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 | Minecraft II |
| Problem 2 | Sketch |
| Problem 3 | Minecraft I |
| Problem 4 | License and Registration |
| Problem 5 | Roundabout |
| Problem 6 | Redlight |
| Problem 7 | Scrambled Eggs |
| Problem 8 | Cyclist |
| Problem 9 | Farmboy |
| Problem 10 | Street Race |
| Problem 11 | City Planning |
| Problem 12 | DMV |

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

**1. Minecraft II**

# Program Name: Minecraft2.java Input File: minecraft2.dat

You have made some serious advancements in Minecraft since leaving the nether, but lately you’ve been having issues with your crafting. You need to write a program to determine what ingredients you need to make certain items. You will be given a list of crafting recipes, a list of your current supplies, and which items you need to make, and you need to determine which raw materials you still need to collect in order to make the given item(s). You will take the recipe for the item you want to craft, determine all the ingredients, and subtract the ones you already have. If a recipe for an item is not given, assume that you must find that item, you want to break down each item to its lowest components, in other words, determine the sum of all items you need in order to craft all items you need to craft the given item.

**Input**

The first line will contain 2 integers, n and m, denoting the number of crafting recipes and the number of data sets, respectively. The next n lines will each contain a crafting recipe in the format: x item\_name: a item1\_name, b item2\_name, …, where x is the amount of the item begin crafted, item\_name is the name of the item being crafted, item1\_name is the name of the first ingredient required, a is the amount of the first ingredient, item2\_name is the name of the second ingredient required, b is the amount of the second ingredient, and so on. After these n lines, there will be m data sets. Each data set will begin with one line containing a string, the name of the item to be crafted, and two integers, the amount of the item you need to craft, and i, the number of items currently in your inventory, respectively. The next i lines will contain a string integer pair, the name of the item in your inventory, and the amount of that item you have, respectively. None of the names will contain any spaces.

**Output**

For each item to be crafted, output a list of all the ingredients you need and the amount of them you need (minus those already in your inventory), sorted by name alphabetically, separated by commas and spaces (in the format shown in the example output file). If all the items needed are in your inventory, output the string "All Supplies in Inventory.".

**Minecraft II continued on next page.**

**Minecraft II continued from prior page.**

**Example Input File**

10 3

1 Dispenser: 1 Bow, 1 Redstone\_Dust, 7 Cobblestone

1 Bow: 3 Stick, 3 String

2 Stick: 3 Wood\_Plank

4 Wood\_Plank: 1 Wood

1 Piston: 1 Redstone\_Dust, 3 Wood\_Plank, 1 Iron\_Ingot, 4 Cobblestone

1 Sticky\_Piston: 1 Piston, 1 Slimeball

1 Minecart: 5 Iron\_Ingot

1 Hopper: 1 Chest, 5 Iron\_Ingot

1 Chest: 8 Wood\_Plank

1 Hopper\_Minecart: 1 Hopper, 1 Minecart

Dispenser 3 4

4 Wood

4 String

12 Cobblestone

6 Redstone\_Dust

Sticky\_Piston 4 2

2 Slimeball

6 Iron\_Ingot

Hopper\_Minecart 1 1

4 Iron\_Ingot

**Example Output to Screen**

9 Cobblestone, 5 String

16 Cobblestone, 4 Redstone\_Dust, 2 Slimeball, 3 Wood

6 Iron\_Ingot, 2 Wood

**2. Sketch-an-Etch**

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

You and Geraldo want to have some fun so you decide to do some snooping around. You have had enough of playing interdimensional chess with Captain Shady so you start looking around his house. You find Shady’s Sketch-an-Etch buried in his underwear drawer, and decide to figure out how it works. This looks like fun.

**Input**

There will be an unknown number of data sets in the data file. Each data set will consist of 3 integers, shape, size, and c, followed by a Boolean, fill. The integers refer to different aspects of the shape, the value of shape corresponds to the shape you are to draw(see the table below), size is the width and length of the shape, and c refers to which character you are to use to draw the shape, while fill denotes whether or not to fill the shape.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Number | Shape | Number | Shape | Number | Char | Number | Char |
| 0 | Square |  |  | 0 | ! | 3 | $ |
| 1 | Down left triangle | 3 | Up left triangle | 1 | @ | 4 | % |
| 2 | Down right triangle | 4 | Up right triangle | 2 | # | 5 | & |

**Output**

You are to output the shape described by the input.

**Example Input File**1 5 3 true4 3 2 false0 4 0 false

**Example Output to Screen**

$$$$$

$$$$

$$$

$$

$

#

##

###

!!!!

! !

! !

!!!!

**3. Minecraft I**

# Program Name: Minecraft.java Input File: minecraft.dat

You and Geraldo have been playing a lot of Minecraft lately, but have gotten lost in the nether, and you’re out of food. You need to write a program to determine if you can get back to your nether portal. You need to sprint the whole distance so you need to have at least 2/10 hunger bars. The only food source available is Hoglins, which drop pork chops, each of which will replenish 2 hunger bars. You will be given a map of the terrain, made up of the following characters:

'.' – represents nether rack, which you can cross at a speed of 1 space per second.

