New Mexico State University Programming Competition Spring 2023

Problem Set

Problem 1: Subnets

Point Value: 2 Points

Bounty: $30

Description:
The IPv4 addressing scheme is used to identify networks and host machines on those networks across the world. IPv4 addresses are 32 bits in length and typically expressed as four 8 bit decimal numbers separated by periods, e.g. `"192.168.0.1"`.

This 32 bit address consists of two parts: the network address and the host address. The network address is comprised of some number of bits at the beginning of the IPv4 address, the host address is then the remaining bits of the full address. For example, the network address of 192.168.0.1 may be the first two bytes (16 bits), in which case it would be expressed as 192.168.0.0. In this case, the host address would be the remaining two bytes, expressed as 0.0.0.1.

How many bits are in the network vs the host address components of the address are specified using the subnet mask, which is another 4 bytes. In a subnet mask, every bit of the original address that is part of the network address is a 1, and every bit in the host address is a 0. So the subnet mask for the previous example would be 255.255.0.0 in the typical notation.

Subnet masks make it easy to compute the network and host addresses of an IPv4 address by performing a bitwise AND on the two collections of bytes. Bitwise AND between two numbers compares each corresponding bit in the two numbers and the resulting bit in the final number is a 1 if the corresponding input bits are both 1s. Bitwise AND is typically performed using the `"&"` operator in most programming languages.

For this problem, given an IPv4 address and subnet mask. Output the network and host address components of the address. Hint: the "~" operator can be used to find the bitwise complement of a number in most programming languages.

Sample Input:
```
192.168.0.1
255.255.0.0
```

Sample Output:
```
Network: 192.168.0.0
Host: 0.0.0.1
```

Problem 2: Change is Inevitable

Point Value: 3 Points

Bounty: $30

Description: In the (tiny) kingdom of Larion, they use an ancient and revered system of coinage consisting of Pennings, Roobles, and Krowns. Write an algorithm that helps clerks at cash registers compute how much change they need to give their customers.

19 Pennings make a Rooble
13 Roobles make a Krown

Money is written with two decimal points separating the three types of coins. Each type of coin is always represented with two decimal numbers. Example:

$01.02.14 is one Krown, two Roobles, and 14 Pennings.

The first input given will be the price the customer needs to pay.
The second line of input will be what the customer paid with (They might pay with any mix of coins).

Print out the exact change they should receive. Give back as few coins as possible. (Giving 1 Krown and 2 Roobles is correct rather than 15 Roobles)

Sample Input 1:
```
$02.12.11
$03.00.00
```

Sample Output:
```
$00.00.08
```

Sample Input 2:
```
$13.04.02
$12.41.02
```

Sample Output:
```
$01.11.00
```

Problem 3: Reachability

Point Value: 4 Points

Bounty: $30

Description: You will be given a graph of **N** nodes labeled 1 through N. You will also be given the set of edges **E** which connect these nodes bi-directionally. Then, we will give you two nodes, **A** ,**B** ∈ **N** and you need to tell us whether or not it is possible to reach **A** from **B** using edges in **E**. Output either TRUE or FALSE.

The first line of input will contain **N**
The second line will be a list of edges **E**
The third line will be **A**
The fourth line will be **B**

Sample Input 1:

```
5
(1,2)(3,4)(1,4)(5,5)
1
3
```

Sample Output:

```
TRUE
```

Sample Input 2:

```
5
(1,2)(3,4)(5,4)(5,5)
1
3
```

Sample Output:

```
FALSE
```

Problem 4: Gearing

Point Value: 4 Points

Bounty: $30

Description:
A common problem when designing internal combustion cars is identifying the proper set of gear ratios in the transmission and differential so that at specific speeds and in certain gears, the engine is operating at a desirable rotational speed (RPM). Given a set of N gears ratios in a transmission, a differential gear ratio, and tire diameter (in inches), output a table for the engine operating speeds in rotations per minute for each gear at 10, 20, 30, 40, 50, 60, and 70 mph.

The formula for computing engine RPM at a particular speed is the following:

$$engineRPM = \frac{MPH * \frac{5280 feet}{1 mile} * \frac{12 inches}{1 foot} * \frac{1 hour}{60 minutes}}{(tireDiameter * \pi)} * ratio_{differential} * ratio_{gear}$$

The table should appear as shown below in the sample output. Final, RPM values should be rounded to the nearest whole number. Table columns should be 8 spaces wide.

Each line of input is as follows, an integer number of gears N, followed by N transmission gears, 1 differential gear, and one tire diameter.

