University of Central Florida

2024 Local Programming Contest (Qualifying Round)

Problems

Problem#	Difficulty Level	Filename	Problem Name
1	Easy	dozen	Farmers' Market
2	Easy	basket	Basketball Score
3	Easy-Medium	injury	Injured Shoulder
4	Easy-Medium	income	Income Inequality
5	Medium	whatorder	What's the Order Anyway?
6	Medium	dryland	Rectangular Dry Land
7	Medium-Hard	picap	Picture Caption
8	Hard	copogonia	Copogonia
9	Hard	weekend	Weekend Gardening

Call your program file: filename.c, filename.cpp, filename.java, or filename.py

For example, if you are solving Farmers' Market:

Call your program file: dozen.c, dozen.cpp, dozen.java, or dozen.py

Farmers' Market

filename: `dozen`
Difficulty Level: `Easy`
Time Limit: `5 seconds`

Dr. Meade went to the local Farmers' Market to get some fresh fruits. Most stores expected the customers to put fruits in bags and then the customers would be charged based on how much each bag weighs. Dr. M was exhausted from grading the final exams and, fortunately, he found a store where apples were already prepackaged!

All bags have the same number of apples (say, 5 apples in each bag) and you pick as many bags as you want. But, rather than charging you by the number of bags or their weights, the store charges you by how many dozens of apples you have! If your total count is not a multiple of dozens, you are still charged for dozens. For example, if each bag has 5 apples and you pick 4 bags (i.e., 20 apples total), you are charged for 24 (two dozens) apples since 20 is not a multiple of dozens.

The Problem:

Given how many apples there are in each bag, number of bags Dr. M has picked and the cost for a dozen of apples, determine how much Dr. M has to pay.

The Input:

The first input line contains two integers: a ($1 \le a \le 20$), indicating the number of apples in each bag and b ($1 \le b \le 20$), indicating how many bags Dr. M has picked. The second input line contains an integer, d ($1 \le d \le 20$), indicating the cost for a dozen of apples.

The Output:

Print how much Dr. M has to pay.

Sample Input	Sample Output
`5 4` `10`	`20`
`5 6` `15`	`45`
`16 3` `20`	`80`

Basketball Score

filename: `basket`
Difficulty Level: `Easy`
Time Limit: `5 seconds`

There are three types of shots in basketball: 1 pointer (free throw), 2 pointers, and 3 pointers.

The Problem:

Given how many shots of each type two teams made, determine the winner.

The Input:

The first input line contains three integers (each between 0 and 99 inclusive), indicating how many shots of each type $Team_1$ made. The first integer indicates 1-pointer shots, the second integer indicates 2-pointer shots, and the third integer indicates 3-pointer shots.

The second input line provides the shots made by $Team_2$, following the same format as the first input line.

The Output:

Print 1, 2 or 0 (zero), indicating which team won or if the game was a tie. The team with higher score (total points) wins. If both teams have the same total points, then the game is a tie.

Sample Input	Sample Output
`10 20 30` `50 10 10`	`1`
`5 5 5` `1 99 1`	`2`
`1 2 3` `0 1 4`	`0`

Explanation of the First Sample Input/Output:

Score (total points) for $Team_1$ is 140 and for $Team_2$ is 100, so $Team_1$ wins.

Injured Shoulder

filename: `injury`
Difficulty Level: `Easy-Medium`
Time Limit: `5 seconds`

Dr. Orooji injured his right shoulder a few months ago and could type with only one hand for a while. For some reasons, Dr. O missed the space character (blank) often and two words would be next to each other without the space (i.e., words concatenated) and the spell-checker would complain.

The Problem:

Given the words in a dictionary and a word Dr. O has typed, determine the status of the typed word.

The Input:

The first input line contains an integer, n ($2 \leq n \leq 20$), indicating the number of words in the dictionary. Each of the next n input lines contains a dictionary word. Each word starts in column 1, contains at least one lowercase letter, at most 20 lowercase letters, and no other characters. Assume that the dictionary words are unique, i.e., no duplicates.

The next input line contains an integer, m ($1 \leq m \leq 100$), indicating the number of words Dr. O has typed. Each of the next m input lines contains a typed word. Each typed word starts in column 1, contains at least one lowercase letter, at most 40 lowercase letters, and no other characters.

The Output:

For each typed word, output one line as follows:

> 1 – If the typed word is in the dictionary.
> 2 – If the typed word is a concatenation of two words in the dictionary, i.e., Dr. O missed typing the space character (blank).
> 0 (zero) – Otherwise.

Note that if a typed word is in the dictionary and it is also a concatenation of two words in the dictionary, output should be 1.

(Sample Input/Output on the next page)

Sample Input **Sample Output**

Sample Input	Sample Output
7 this is a test th the e 12 is test isthis the th ee eee thee best istestis e ea	1 1 2 1 1 2 0 2 0 0 1 2

Income Inequality

filename: `income`
Difficulty Level: `Easy-Medium`
Time Limit: `2 seconds`

According to Wikipedia, in 2021, the top 1% of households own 32.3% of the wealth in the United States. More generally, for any given percentage x, we could look at the data and state that the top x% of households own y% of the wealth.

