ECOO 2016

Programming Contest Questions

Local Competition (Round 1)

March 30 – April 6, 2016

Sheridan | Get Creative

Questions made possible in part through the support of the Sheridan College Faculty of Applied Science and Technology.

Problem 1: Pass or Fail

In Ms. Echo's ICS4U class there are 4 components that determine a student's final grade: Tests, Assignments, Projects and Quizzes. She changes the weights on each of these components from year to year. Last year it was 20% tests, 20% assignments, 50% projects and 10% quizzes, but who knows what it will be this year? To pass the course, a student has to get 50% or more on the weighted average of all four components.

For example, last year Rosa got 98% on the tests, 85% on assignments, 76% on projects and 100% on the quizzes. That means her mark is:

$$98 \times 20\% + 85 \times 20\% + 76 \times 50\% + 100 \times 10\% = 19.6 + 17 + 38 + 10 = 84.6$$

Ms. Echo plays hardball – she never passes a student with less than 50%, even if that student got 49.9999%. All the marks are in for this year. How many students will be passing?

DATA11.txt (DATA12.txt for the second try) will contain 10 test cases. The first line of each test case contains four integers W_T, W_A, W_P and W_Q separated by spaces, representing the weights of the four components ($0 \leq W_T, W_A, W_P, W_Q \leq 100$ and $W_T + W_A + W_P + W_Q = 100$). This is followed by a line with a single integer N representing the number of students in the class ($1 \leq N \leq 35$). The next N lines each contain four integers T_i, A_i, P_i and Q_i, separated by spaces, representing the marks of an individual student (out of 100) for each component ($1 \leq i \leq N$ and $0 \leq T_i, A_i, P_i, Q_i \leq 100$). Your program should output a single integer for each test case representing the number of students who passed the course that year.

Note that the sample data below contains only 4 test cases but the test data will contain 10.

Sample Input

```
72 4 8 16
7
68 89 4 93
79 5 74 49
38 89 62 41
24 96 49 56
73 32 17 55
65 37 64 73
8 99 94 80
4 85 0 11
2
57 84 70 57
81 1 85 31
88 1 3 8
6
60 76 21 84
61 86 1 61
54 49 41 78
6 38 74 83
66 39 68 72
82 16 19 16
92 8 0 0
4
66 33 93 84
14 32 68 17
72 59 43 1
47 53 69 89
```

Sample Output

```
4
1
5
2
```

Problem 2: Spindie

In the game of "Spindie", players take turns spinning a spinner and rolling a die. On each turn, they spin the spinner three times and roll the die between each pair of spins (i.e. the sequence on a single turn is: Spin, Roll, Spin, Roll, Spin). Each spin of the spinner lands on some integer and each roll of the die results in an integer from 1 to 6. The first spinner number is the base score. Then if a die roll is 1 through 5, the player adds the next spinner number to their score. If they roll a 6, the next number is used to multiply their score. The winner is the player with the highest score after a set number of rounds.

Here are some example turns of Spindie:

Spin	Roll	Spin	Roll	Spin	Score
10	4	7	1	8	(10 + 7) + 8 = 25
1	3	2	6	5	(1 + 2) × 5 = 15
6	6	6	6	6	(6 × 6) × 6 = 216

DATA21.txt (DATA22.txt for the second try) will contain 10 test cases. The first line of each test case will consist of an integer N representing the number of integers on the spinner, where $1 \leq N \leq 5000$. The next line contains the N integers on the spinner, S_1 through S_N, separated by spaces, where $1 \leq S_i \leq 100$. The next line will contain five target integers T_1 through T_5 separated by spaces, where $1 \leq T_i \leq 1000000$. For each test case, your program should output a single line consisting of 5 letters. Each letter should represent one of the five targets (in order). If the target represents a possible score in a single round of Spindie, then output a T. If it is not possible, output an F.

Note that the sample data below contains only 5 test cases, but the test data will contain 10.

Sample Input

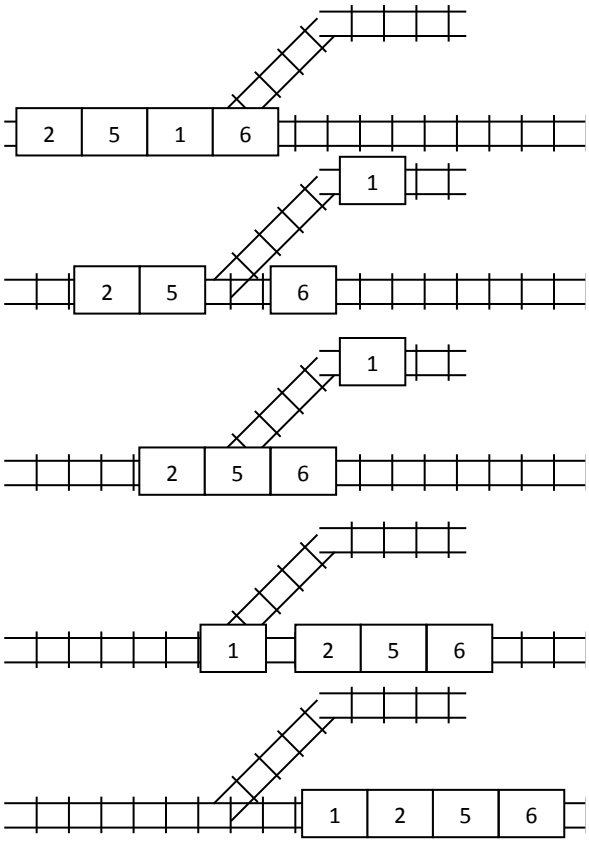
```
5
23 74 7 64 47
128605 205 2162 2709 71346
3
26 5 11
407 962 455 21 902
4
23 75 89 24
933 484 13248 102 44640
9
23 61 77 83 12 92 1 7 65
72900 144 5704 145 6370
7
87 20 94 99 14 26 87
241956 177 749331 221 4066
```

