to Screen

```
7 7
3 3
10 10
```

A+ Computer Science – 2018-2019 PACKET 4

3. Pups

Program Name: pups.java

Input File: pups.dat

Congratulations, you adopted some little puppies! Now you just need to go grab food for them at the store. Your vet tells you how many pounds of food each pup will eat before your next trip to the store, so you just need to calculate the total amount of food that you will need to buy. You also know how much food costs per pound, so you just need to make sure that you bring the right amount of money to pay for the food. Write a program that, given the number of puppies, the amount of food per puppy, and the price per pound of food, calculates the amount of money that you will need to bring.

Input

The first line will contain a single integer n that indicates the number of lines that follow. Each line will contain three non-negative numbers d , f , and p for the number of dogs, the amount of food per dog in pounds, and the price of the food per pound.

Output

For each data set, output the total amount of money required to buy the food, rounded to the nearest hundredth. Include a dollar sign before the number.

Example Input File

```
3
3 2 1
5 .16 4.54
3 3.7 3.6
```

Example Output to Screen

```
$6.00
$3.63
$39.96
```

A+ Computer Science – 2018-2019 PACKET 4

4. Date

Program Name: date.java

Input File: date.dat

You are starting to get annoyed with how your phone shows you the date, so you want to put it into a different format. Currently, your phone gives you the date in the format `Month Day, Year` (ex. `July 25, 2018`). Your goal is to shorten the format exactly like this, `MM/DD/YY`, where M is month, D is day, and Y is year (ex. `07/25/18`). Write a program that will calculate this format change.

Input

The first line will contain a single integer `n` that indicates the number of lines that follow. The next `n` lines will be in the format `Month Day, Year`. The full year integer could be any number of digits.

Output

Your output should be the same date as the input, but in the new format `MM/DD/YY`. If the month, day or year is only one digit, pad the format with a 0. If the date is not valid (the month is spelled incorrectly, the day is less than 1 or greater than 31), print `Invalid`. You do not need to check that the day fits within the month's days, just that it is not greater than 31. The year format will be the last digits of the input year.

Example Input File

```
3
February 5, 990
December 25, 12018
Decembuary 31, 2000
```

Example Output to Screen

```
02/05/90
12/25/18
Invalid
```

A+ Computer Science – 2018-2019 PACKET 4

5. Math

Program Name: math.java

Input File: math.dat

While doing your math homework, you decide that it might be easier for your computer to do the work instead of having to do them yourself with a calculator. Write a program that will calculate the answer to the equation.

Input

The first line will contain a single integer n that indicates the number of data sets that follow.

Each data set will consist of a single line containing alternating integers and operators. There will be a space between each number and operator, and the only operators will be $+$ $-$ $*$ $/$ $\%$.

Output

Output the integer value solution to the equation. The final answer will always be an integer, but it is not guaranteed that the intermediate steps will be. The equation will be solved using the default order of operations.

Example Input File

```
3
5 * 2 + 9 / 3
2 + 2 + 6 % 5
6 + 2 * 10 - 4
```

Example Output to Screen

```
13
5
22
```

A+ Computer Science – 2018-2019 PACKET 4

6. Word

Program Name: word.java

Input File: word.dat

Word searches are hard. Luckily, you're a programmer, so you can just write a program to find the words! Fortunately, the word searches that you are doing are only looking for one word, and that word is "word". Write a program that finds the number of instances of the word "word" in a word search.

Input

The first line will contain a single integer n that indicates the number of data sets that follow. Each data set will start with two integers r c with r being the number of rows and c being the number of columns, respectively. The next r lines will contain c random letters, creating a word search grid.

Output

Output the number of times that the word "word" appears in the word search. The letters can be going in any direction, including backwards.

Example Input File

```
2
4 4
word
aoah
nerd
qpjd
4 4
sowk
dyuf
asaf
diub
```

Example Output to Screen

```
2
0
```

A+ Computer Science – 2018-2019 PACKET 4

7. Bomb

Program Name: bomb.java

Input File: bomb.dat

Bombberman is having some trouble defeating his enemies, and he has asked you to help him figure out where to place his bomb in order to defeat the most enemies. In this scenario, bomberman's bombs destroy everything in the four cardinal directions to the edge of the map, except for stopping at walls that cannot be broken. Write a program that finds the best spot to place a bomb to destroy the most enemies.

Input

The first line will contain a single integer n that indicates the number of data sets that follow. Each data set will start with two integers r c representing the number of rows and columns in each map, respectively. The next r lines will consist of c characters, representing the map. # denotes a wall, . is a blank space, and @ is an enemy. The bomb can only be placed in a blank space.

Output

Output the coordinate for the bomb, in the format r , c , where r is the row number and c is the column number. The bomb will explode outward in the four cardinal directions, destroying any enemy in its path. The explosion only stops in one direction when it reaches the end of the map or when it hits a wall, in which case any enemy past that wall is safe. Assume that there will always be at least one enemy that a bomb can hit, and there will never be more than one best placement.

Example Input File

```
2
9 17
#.#.#.#.#@#.#.#.#
..@.....
#.#.#.#.#.#.#.#
.....@.....
#.#.#.#.#.#.#.#
.....@.....@..
#.#.#.#.#.#.#.#
.....@.....
#.#.#.#.#.#.#.#
5 5
#####
#...#
#.#@#
#.@.#
#####
```

Example Output to Screen

```
5, 9
3, 3
```


A+ Computer Science – 2018-2019 PACKET 4

8. Rain

Program Name: rain.java

Input File: rain.dat

In your city, there is a shortage of drinking water so the city planner is finding more ways to trap rainwater to be used for drinking. The city has varying heights of buildings, and rainwater gets trapped between the buildings. The city planner has asked you to write a program that, given the heights of the buildings, can find the largest amount of rainwater that can get trapped between the buildings.

