Problem Set 1

Review of AP CS Skills

File Input/Output

Competition-Style Problems

Grading Standards (Homework Category)

Standards-based Grading (percentage of indicated max points):
Outstanding – 100%, Excellent – 95%, Acceptable – 87%, Unacceptable – 75%, No

1. I can produce an externally correct solution to each of the following problems as demonstrated through test case pass rate. (5 pts per problem)
2. I can produce an internally correct solution through proper use of buffered reader, try/catch, commenting, and otherwise professional style. (10 pts)

Problem Set 1

Problem 0 - Hello, world!

Introduction

Kindness and cordiality go a long way in today's world. If you're going to build a robot to conquer the world, the least you can do is make it polite. You are to complete the method helloWorld, which takes a list of names and offers a polite and personalized greeting.

Program Input

The first line of helloWorld.in.txt will contain a positive integer T denoting the number of test cases that follow. Each test case will have the following input:

- A single line of text containing the name of the person to greet.

Example Input

2
Jesse
Taylor

Program Output

Your program should print a greeting in the form of "Hello, NAME." where NAME is tailored to the individual name.

Example Output

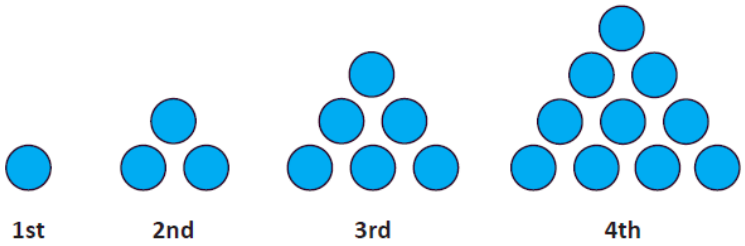
Hello, Jesse.
Hello, Taylor.

Problem Set 1

Problem 1 - Don’t Triangle Too hard

Introduction

In mathematics the triangular numbers are generated by summing the natural numbers (positive integers). The first triangular number is 1. The second triangular number is the sum of the first two positive integers: 1 + 2 = 3. The third triangular number is the sum of the first three natural numbers: 1 + 2 + 3 = 6. A visual representation helps explain why the numbers are known as the triangular numbers. You are tasked with writing a method, triangularNumbers, that will generate the n^{th} triangular number.



Program Input

The first line of triangularNumbers.in.txt will contain a positive integer T denoting the number of test cases that follow. Each test case contains a single integer, n, denoting the n^{th} triangular number. The number n will be less than or equal to 200.

Example Input

4
1
2
5
10

Program Output

For each test case your program should output one line containing which triangular number is being output and its corresponding value. The format of each line of output is <POSITION> = <VALUE>. There is a time-constraint on this problem. I.e., to validate the efficiency of your solution, you should generate all of your output in under a predetermined time.

Example Output

1 = 1
2 = 3
5 = 15
10 = 55

Problem Set 1

Problem 2 - It’s-a Me, Mario!

Introduction

Toward the end of World 1-1 in Nintendo’s Super Mario Brothers, Mario must ascend a "half-pyramid" of blocks before leaping (if he wants to maximize his score) toward a flag pole. Below is a screenshot.

Write a method named mario that recreates this half-pyramid using hashes (#) for blocks. Take care to align the bottom-left corner of your half-pyramid with the left-hand edge of your terminal window, as in the example output below.

Program Input

The first line of mario.in.txt will contain a positive integer T denoting the number of test cases that follow. Each test case contains a single, non-negative integer less than or equal to 23.

Example Input

```
2
2
4
```

Program Output

Each block is represented by a hash. The lower-left corner of the pyramid should align with the left side of the terminal window.

Example Output

```
 ##
###
   ##
  ###
 ####
#####
```

Problem Set 1

Problem 3 - URLify

Introduction

Web URLs cannot contain spaces, but must exclusively contain ASCII characters. Often, though, when web forms are submitted, spaces are thrown into the mix. To handle this, each space in the would-be URL is typically encoded with '%20'. Write a method, urlify, which takes a String and replaces all instances of a space with a '%20'.

Program Input

The first line of urlify.in.txt will contain a positive integer T denoting the number of test cases that follow. Each test case contains a single line of text, representing the String that should be converted into a space-free URL.

Example Input
2
Mr John Smith
Who let the dogs out

Program Output

For each test case your program should output one line containing the space-free URL. Note that if there are any trailing spaces in the input String, that a URL should remove the padded spaces. I.e., any leading or trailing whitespace should be eliminated. See Java's trim method for some help.

Example Output
Mr%20John%20Smith
Who%20let%20the%20dogs%20out

Problem Set 1

Problem 4 - What's the 411?

Introduction

Write a method named fourOneOne that provides some information about a group of integers. For each group of integers you are to report the count of the numbers, the frequency of even numbers, and the percent of even numbers in the group of numbers.

Program Input

The first line of fourOneOne.in.txt will contain a positive integer T denoting the number of test cases that follow. Each test case contains a single line of comma-separated positive integers, representing the group of numbers for one test case.

Example Input
2
5,7,2,8,9,10,12,98,7,14,20,22
3,2,6,9

Program Output

For each test case your program should output one line containing the summary information. The percentage of even numbers should be rounded to two decimal places. Read about Java's System.out.format for assistance. The output should match the format of:

"COUNT numbers, NUM_EVENS evens (PERCENT%)"

COUNT represents the number of integers in the group and NUM_EVENS is the frequency of even numbers, and PERCENT represents the percentage of even numbers rounded to two decimal places.

Example Output
12 numbers, 8 evens (66.67%)
4 numbers, 2 evens (50.00%)

Problem Set 1

Problem 5 - Gotta Catch ‘em All

Introduction

I have a bucket of Poké balls that needs to be sorted into *complete sets*. These sets aren’t all that impressive, though. The poké balls found in the bucket only contain the three starter Pokémon: Bulbasaur, Charmander, and Squirtle. What I need help with is determining when I have a *complete set* of poké balls. A complete set is when there are an equal number of Bulbasaur, Charmander, and Squirtle poké balls taken out of the bucket.

The method, pokemon, will read in a list of letters representing each Pokémon (B for Bulbasaur, C for Charmander, and S for Squirtle) that comes out of the bucket. When a complete set (greater than 0, 0, 0) has been achieved, the method will print “Complete set obtained at NUMBER”, where NUMBER represents the *complete set number*. The *complete set number* is the number of Poké balls of each Pokémon when a complete set has been formed.

Program Input

The first line of pokemon.in.txt will contain a positive integer T denoting the number of test cases that follow. After that, for each test case there will be one letter (B, C, S) per line representing the Pokemon found in the pokemon that was removed from the bucket.

Example Input

```
2
C
B
S
B
S
S
C
B
C
```

Program Output

You can assume that for the given input file, complete set will be achieved exactly as many times as the number of test cases listed, and that there will be no more lines in the input after the last balance.

Example Output

```
Complete set obtained at 1.
Complete set obtained at 2.
```