**Cypress Woods**

**Computer Science Competition 2019**

1. Problems may be solved in any order you choose. They do not have to be done in order from 1 to 18.
2. All problems are worth 60 points. Incorrect submissions will subtract 5 points from the points rewarded if the problem is submitted correctly. No points are subtracted if the problem is never submitted correctly.
3. There is no extraneous input. All input is exactly as specified in the problem.
4. Unless specified by the problem, integer inputs will not have leading zeroes. Your program should read to the end of file unless otherwise specified.
5. Your program should not print extraneous output. Follow the form exactly as given in the problem.
6. All programs must run under 2 minutes.

|  |  |  |
| --- | --- | --- |
| Problem # | Problem Name | Solved? |
| 1 | EOS or EOF? |  |
| 2 | Tree |  |
| 3 | Never Have I Ever |  |
| 4 | Yeet |  |
| 5 | Quiz Bot |  |
| 6 | Data Reader |  |
| 7 | The Power of 212 |  |
| 8 | Ocho |  |
| 9 | ChristMaths |  |
| 10 | Simulator Accuracy |  |
| 11 | Portmanteau |  |
| 12 | Sandwiches |  |
| 13 | Messy Written |  |
| 14 | Like, Um, Uh |  |
| 15 | Recruits |  |
| 16 | There’s Always Tomorrow |  |
| 17 | Babel Fish |  |
| 18 | Bodies |  |

**1. EOS or EOF?**

# Program Name: EOF.java Input File: eof.dat

Write a program to look for the EOF. The EOF will be somewhere in the file; output all input prior to reaching it.

**Input**

The input file contains an unknown number of lines of unknown data.

**Output**

Print out the input data until you reach the end of the file.

**Example Input File**

Ajskdfvnlasdfknsdfanb

Enr

BaskldnfEOFlna

Dfvnsdfjdnvfndknfdvjklnfjkalsbnjnkldfjklndsbjklfjdsvkajfdsfv333

3reej3iu25o3u5235

r

**Example Output to Screen**

Ajskdfvnlasdfknsdfanb

Enr

Baskldnf

**2. Tree**

# Program Name: Tree.java Input File: None

Print out this picture of a Christmas tree.

**Input**

None.

**Output**

The tree.

**Example Input File**

None

**Example Output to Screen**

\*

/|\

// \\

// \\

// o \\

// o \\

// o \\

//\_\_\_\_\_\_\_\_\_\_\_\\

|||

|||

|||

|||

---

**3. Never Have I Ever**

# Program Name: Never.java Input File: never.dat

In the game of Never Have I Ever, one person says “Never Have I Ever…” and then something they have never done. If more than half of the group has done the action at some point in their lives, the asker wins. If less than half of the group has done the action at some point in their lives, the asker loses. Given some things you have never done, print them out.

**Input**

There will be an unknown number of lines, each with a phrase.

**Output**

Print each phrase as is, but with “Never Have I Ever” added at the beginning, and a period added at the end.

**Example Input File**

failed a class

gotten a 240

had a girlfriend

**Example Output to Screen**

Never Have I Ever failed a class.

Never Have I Ever gotten a 240.

Never Have I Ever had a girlfriend.

**4. Yeet**

# Program Name: Xx.java Input File: xx.dat

Zeki is really convinced that Yeet Theorem really works, and decides to dedicate a whole line of research behind this theorem. Yeet theorem states that to raise a number n to the power p, you can “yeet” the power in front of the number and concatenate them to get your resultant number. However, Zeki is really stupid and needs your help to both determine the result using Yeet Theorem and find out if it equals the real value of n raised to p.

**Input**

Given t describing the number of test cases. For each test case, you are given n and p (0 ≤ n, p ≤ 10^100).

**Output**

Return the value after yeeting the power and if the value is valid. If the value equals n^p, print “Yeet”. Otherwise, print “Get Yoted”.

**Example Input File**

3

5 2

3 3

100 0

**Example Output to Screen**

25 Yeet

33 Get Yoted

100 Get Yoted

**5. QuizBot**

# Program Name: quizbot.java Input File: quizbot.dat

Jonathan decided that it was a good idea to start to develop a quiz bot that could quiz his friends. However, he couldn’t figure out how to compare the answers that the players gave to the answers of the question. So he’s forced to turn to you. Could you help Jonathan figure out how to analyze the answers and print out the winner in the quiz?