Input

The first line will contain a single integer n that indicates the number of data sets that follow. Each data set will consist of a single line of integers, denoting the heights of the buildings. The building height can be positive or negative.

Output

Output the total amount of rainwater that can be trapped between/above the buildings of varying height. Disclaimer/Hint: It is not calculating the volume.

Example Input File

```
2
5 0 0 0 5
6 3 1 6 4 5 1 0 3 5 3 1 4
```

Example Output to Screen

```
15
24
```

A+ Computer Science – 2018-2019 PACKET 4

9. Repeat

Program Name: repeat.java

Input File: repeat.dat

Your teacher has been marking students off on their essays for repeating words and phrases when they could be exploring new syntax and using a thesaurus to spice up their writing. In order to save yourself from being called out for using repeats, you decide to create a way to check your essay for repeated substrings and find the longest one to fix. Write a program that, given a string of characters, finds the greatest repeated substring with no overlapping characters.

Input

The first line will contain a single integer n that indicates the number of data sets that follow. Each data set will consist of one line with a string of letters of unknown length.

Output

Output the integer length of the longest repeated substring, a space, and then the repeated string. If there are multiple with the same length, print out the first one to appear in the string.

Example Input File

2

thequickbrownfoxjumpedoverthelazydog

ratsarejustlikebigmicebutratsarecompletelydifferentanimals

Example Output to Screen

3 the

7 ratsare

A+ Computer Science – 2018-2019 PACKET 4

10. Pegs

Program Name: pegs.java

Input File: pegs.dat

There is a game called peg solitaire that requires the player to remove all of the pegs from the game board except for one. Pegs can be removed by moving a peg from one side of them to the opposite side, but there must be an open space to move the peg to. When the outside peg is moved, the peg that it 'hopped' over is removed. If the hops and removals are done correctly, there should be only one peg remaining at the end of the game. This game is very difficult, but you decide that you can use your computer to solve it for you! Given a peg solitaire board, write a program to tell you if it is possible to solve the game of peg solitaire.

Input

The first line will contain a single integer n that indicates the number of data sets that follow. Each data set will start with two integers r c representing the number of rows and columns in each board, respectively. The next r lines will represent the map. `.` denotes a blank space on the board and `@` is a peg. Pegs can only be moved over another peg and into the blank space next to it.

Output

If the board is solvable, print `Solvable!`

If the board is not solvable, print `Impossible.`

The board is solvable if it is possible to get down to only one peg remaining on the board.

Example Input File

```
2
7 7
. . .
. @ .
. @ @ . @ . .
. . . @ . @ .
. . . . @ . .
. . .
. . .
6 6
@ @
. .
@ . . . . @
@ . . . . @
. .
@ @
```

Example Output to Screen

```
Solvable!
Impossible.
```

A+ Computer Science – 2018-2019 PACKET 4

11. Treasure

Program Name: treasure.java

Input File: treasure.dat

After doing some digging on the beach, you happen to pull up an old treasure chest! It is full of rare artefacts worth a lot of money! Unfortunately, you can only carry so much with you back to your boat before the tide washes the treasure chest away. Write a program that will calculate the highest total item value that you can take back in one trip, given several item weights and values, and your carrying capacity.

Input

The first line will contain a single integer n that indicates the number of data sets that follow. Each data set will start with two integers i w that indicates the number items that follow and your carrying capacity, respectively. The next i lines will each have two numbers, the value of each item and its weight.

Output

Output the highest value that you can carry in one trip for each data set.

Example Input File

```
2
5 50
70 10
90 32
40 4
150 45
60 46
1 10
500000 11
```

Example Output to Screen

```
200
0
```

A+ Computer Science – 2018-2019 PACKET 4

12. Rat

Program Name: rat.java

Input File: rat.dat

In a test to find out how intelligent rats are, you assemble a three-dimensional maze for a rat to test both his speed and ability to find the shortest path. It takes the rat one second to go from space to space on the same level, and twice as long to climb to the level above or below him. You also decide to place rat traps in the maze, which the rat can pass through safely, but it takes him three seconds to do so instead of one. Moving up or down a level still takes twice as long, so moving up or down a level into a trap would take six seconds. In order to tell if the rat is finding the shortest path or the quickest path, write a program that finds both the shortest path (lowest number of spaces passed through) and the quickest path (shortest amount of time).

Input

The first line will contain a single integer n that indicates the number of data sets that follow. Each data set will start with three integers l r c representing the number of layers, rows, and columns, respectively. The next l sets of r rows will be the maze, where every r rows represents a new layer of the maze. # will be a wall, T will be a trap, and . will be a free space. S will be the start, and E will be the end of the maze. The trap spaces will have no influence on the distance for the shortest path, but it will take longer for the quickest path.

Output

The output should be in the format shown in the example output, two lines for each set representing the shortest path and the quickest path.

Example Input File

```
2
2 4 8
#####
#S.###.
##.....#
#TT####
E.....
T#####.
T#####
TT#####
2 2 2
ST
T.
TT
TE
```

Example Output to Screen

```
Shortest Path: 9 spaces
Quickest Path: 18 seconds
Shortest Path: 3 spaces
Quickest Path: 6 seconds
```