'W' – represents lava, which cannot be traversed.  
's' – represents soul sand, which you can cross at a speed of 1 space every 2 seconds, unless you have your special boots on, then your speed is one space per second.  
'f' – represents a nether fortress, which you can cross at a speed of 1 space per second, but must fight monsters, so every 5 seconds you will drop a hunger bar from fighting (this counter resets whenever you leave the fortress.)

'G' – represents a ghast, which you must stay at least 3 spaces away from in every direction.

'H' – represents a Hoglin, which takes 1 second to kill, then turns into a '.', and yields 3 pork chops.

'b' – represents a blaze, which you must stay at least 2 spaces away from in every direction.

'#' – represents a wall or a tree, or an otherwise impassable block.

'p' – represents a piglin, which can be passed near, but not on top of if you have your special boots on, otherwise, you cannot pass within 2 blocks of them.

'S' – represents Steve’s position (your starting point).

'O' – represents the location of your nether portal (your ending point).

You can only move in the 4 cardinal directions (up, down, left, right). Every 5 seconds of travelling costs 1 bar of hunger, however, you take 1 second to look at the map at the beginning of your journey (assume you have already travelled for 1 second to get to the starting point).

**Input**

The file will begin with an integer n denoting the number of data sets to follow. Each data set will begin with 3 integers, r, c, and h, denoting the number of rows and columns in the map, and the number of hunger bars you currently have respectively, and a Boolean, b, denoting whether or not you are wearing your special boots. Each of the following r lines will contain c characters denoting the map of the layout of the terrain.

**Output**

If it is possible to get from the starting point to the exit, print the string "Nether gonna give you up.", otherwise output the string "Anether one.".

**Minecraft I continued on next page.**

**Minecraft I continued from prior page.**

**Example Input File**

2

5 6 7 false

..S.ss

ssssss

ssss.#

WWW..#

O....H

8 8 10 true

S..sss.G

..WWssss

ssssWW##

ffffWWWW

ffffbWWW

ffffWWWW

ffff...p

H....p.O

**Example Output to Screen**

Nether gonna give you up.

Anether one.

**4. License and Registration**

# Program Name: License.java Input File: license.dat

After the party is over, Geraldo goes to sleep to get ready for work the next day. On his way to the office, he gets stopped by a police officer, and is asked to hand over his license. He doesn’t have it, but the Compsci City Police Department has a machine in each vehicle that can automatically retrieve a digital copy of the license, as long as there is one in the system. Design a program that takes in the name of a person and prints a license for them

**Input**

The first line will contain a single integer n that represents the number of single lines that follow. Each single line will contain the name of a citizen within Compsci City.

**Output**

Output the citizen's name surrounded by one character of blank space, and then by one character of "\*". Each license will be separated by one line.

**Example Input File**

3

Alfred Jones

Nomis Dorris

Maggie Wallace III

**Example Output to Screen**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Alfred Jones \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Nomis Dorris \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Maggie Wallace III \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**5. Roundabout**

# Program Name: Round.java Input File: round.dat

After a long, tiring day at work Geraldo decides to do the most exciting thing he can think of: Joyride in a roundabout until he runs out of gas. Given that he only has a limited amount of gas, find some way to determine how many times he can go around a given roundabout, ignoring any partial revolutions that he could make. His car runs 20 miles to the gallon.

**Input**

The first line will contain a single integer n that represents the number of single lines that follow. Each single line will contain two integers that will represent the gallons of gas in his car, and the total distance around a roundabout in feet.

**Output**

Output the integer number of times that Geraldo can make it all the way around the roundabout.

**Example Input File**

3

1 580

2 650

3 7000

**Example Output to Screen**

182

324

45

**6. Redlight**

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

Traffic is the worst part about Compsci City. The redlights cause you to be late to work no matter how early you get there! Given a certain travel time, see if the number of redlights that you run into will cause you to be late. They average around 2 minutes per stop.

**Input**

The first line will contain a single integer n that represents the number of single lines that follow. Each single line will contain the time to get to a location, the maximum time allowed to get there, and the number of redlights between you and that location.

**Output**

Output "yes" if you will make it on time, and "no" if you will not make it on time.

**Example Input File**

3

20 30 6

20 30 5

10 15 2

**Example Output to Screen**

no

yes

yes

**7. Scrambled Eggs**

# Program Name: Eggs.java Input File: eggs.dat

Geraldo needs to determine how many eggs he has left in the carton to make breakfast. You will be given the size of the carton, and you need to determine how many eggs there are in a continuous group. Geraldo is very geometrically oriented, and he wants to find the maximum sized contiguous block of eggs in the carton. Basically, find the largest contiguously (connected up, down, left, and right) blob of eggs so Geraldo can use it for breakfast.

**Input**

The first line will contain a single integer n that represents the number of single lines that follow. Each single line will contain two integers, r and c, denoting the number of rows and columns in the carton of eggs. Each of the next r lines will contain c characters denoting the egg carton. An '0' denotes an egg, and a '\_' denotes an empty space with no egg.