Sample Input:

```
6
4.027
2.364
1.532
1.152
0.852
0.667
4.11
35
```

Sample Output:

Gear	10 mph	20 mph	30 mph	40 mph	50 mph	60 mph	70 mph
1	1590	3179	4769	6358	7948	9537	11127
2	933	1866	2799	3732	4666	5599	6532
3	605	1209	1814	2419	3024	3628	4233
4	455	909	1364	1819	2274	2728	3183
5	336	673	1009	1345	1682	2018	2354
6	263	527	790	1053	1316	1580	1843

Problem 5: Custom Sorting

Point Value: 5 Points

Bounty: $30

Description:
In this problem you will be asked to sort a collection of data records with multiple fields. The first line of input will specify all of the data fields for the data records to be sorted. Each field will have a label, which will contain only alphanumeric characters. After the field's label, the type of the data will be specified. Which will either be a `String` or an `Integer`. The next line of input will then specify how many data records are present in the following lines of input.

Each of the following records will be input as a space separated list of data values corresponding to the fields previously specified in the first line of input.

The final line of input will be a list of fields to sort the records by in order of precedence, i.e. the first appearing field should take the most precedence when sorting the list. In the case that two equal fields are encountered when sorting, the next listed field would then be used to determine the ordering. Each sorting field will also be paired with a D or an A after a colon. This indicates that the records should be sorted in Descending or Ascending order respectively.

Note that there will be at least one, but potentially all, of the data fields specified for sorting precedence. Strings should be sorted case insensitively. Integer values will be between 0-2^16.

Sample Input:
```
FirstName:String LastName:String Age:Integer NetWorth:Integer
6
Bill Gates 67 104
Larry Ellison 78 106
Elon Musk 51 180
Warren Buffet 92 106
Michael Bloomberg 81 94
Jeff Bezos 59 114
NetWorth:D LastName:A
```

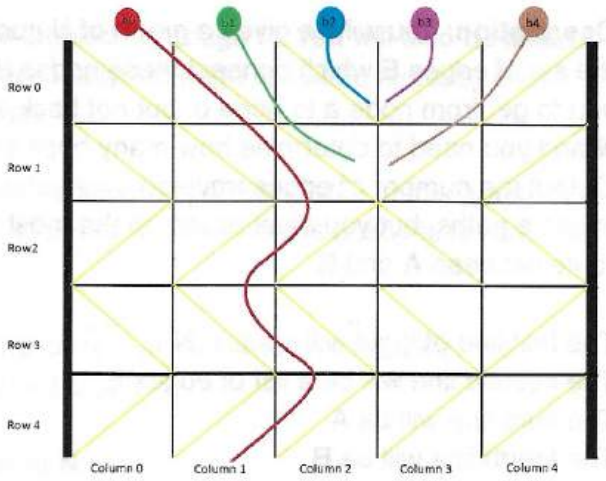
Sample Output:
```
Elon Musk 51 180
Jeff Bezos 59 114
Warren Buffet 92 106
Larry Ellison 78 106
Bill Gates 67 104
Michael Bloomberg 81 94
```

Problem 6: Falling Ball Game

Point Value: 6 Points

Bounty: $60

Description: You are playing a game similar to Plinko that is played on a 2-D grid with `r` rows and `c` columns, and you have `c` balls. The box is open on the top and bottom sides.



Each cell in the box has a diagonal board spanning two corners of the cell that can redirect a ball to the right or to the left.

You drop one ball at the top of each column of the box. Each ball can get stuck in the box or fall out of the bottom. A ball gets stuck if it hits a "V" shaped pattern between two boards or if a board redirects the ball into either wall of the box.

The first line of input consists of two numbers, `r` and `c`, the number of rows and columns respectively. The following `r` lines represent each row of `c` boards in the game box. A board that redirects the ball to the right spans the top-left corner to the bottom-right corner and is represented in the grid as 1. A board that redirects the ball to the left spans the top-right corner to the bottom-left corner and is represented in the grid as -1.

Output an array of `c` integers representing which column each of the balls dropped into each of the columns ends up at the bottom of the box. If a ball gets stuck, output -1 for that ball.