**Input**

The first line will have an integer n, the number of test cases to follow. Each test case consists of 3 lines: The first line is the two names of the players separated by a space. The second line contains the proper answers to the questions separated by a space and the last line is the answers that each player has put in player1 player2 order. There will always be two player answers for each question answer. The answers will always be in the same case.

**Output**

The output will be a single line that says who won the round in the format “(Winner) has won this round!”. If the round ends as a tie, output “(player1) and (player2) are tied this round!

**Example Input File**

2

JohnChris ChrisJohn

Five Gray Green Orange

Five Blue Blue Green Orange Green Orange Yellow

Marie Clarie

Five Four

Five Four Five Four

**Example Output to Screen**

JohnChris has won this round!

Marie and Claire are tied this round!

**6. Data Reader**

# Program Name: Datareader.java Input File: datareader.dat

Given a string of letters, and a string that represents the correct string, find the line that represents the largest amount of correct letters in position. If no letter is correctly placed, a message ‘Failed’ will be printed. If more than one line has the same number of letters in place, the first will be printed. For the correct solution, print out a confidence interval that represents the percentage of letters that were in the correct position following this equation:

**Input**There will be a number that represents the number of dataset that will follow *n,* each section will have a number, *l,* that represents the number of lines that follow. The first line will be the string to compare to, the rest are the incorrect attempts. All lines will have the same number of letters.

**Output**

Output a string that represents the line with the most letters in the correct position, and the next line represents the confidence interval.

**Example Input File**

2

4

wildcat

wdatlci

awtcdil

wiadclt

4

sid

isd

ids

dsi

**Example Output to Screen**

wiadclt

Confidence Interval: 71%

isd

Confidence Interval: 33%

**7. The Power of 212**

# Program Name: Power.java Input File: power.dat

Sammy has run into a problem! He woke up on a Monday and lost all of his 212 Spirit! Luckily for him, Mr. Armstrong, Mr. Kinninger, and Mr. Heath are all dedicated to getting their most spirited Wildcat back in the game. They have devised a series of numbers designed to brainwash Sammy into having Wildcat Pride and, most importantly, 212 spirit again. All they need is a program to take in the numbers and letters and return the brainwashing phrase.

**Input**

The first line will contain a single integer N denoting the number of test cases to follow.

The following N lines will contain an integer M and a single word (without spaces) on the same line.

**Output**

If the number M is a multiple of 212 AND the word can be spelled using only characters found in the word “wildcat” (using a character more than once is allowed), then print “212 spirit attained”.

If only one of the conditions is met, print “Be the hyena”.

If neither of the conditions are met, print “This is some Eight Rivers garbage”.

However, if the number is 212 AND the word is “wildcat”, automatically print “Heaven on Earth” and nothing else.

**Example Input File**

4

424 call

212 wildcat

7 sevenlakes

1060 time

**Example Output to Screen**

212 spirit attained

Heaven on Earth

This is some Eight Rivers garbage

Be the hyena

**8. Ocho**

# Program Name: Ocho.java Input File: Ocho.dat

For some weird reason, you work at a company that converts numbers written in Spanish and compares them to other numbers. While this is simple enough, your boss graduated from 8 Rivers Middle School and enforces a policy requiring 8 to be greater than everything (except for itself, in which it is equal to 8). Write a program to do your job for you.

**Input**

The first number t indicates the number of test cases that follow. For each test case, there are two all uppercase words in Spanish. Spanish numbers are as follows:

* 0 = CERO
* 1 = UNO
* 2 = DOS
* 3 = TRES
* 4 = CUATRO
* 5 = CINCO
* 6 = SEIS
* 7 = SIETE
* 8 = OCHO
* 9 = NUEVE
* 10 = DIEZ

**Output**

For each test case, output the first number followed by a greater than, equal to, or less than sign followed by the second number