It's obvious that for all societies with some inequality, it is always the case that $y > x$, except for $x = 0$ and $x = 100$. Of all possible choices for x, what is the maximum value of $y - x$?

The Problem:

Given the incomes of all people in a society, determine the maximum value of $y - x$ where the top x% of the people have y% of the society's wealth.

The Input:

The first input line contains a single integer, n ($2 \le n \le 10^6$), indicating the number of people in the society.

The second input line contains n integers; each integer, m ($1 \le m \le 10^{12}$), provides the wealth of a household in the society. Please note that the income values are not necessarily distinct.

The Output:

Print the maximum possible value of $y - x$, where the top x% of households in the society own y% of the society's wealth. Any answer within an absolute or relative error of 10^{-6} will be accepted.

Sample Input	Sample Output
4 11 1 2 6	35.000000
5 35 25 30 60 50	15.000000

What's the Order Anyway?

filename: `whatorder`
Difficulty Level: `Medium`
Time Limit: `2 seconds`

Alice, Bob, Carol, Denise, Eddie and Frank are up to their usual tricks. They're performing in the local comedy show but haven't told you the order in which they are presenting their stand-up routines. Instead, they've given you cryptic clues such as (with A standing for Alice, B for Bob, C for Carol, D for Denise, E for Eddie and F for Frank):

A > B

F < A

not C D

The first clue means that A is performing before B. The second clue means that F is performing after A. The third clue means that C and D are not performing consecutively. (In essence, if we were treating the acts as numbers, the inequality signs are assuming that the acts are sorted from greatest to least.)

Given the above clues, a possible order for the acts is Alice, Carol, Bob, Denise, Frank, Eddie.

What you've realized is that even with their clues, it might be impossible to pin down the exact order of their acts. Thus, you've settled for figuring out how many orderings are possible given the clues they've given you.

The Problem:

Given clues about the order of comedy acts, determine the number of valid possible orderings for the acts.

The Input:

The first input line contains two space-separated integers: n ($2 \leq n \leq 10$), indicating the number of comedy acts, and c ($1 \leq c \leq 10$), indicating the number of clues you've been provided. The acts are denoted by the first n uppercase letters.

The clues are provided in the following input lines, one clue per line. Each of these input lines has three space-separated pieces of information: a ($1 \leq a \leq 3$), x and y, as described below:

- x and y are guaranteed to be distinct uppercase letters out of the first n letters,
- a represents the type of restriction where 1 indicates that x's act comes before y's, 2 indicates that x's act comes after y's and 3 indicates that x's act and y's act don't occur consecutively (in either order).

It's guaranteed that the input clues won't be contradictory, and that there will be at least one valid ordering of the comedy acts.

The Output:

Print the number of different orders in which the comedy acts could be.

Sample Input **Sample Output**

Sample Input	Sample Output
6 3 1 A B 2 F A 3 C D	160
3 2 1 C A 2 B A	1

Rectangular Dry Land

filename: `dryland`
Difficulty Level: `Medium`
Time Limit: `7 seconds`

There are a lot of wetlands in Florida and finding space for building houses is a challenge. This problem is even more serious when people insist their houses to be rectangular shapes.

The Problem:

Let's assume our land is a two-dimensional (rectangular) grid and each grid cell is either wet or dry. We are interested in finding the largest rectangular dryland, i.e., we want to build the largest house (in area). Note that the house must be dryland in rectangular shape, i.e., the dry cells (area to build the house) must be rectangular and there can't be any wet cells in that rectangle.

The Input:

The first input line contains two integers: r ($1 \le r \le 200$), indicating the number of rows in our land and c ($1 \le c \le 200$), indicating the number of columns in our land. Each of the next r input lines specifies a row in the land; each line contains c characters (starting in column 1), each character being either 1 (wetland) or 0 (dryland).

The Output:

Print the area of the largest rectangular dryland.

Sample Input	**Sample Output**

`4 5`
`10010`
`01111`
`11110`
`00001`
	`6`

`3 2`
`10`
`11`
`11`
	`4`

`2 2`
`10`
`01`
	`1`

Picture Caption

filename: `picap`
Difficulty Level: `Medium-Hard`
Time Limit: `1 second`

The UCF Programming Team Lab (PTL) is located in HEC. The team photos are posted outside this lab. When preparing the caption for a picture, there is always the question of how many names should be listed on each line under the picture.

The Problem:

Given the number of names, number of lines to use for the picture caption, and number of letters in each name, we would like to minimize the length of the longest line in the caption.

Note that the names must appear in the caption in the order of the input, i.e., we don't want to reorder the names. Also note the following common constraints:

- No space before the first name on each caption line.
- No space after the last name on each caption line.
- Exactly one space between two consecutive names on each caption line.

The Input:

The first input line contains two integers n and k ($1 \le k \le n \le 10^5$), indicating (respectively) the number of names and the number of lines to use for the picture caption.

The second input line contains n integers c_i ($1 \le c_i \le 10^4$), representing the number of letters in each name in order.