Sample Input 1:

```
5 5
1 1 1 -1 -1
1 1 1 -1 -1
-1 -1 -1 1 1
1 1 1 1 -1
-1 -1 -1 -1 -1
```

Sample Output 1:

```
1 -1 -1 -1 -1
```

Sample Input 2:

```
4 6
1 1 1 1 1 1
-1 -1 -1 -1 -1 -1
1 1 1 1 1 1
-1 -1 -1 -1 -1 -1
```

Sample Output 2:

```
0 1 2 3 4 -1
```

Problem 7: Hop Distance

Point Value: 6 Points

Bounty: $60

Description: You will be given a graph of **N** nodes labeled 1 through N. You will also be given the set of edges **E** which connect these nodes directionally. An edge specified as (a, b) allows you to get from node a to node b, but not back again. Then, we will give you two nodes, **A** ,**B** ∈ **N** and you need to determine how many hops are required to get to **A** from **B** using edges in **E**. Output the number of edges traveled over between A and B as a positive integer. There may be multiple paths, but you must report on the most efficient route. Output "-1" if there is no possible route between **A** and **B**.

The first line of input will contain **N**
The second line will be a list of edges **E**
The third line will be **A**
The fourth line will be **B**

Sample Input 1:
```
5
(1,2)(3,4)(1,4)(4,3)
1
3
```

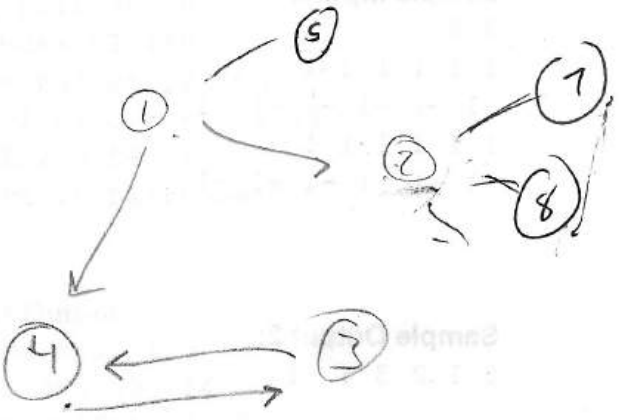
Sample Output 1:
```
2
```

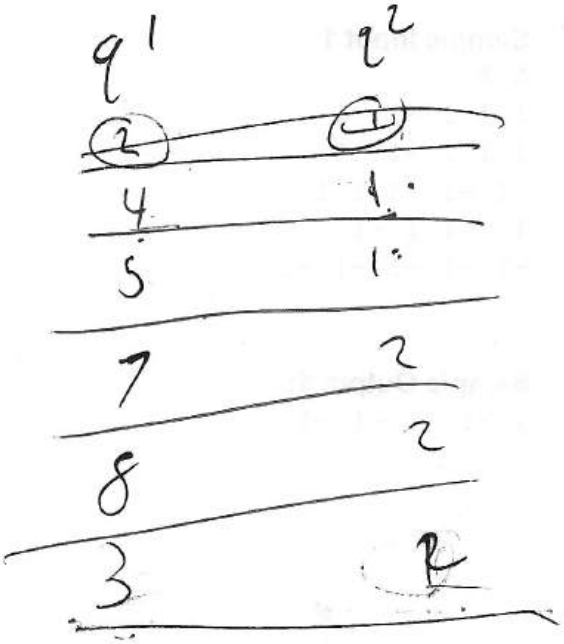
Sample Input 2:
```
5
(1,2)(3,4)(5,4)(3,1)
1
3
```

Sample Output 2:
```
-1
```

Sample Input 3:
```
5
(1,2)(2,3)(3,4)(1,4)
1
1
```

Sample Output 3:
```
0
```





Problem 8: Strongly Connected Components

Point Value: 9 points (Hard)

Bounty: $90

Description: You will be given a graph of **N** nodes labeled 1 through N. You will also be given the set of edges **E** which connect these nodes directionally. An edge specified as (a, b) allows you to get from node a to node b, but not back again.

Your job is to find the size of the largest strongly connected component (SCC) in this graph. A strongly connected component is defined as a set of nodes from which it is possible to reach all other nodes in the set.

The graph we specify may have multiple SCCs. The correct answer will be the size of the largest. Every graph necessarily has a SCC of at least size 1.

The first line of input will contain the number of nodes **N**
The second line will be a list of edges **E**

Sample Input 1:
```
07
(1,2)(2,3)(3,1)(3,5)
```

Sample Output:
```
3
```

Sample Input 2:
```
5
(1,2)(3,4)(5,4)(3,1)
```

Sample Output:
```
1
```

Sample Input 3:
```
5
(5,6)(1,2)(2,3)(3,4)(4,5)(6,7)(7,3)(3,2)
```

Sample Output:
```
6
```