**Example Input File**

4

UNO DOS

TRES UNO

DIEZ CERO

DIEZ OCHO

**Example Output to Screen**

1 < 2

3 > 1

10 > 0

10 < 8

**9. ChristMaths**

# Program Name: christmaths.java Input File: christmaths.dat

Santa is finally finished giving all his presents to the good children, and he’s ready to call it a night. Santa is feeling extra generous this Christmas, however, and decides to give all the children on the naughty list a second chance. He wants to see if the children have learned anything in their math classes, so he decides to give them a basic arithmetic problem. Unfortunately, it turns out that Santa dropped out of school a little too early in order to pursue the art of gift giving, so he has no concept of order of operations. Instead, Santa does each operation in the order that it is given to him. Given a mathematical expression and a child’s answer to the expression, determine whether Santa will give them a present or not. Santa will give the child a present if and only if the child’s answer to the expression is equal to what Santa would have gotten. (Note: Santa can use all 4 basic operators: +, -, \*, /. All division will be integer division.)

**Input**

The first line of input contains 1 integer n, the number of test cases to follow. The next n lines of input contain a mathematical expression and the child’s answer for the expression.

**Output**

Determine whether Santa will give the child a present. If Santa is willing to give the child a present, print out “Present for You”, and if he will not give the child a present, print out “Coal for You”.

**Example Input File**

3

3+5\*4–2 30

4-6\*2+12 4

6/2+5-1 7

**Example Output to Screen**

Present for You

Coal for You

Present for You

**10. Simulator Accuracy**

# Program Name: Simulator.java Input File: simulator.dat

Felix is trying to build his own simulator to predict how a stock will do over a certain time period. He creates a program to that predicts what a stock’s price will be during the course of year. He wants to test the accuracy of his model, and he needs your help doing this. Given three trials simulated trials and a real stock price after a period of time, output the accuracy of his model in the form of a percentage.

**Input**

The first line of input will contain an integer n which is the number of test cases. The first three numbers in any following line are the three simulated trials. The next value is the price of the stock after a time period.

**Output**

The output should be the percent error between the three data trials and the actual stock price, formatted as a percentage with two decimal places.

**Example Input File**

4

110.89 111.61 111.98 129.30

209.69 210.68 212.69 257.24

188.01 187.07 188.54 234.01

1106.79 1112.62 1101.08 1291.01

**Example Output to Screen**

13.77%

17.97%

19.72%

14.27%

**11. Portmanteau**

# Program Name: Portmanteau.java Input File: portmanteau.dat

A portmanteau is a combination of two words into a single word. Examples include motel (motor + hotel), brunch (breakfast + lunch), and hangry (hungry + angry). Given a portmanteau and two words, determine if it is possible to form the portmanteau with said words. A portmanteau is said to be formable if a continuous stream of characters from the beginning of one word combined with a continuous stream of characters from the end of the other word makes the portmanteau. The streams must both be of nonzero length.

**Input**

The first line of input will contain a single integer n that indicates how many test cases to follow. Each test case will contain the portmanteau, then two words.

**Output**

If it is possible to create the portmanteau from the words, output “YES”. Otherwise, output “NO”.

**Example Input File**

4

motel motor hotel

brunch lunch breakfast

jangry hungry angry

whism antidisestablishmentarianism what

**Example Output to Screen**

YES

YES  
NO

YES

**12. Sandwiches**

# Program Name: Sandwiches.java Input File: sandwiches.dat

To make a sandwich, you need bread, meat, and cheese. You went to the grocery store earlier this week and made a log of what items you bought as you were putting them in your refrigerator. Being an organized person, you separated each type of food into different drawers. However, this also means the first item you wrote down is at the very back of your refrigerator, and you don’t want to just go off the list and dig out everything from the back. So, you’ve decided to write a program to create sandwich combinations for you to eat. Items will not be put back into the refrigerator once you have eaten them for obvious reasons. If any of the three items are missing, then you cannot make a sandwich.

**Input**

The first line of input will contain a single integer n that indicates the number of things you bought. The next n lines will contain a log of all the items you bought, with the name, a dash, and then the type of food it is – bread, meat, or cheese. These lines are in the order you put the food into the refrigerator.

**Output**

For each possible sandwich combination, output the sandwiches in the order you will be eating them, and the sandwiches’ ingredients in the format bread, meat, cheese.