**Output**

Output a how many eggs are in the largest group in the carton so that Geraldo knows how much breakfast he can make (he wants to invite friends).

**Example Input File**

3

3 5

00\_\_0

\_00\_\_

0\_00\_

2 6

000\_00

0\_00\_0

1 12

00\_000\_00\_00

**Example Output to Screen**

6

6

3

**8. Cyclist**

# Program Name: Cyclist.java Input File: cyclist.dat

After eating his egg of the day, Geraldo looks out his window to view the many cyclists that pass by. He plays a game where he counts the number that pass by in a five minute interval, and starts over when 9 pass by. Create a list of these chains of cyclists and sort them by number, but make sure to use words and not numbers in the final lists!

**Input**

The first line will contain a single integer n that represents the number of single lines that follow. Each single line will contain the number of cyclists that pass by Geraldo’s window over the course of five minutes. There will always be a 9 at the very end of all of the lines.

**Output**

Output on separate lines the sorted lists of cyclist numbers as per the rules of the game. Use words instead of numbers, so a final list would be [one,two,nine]

**Example Input File**

5

2

1

9

2

9

**Example Output to Screen**

[one,two,nine]

[two,nine]

**9. Farmboy**

# Program Name: Farm.java Input File: farm.dat

Take a shipment of a certain food, and pack it into the smallest square box you are able to in order to transport it to the city. This is the job Geraldo has managed to find himself performing as he has taken a break from city life. However, if the food matches any food on the “Do NOT Import List”, then it must not be packed, instead a certificate of disapproval must be sent to the store that ordered it.

**Input**

The first line will contain a single integer n that represents the number of single lines that follow. Each single line will contain a string so that its length is less than 37.

**Output**

Output a matrix, each separated by a line of space, where the string’s characters are layered into the smallest possible square matrix, or output "SHIPMENT DISAPPROVED". Empty space can be filled with "\*"

**Example Input File**

3 4

durian peanut pear avocado

apple

banana

durian

**Example Output to Screen**

\*\*\*

le\*

app

\*\*\*

ana

ban

SHIPMENT DISAPPROVED

**10. Street Race**

# Program Name: Race.java Input File: race.dat

Deciding that country life just isn’t for him, Geraldo goes and does the most city thing he can think of. He becomes a street racer. However, after an intense race he finds himself caught in an unusually maze-like part of the city, and needs help to find his way out. Sometimes you can get completely stuck in Comspsci City if someone moves their car in front of an alley, or the city police put a barricade up. Let Geraldo know if he can get out on his own, or needs to call for help.

**Input**

The first line will contain a single integer n that represents the number of single lines that follow. Each single line will contain two integers representing the width and height of a matrix that follows. Geraldo’s starting point will be represented by an "@" and the free space he can travel on by "#". All other characters will represent blocked space, and all free spaces are connected to each other. Geraldo will never start on the edge of a matrix.

**Output**

Output "OUTSTANDING. I can escape from my predicament" if it is possible to leave the area Geraldo is in, or "OH NO. I am very stuck and must call for help" if it is not possible to leave the area.

**Example Input File**

2

4 4

\*\*\*\*  
\*###  
\*#@\*  
\*\*\*\*

4 4

\*\*\*\*  
\*@#\*  
\*##\*  
\*\*\*\*

**Example Output to Screen**

OUTSTANDING. I can escape from my predicament

OH NO. I am very stuck and must call for help

**11. City Planning**

# Program Name: City.java Input File: city.dat

Compsci city is planning on holding an anniversary celebration for their city founding. It needs to be within approximately 30 days of April 14th, the date the city was founded, in the next year, 2023.

**Input**

The first line will contain a single integer n that represents the number of single lines that follow. Each single line will contain a day and month in 2023 that is a possible date for the celebration.

**Output**

Output "outstanding work gentlemen" if the date is within the appropriate date range, and "this is the worst of all possible worlds" if the date is not.

**Example Input File**

3

14 4

14 3

30 1

**Example Output to Screen**

outstanding work gentlemen

this is the worst of all possible worlds

this is the worst of all possible worlds

**12. DMV**

# Program Name: DMV.java Input File: dmv.dat

The DMW has hired Geraldo for a job! He is very excited, and wants your help in making sure he doesn’t mess it up. Some people are really bad at spelling, and it is your job to determine how far off they are from the name they are actually trying to spell. You will be given the correct spelling and the written spelling for people's first and last names, and you must determine how many of the following actions must be done to turn the written spelling into the correct spelling.  
Possible Actions:  
 Insert: insert any character before or after a given index in the written spelling.  
 Remove: remove a character from the written spelling.  
 Replace: replace a character at any index of the written spelling with some other character.

**Input**

There will be an unspecified number of test cases. Each test case will consist of two lines, with the first containing the written spelling of the name, and the next line containing the correct spelling.

**Output**

Output the minimum number of edits needed to get the written spelling to the correct spelling.

**Example Input File**

sum smoth

sam smith

binjomenn anndirsunn

benjamin anderson

jameson braun

james brown

**Example Output to Screen**

2

8

4