Sample Output

```
FFTFF
TTFTF
FFTFF
FTTTF
TFTFF
```

Problem 3: Railway Sort

The “Railway Sort” is a sorting algorithm based on the metaphor of arranging the cars of a train into the correct order. Your one move in a Railway Sort is to remove a “car” from the “train” by shunting it onto a side rail, then reattach the two pieces, move the train forward and reattach the car onto the back end of the train. The cost of such a move is proportional to the number of positions you moved the shunted train car.



In the example at the left, car 1 is shunted out and then reattached at the back end of the train (which is on the left because the train in this example is moving rightwards). The cost of this move is 2 because the car had to move 2 positions. The cost of sorting this train into ascending order is also 2 since the car numbers are now sorted correctly. If this sort had required more moves, the total cost would have been the sum of the cost of each of the moves.

DATA31.txt (DATA32.txt for the second try) will contain 10 test cases. The first line of each test case will consist of a single integer N indicating the number of cars in the train, where $1 \leq N \leq 1000$. The next line will contain all of the integers from 1 to N arranged in some random order, separated by spaces. Each integer will appear exactly once.

For each of the 10 test cases, you must print the minimum cost of performing a Railway Sort to arrange the integers into ascending order.

Note that the sample data below only contains 2 test cases but the test data will contain 10.

Sample Input

```
5
3 5 1 4 2
10
2 4 6 8 10 1 3 5 7 9
```

Sample Output

```
12
67
```

Problem 4: Kayenne

The village of Kayenne is divided into a square grid. Residents build their houses where the grid lines intersect. The location of each house is described by a pair of X and Y coordinates, where (0, 0) is the point at the exact center of town. Each grid line is exactly the same distance from the grid lines on either side of it. Kayenne stretches out from the center for 200 grid lines North, South, East and West. Individuals or families who move to Kayenne are assigned a circular area in which to build their house. This area is centered on a particular grid point and has a radius of 50. The newcomers are allowed to build their houses on any empty grid point within that circular area (including on the circle boundary).

Some households in Kayenne are Democrat and others are Republican. New households don't get to choose their political affiliations. Instead, their three closest neighbours (determined by the straight line distance between grid locations) vote on their affiliation and the newcomers must abide by the majority decision. Of course, Democrat households always vote that newcomers should be Democrats, and Republican households always vote that they should be Republicans. Occasionally there are ties for the 3rd closest neighbour. When this happens, all the neighbours who are the same distance as the 3rd closest are also allowed to vote. This can result in an even number of votes, which means that the vote can be tied. In these cases, the newcomers will be Democrats.

DATA41.txt (DATA42.txt for the second try) will contain 10 test cases. The first line of each test case will consist of two integers C_x and C_y, separated by a space, representing the coordinates of the center of a newcomer's circular building area ($-150 \leq C_x, C_y \leq 150$). The next 100 lines will each contain two integers H_x and H_y and an uppercase letter A, separated by a space, representing the coordinates and political affiliation of the existing houses in Kayenne ($-200 \leq H_x, H_y \leq 200, A =$ "D" *or* "R"). Your program must output a number representing the percentage chance (rounded to one decimal place) that a newcomer assigned to this circular building area will end up a Democratic household, assuming they choose their building site randomly from the available sites in their area.

Note that the sample data below contains only 1 test case with 20 houses but the test data will contain 10 test cases with 100 houses each.

Sample Input

```
-81 83
15 -198 R
-17 89 R
197 -174 R
-67 89 D
180 -101 D
-78 -173 R
182 121 D
-129 179 R
-100 -53 D
64 -61 D
-123 -152 D
-15 -67 R
20 194 D
125 16 D
133 -28 D
19 -9 R
121 168 D
165 -39 R
-170 -3 D
-27 61 D
```

Sample Output

```
88.7
```


Question Development Team

Sam Scott (Sheridan College)
Kevin Forest (Sheridan College)
Stella Lau (University of Cambridge)
Greg Reid (St. Francis Xavier Secondary School, Mississauga)
Dino Baron (University of Waterloo)
John Ketelaars (ECOO-CS Communications)
David Stermole (ECOO-CS President)