**Example Input File**

10

Whole wheat-bread

White-bread

Turkey-meat

American-cheese

Swiss-cheese

Ham-meat

Cheddar-cheese

Chicken-meat

Hamburger buns-bread

Patty-meat

**Example Output to Screen**

Hamburger buns, Patty, Cheddar

White, Chicken, Swiss

Whole wheat, Ham, American

**13. Messy Written**

# Program Name: written.java Input File: written.dat

Written tests are hard to grade, and sometimes, something goes wrong. As we look around, Ral has received a 236 on the written test! The way written tests are scored is 6 points for every question correct, no points awarded for skipped questions, and -2 points for incorrect answers. There are always 40 questions on a written test. All of us trying to figure out if this is actually possible ask a programmer for some help. Write a program to determine whether a given written score is possible.

**Input**

The first line of input is n, the number of data sets to follow. The next n lines will be an integer within the range -80 to 240.

**Output**

For each test case, print “Possible” if the written score is possible or “Impossible” if the written score is impossible.

**Example Input File**

5

118

101

236

-12

86

**Example Output to Screen**

Possible

Impossible

Impossible

Possible

Possible

**14. Like, um, uh**

# Program Name: Like.java Input File: like.dat

Alex stutters when he talks too much, and fills up the empty space with words like like, um, and uh. Mr. A wants to help him overcome his speech problem. Write a program to remove all filler words from a sentence.

**Input**

The first line of input will contain a single integer n that indicates how many test cases to follow. Each test case will contain a sentence on one line.

**Output**

For each sentence, output the sentence without any filler words and commas that may be around the filler words. Commas should only be removed if they surround only the filler word and nothing else. If a phrase is in the format , like word, only the first comma and like will be removed. You may assume all commas in the input will have a character before the comma and a space after the comma.

**Example Input File**

7

What if, like, we, uh, wrote more contest problems about like stupid stuff

Your thing is, like, not working

He, um, didn't like do that

Can you like please help me solve this foobar problem

What about like the equation for the uh, um, left branches

I was made in his likeness

First in, first out

**Example Output to Screen**

What if we wrote more contest problems about stupid stuff

Your thing is not working

He didn’t do that

Can you please help me solve this foobar problem

What about the equation for the left branches

I was made in his likeness

First in, first out

**15. Recruits**

# Program Name: Recruits.java Input File: recruits.dat

A team is a group of at most 3 people. You are in charge of handling the people who are interested in joining teams. Given a list of raw data consisting of recruits and a team they want to be on, sort them into respective teams and print them out in a more friendly format. Teams are created on a first come, first serve basis. If somebody wants to join a team that already has 3 people, they will not be able to join.

**Input**

The first line of input will contain a single integer n that indicates how many test cases to follow. The first line of each test case will be a single integer r that indicates the number of recruits that signed up. Each of the following lines consists of the recruit’s name, then a dash -, followed by their preferred team’s name. If a recruit does not have a preferred team, they will be placed with other team-less recruits and recruits who could not get into their preferred team.

**Output**

Output each team’s name followed by the members of the team, both in alphabetical order. Any recruits without a team will be placed a team named Untitled Team X, where X is a number that starts at 1 and increments for each untitled team that is created. Separate each team with a newline, and separate each test case with ----- (5 dashes). There will also be a newline both in front of and after each set of dashes.

**Example Input File**

2

8

Alex-Team A

Mihir-Team A

Zeki-Team B

Ashay-Team A

Sidh-Team A

Ronak-Team B

Tristan

Ral

6

ASDF-GHJK

QWERTY

2539

J-J

foo

bar

**Example Output to Screen**

Team A

Alex

Ashay

Mihir

Team B

Ronak

Zeki

Untitled Team 1

Ral

Sidh

Tristan

-----

GHJK

ASDF

J

J

Untitled Team 1

2539

foo

QWERTY

Untitled Team 2

bar

-----

**16. There’s Always Tomorrow**

# Program Name: Tomorrow.java Input File: tomorrow.dat

After having his business idea brutally rejected by Mark Cuban, Lil’ Willy Brazofuerte, world-renowned rapper, is feeling a little upset. As his best friend, you want to help him cope with his struggles, so you tell him that “there’s always tomorrow”. “But what is tomorrow?” Lil’ Willy B. asks you. Now, your task is to tell him.