The Output:

Print the length of the longest line in the caption, keeping in mind that we would like to minimize this value.

Sample Input	**Sample Output**
`7 4` `1 2 8 3 5 2 7`	`10`
`7 5` `1 2 8 3 5 2 7`	`8`
`7 2` `3 1 1 3 9 5 2`	`18`

Copogonia

filename: `copogonia`
Difficulty Level: `Hard`
Time Limit: `2 seconds`

Copogonia is a country with a set of ***n*** cities, which currently has n roads. The set of roads forms a convex polygon, hence the name of the country. Unfortunately, some of the citizens have been complaining about traveling times between cities. Due to the limited road system, there are always precisely two ways to travel between any pair of distinct cities (clockwise or counter-clockwise), and for some pairs of cities, even the shorter of the two possible routes seems like it takes too long.

After many rounds of careful surveys, it's been determined that the citizens would be happy if no individual travel distance between a pair of cities exceeded length ***m***, i.e., the path between any two cities has length m or less.

The government of Copogonia has done extensive research and decided that they could potentially build up to ***k*** new roads where each road directly connects two cities that were not previously connected via a direct road with a distance equal to the straight line distance between the two cities. Each road, however, comes with a cost of development. Naturally, the government of Copogonia would like to minimize the cost of development while keeping all of its citizens happy. They have tasked you with finding the precise subset of roads to develop to achieve their goals.

The Problem:

Given the locations on the Cartesian plane of the n cities of Copogonia (in the order they are connected with roads, noting that there is also a road from the last city to the first), a list of k potential additional roads that could be added (pair of cities and cost to build the road), and the maximum distance, m, that citizens are willing to travel on a single trip between pairs of cities, determine the minimal cost of developing a subset of roads that will keep the citizens of Copogonia happy.

The Input:

The first input line contains three space separated integers: n ($4 \le n \le 50$), indicating the number of cities in Copogonia, k ($1 \le k \le 10$), indicating the number of potential additional roads to build, and m, indicating the maximal travel distance between any pair of cities that the Copogonians are willing to tolerate.

Each of the next n input lines contains two space separated integers, x_i and y_i ($0 \le x_i$, $y_i \le 10^4$, $1 \le i \le n$), indicating the location of city i on the Cartesian grid (assume that no two cities are at the same location). There is a straight line road connecting city i to city i+1 ($1 \le i \le n - 1$) and a road connecting cities n and 1. Roads can be travelled in either direction.

The potential roads that can be added are provided in the next k input lines. Each of these k input lines contains three space separated integers: u , v ($1 \le u, v \le n, u \ne v$) and c ($1 \le c \le 10^8$), indicating that a road could be added between city *u* and city *v* at a cost of *c*. It's guaranteed that there's currently no road from city *u* to *v* and that no two of the potential roads listed will be between the same pair of cities.

Assume that:

- The input is such that at least one road must be added, i.e., with no extra roads the citizens are not happy (at least one of the shortest paths is greater than length *m*).
- If all of the potential roads are built, the citizens will be happy, i.e., all of the shortest paths will have length *m* or less.
- The value of *m* will be such that adding or subtracting 0.001 to *m* will not modify the answer.

The Output:

Print the minimum cost of a set of roads to build that would keep all the citizens of Copogonia happy.

Sample Input	**Sample Output**
4 2 15 0 0 0 10 10 10 10 0 1 3 1000 2 4 500	1500
5 3 2003 0 0 0 100 1000 100 2000 99 1000 0 2 4 50 4 1 123 3 5 47	170

Weekend Gardening

filename: `weekend`
Difficulty Level: `Hard`
Time Limit: `1 second`

Your roommate is planning some weekend projects to make your yard look better. The plan is to buy some plants. Your roommate wants to spend at least L dollars and at most H dollars. There are three different types of plants: cheap, normal, and expensive. Unfortunately, you cannot tell the difference by just looking at a plant. You know that each type has a different cost, and you know how many of each type your local nursery has in stock.

Now you are spending your time walking through the nursery aisles, picking out a plant, bringing it over to be rung up by the clerk, and praying that you stay within your desired total cost (your allotted budget). If you are not within your desired total cost (i.e., if your total cost is too low or too high), your roommate will get frustrated.

The Problem:

Given the cost and number of each plant type along with your desired budget range, determine the probability that you can buy plants one at a time and staying within your total cost range.

The Input:

The first input line contains two integers: L and H ($1 \le L \le H \le 10^9$), representing the low and high end of the budget, respectively.

The second input line contains three integers: C, N, and E ($1 \le C < N < E \le 10^9$), representing the cost of the cheap, normal, and expensive plants, respectively.

The third input line contains three integers: c, n, and e ($1 \le c, n, e \le 100$), representing the number of cheap, normal, and expensive plants, respectively.

The Output:

Print the probability that you will successfully complete the purchase within the desired budget by choosing plants one at a time. Your answer will be accepted if it is within 10^{-6} of the correct answer.

(Sample Input/Output on the next page)

Sample Input	**Sample Output**
30 60 5 15 70 3 1 1	0.2
40 60 5 15 70 3 1 1	0.0