**Input**

The input will begin with a single integer, t, denoting the number of test cases to follow. Each test case will contain a date formatted as MM DD YYYY.

**Output**

Output the date of exactly one day after the date given, taking into account leap years. Please output each date in the MM DD YYYY format and do not print blank lines between each test case.

**Example Input File**

3

12 31 6969

12 14 2019

02 28 2004

**Example Output to Screen**

01 01 6970

12 15 2019

02 29 2004

**17. Babelfish**

# Program Name: Babelfish.java Input File: babelfish.dat

The Babel fish is small, yellow, leech-like, and probably the oddest thing in the universe. It feeds on brainwave energy; if you stick a Babel fish in your ear, you can instantly understand anything said to you in any form of language. The speech patterns you actually hear decode the brain wave matrix, which has been fed into your mind by your Babel fish. However, you dropped your Babel fish in the toilet and now it’s defective. Luckily, there’s a way to fix it.

Your Babel fish can only decode Vogon. Provided below is the algorithm the Babel fish uses to translate Vogon to English. Your task is to follow this algorithm to fix your Babel fish.

1. At the beginning of each test case, there will be an integer s, which will be either positive or negative. A positive value represents a shift s places to the right, while a negative value represents a shift s places to the left. Vogon only includes the characters with the ASCII values between 32 and 122. If you shift past these boundaries, you should wrap around. For example, if a character with the ASCII value 32 shifts to the left 1, it should now have the ASCII value 122.
2. With the new string, reverse all values; for example, the string ABCDE would become EDCBA.
3. Now, create a new string using the value of d. First, remove the dth character of the string and add it to a buffer. Next, keep removing at each multiple of d until the string is empty, wrapping around when d is greater than the length of the string. Look below for an example that is traced out with a d value of 3.

String Buffer

EDCBA null

EDBA C

DBA CE

DB CEA

B CEAD

CEADB

**Input**

The first line of the input file will have an integer n denoting the amount of test cases to follow. The first line of each test case will have two integers, s and d. s can be either positive or negative to indicate the direction and magnitude of the shift, and d will always be >0. The second line of each test case will contain an encoded sentence that needs to be decoded.

**Output**

Output the decoded Babel fish result of each encoded message.

**Example Input File**

3

-5 2

omoq%\*qd$F]0F$q0s]Wq=

3 11

7((M:(M0exi8e0iK0(<jr(WHG

-1 4

/;i;6w@;ZTJmwxOwTpZwCZOTw@;ZTJm\_

**Example Output to Screen**

212 IS GREATER THAN 7

Beeblebrox for President!

Vogon poetry is the best poetry.

**18. Bodies**

# Program Name: bodies.java Input File: bodies.dat

Mr. A just killed someone. He has tasked his butler, Tristan Weaslepopsicle, to dispose of the body at night. Mr. A has already identified possible sites to dispose of the body, but there are police officers who roam around. Create a program to see if Tristan can dispose of the body without being caught by the police.

**Input**

The first line represents the number of data sets to follow. In each data set the first line contains the rows and columns of the map, respectively. the next “r” lines will contain the map. The “T” represents where Tristan starts. There can be 1 to 3 “S”’s, which represents possible sites to dispose of the body. There can be 0 to 3 “P”’s, which represent police officers. Tristan can travel in any square except one space around a police officer, including diagonals, or a wall, denoted by a “#”. You can assume Tristan’s starting position won’t be right next to a police officer, but a possible site could be right next to a police officer, making it automatically unaccessible.

**Output**

Always output “Bravo Six Going Dark...” to signify that Tristan has started his night time mission. If it is possible for Tristan to dispose of the body without getting caught, output “at the end of the tunnel is a light” on the same line. Otherwise output “Mission Failed. We’ll get em next time.”

**Example Input File**

2

6 7

.S.####

.#P..##

T.##P##

...#..S

#....##

#####S.

5 5

.....

....S

..P.#

..#.#

T...#

**Example Output to Screen**

Bravo Six Going Dark... Mission Failed. We'll get em next time.

Bravo Six Going Dark... at the end of the tunnel is a light.