South African Computer Olympiad
Training camp 3, 2004
Day 1

Overview

Author	**Nick Pilkington**	**Marco Gallotta**	**Richard Starfield**	**Linsen Loots**
Problem	**phone**	**sumfac**	**roads**	**primes**
Source	phone.pas phone.c phone.cpp	sumfac.pas sumfac.c sumfac.cpp	roads.pas roads.c roads.cpp	primes.pas primes.c primes.cpp
Input file	phone.in	sumfac.in	roads.in	primes.in
Output file	phone.out	sumfac.out	roads.out	primes.out
Time limit	1 second	2 seconds	2 seconds	1 second
Number of tests	10	10	10	10
Points per test	10	10	10	10
Total points	**100**	**100**	**100**	**100**

The maximum total score is 400 points.

South African Computer Olympiad
Training camp 3, 2004
Day 1

Telephone

Author

Nick Pilkington

Introduction

A long time ago in a galaxy far, far away, an Alien race was introduced to cellphones. Suddenly interplanetary smsing became hugely popular! The Alien race is such that any excess physical exercise causes them to explode; a most unfortunate consequence of being on a planet with no atmosphere. Each key of the Alien phones is assigned a certain number of letters of their alphabet. But because there are more letters in their alphabet than keys on the phone, extra presses are needed to generate some letters. Consider the keypad shown in Table 1, which has been translated into EarthSpeak:

1 ABC	2 DEF	3 GHI
4 JKL	5 MNO	6 PQR
7 STU	8 VW	9 XYZ

Table 1: An example Alien keypad

To generate a 'D' you'd simply press 2 once, however an 'E' would require two presses and an 'F' three and so forth...

The Alien race, fearing for their own survival, have recalled all phones on the planet and would like to consider the letter to key assignment so as to put the users through as little physical strain as possible. You have been outsourced to solve the problem of assigning the letters to keys of their phone.

Task

You will be told the number of keys on the Alien phone and the number of letters in the Alien alphabet. You will then be given the frequency that each of the letters appears in the language. You will have to calculate which letters are assigned to which keys such that the value of the keypad is minimized and adheres to the following constraints:

If read from left to right, top to bottom on the key pad the letters should appear in order. The value of the keypad is given by the sum of each letters frequency multiplied by its position on the key from the left.

The keys and letter will be given as numbers for simplicity!

Example

Consider a keypad with two buttons and 3 letters to assign to it. The first letter has frequency 2, the second has frequency 8, and the third has frequency 1. In this case, it would be best to assign the first letter to the first key, and the remaining two letters to the second key for a total value of $(1 \cdot 2) + (1 \cdot 8) + (2 \cdot 1) = 12$.

Input (phone.in)

- **Line 1:** Two positive integers K, L. K represents the number of keys on the phone $1..K$. L represents the number of letters in the alphabet $1..L$.
- **Line 2..L+1:** Each contains one integer F that represents the frequency of the corresponding letter corresponding to that line.

Sample input

```
2 3
2
8
1
```

Output (phone.out)

The output should contain K lines of output corresponding to the K keys on the phone. Each line should contain some number of space separated integer in increasing order that represent the letters assigned to this key.

Sample output

```
1
2 3
```

Constraints

- $1 \leq K < L \leq 1000$
- $1 \leq F \leq 1000$ for any frequency F

South African Computer Olympiad
Training camp 3, 2004
Day 1

50% constraints

- $1 \leq K < L \leq 30$
- $1 \leq F \leq 1000$ for any frequency F

Time limit

1 second.

Scoring

If the output is in any way invalid you will score 0 and be solely responsible for the spontaneous combustion of countless innocent Aliens. If the output reflects the optimal keypad you will score 100% and be offered a job on the planet without an atmosphere, cool!

South African Computer Olympiad
Training camp 3, 2004
Day 1

Sum Those Factors

Author

Marco Gallotta

Introduction

The Guji tribe has just invented an adding machine. The adding machine can only add numbers with 3 or fewer digits and the answer can only be 3 or fewer digits long. You have been trying to impress the Guji tribe by showing them modern appliances that have been around for a long time, but which they have never seen before. Up until now, you have failed to impress them. However, you are about to attempt to impress them again. This time you must use a computer to show them that you can add numbers as well as multiply numbers of longer length and produce a very long answer.

Task

To do the calculations, you must write a program. You must be able to add and multiply any numbers that the Guji tribe supply you with and must be able to solve their sum exactly. The calculations must be done in the usual order, i.e. multiply out the factors and then add the terms. Every digit must be correct, since, if you make a mistake with just one digit, the whole of the Guji tribe will think that you are a mad man and never listen to you again.

Example

You have three values to work with. The first value is a 5, which must be multiplied by the second value, 345. The result is $5 * 345 = 1725$. Then the third value, a 6, must be added to the 1725 to give a final answer of $1725 + 6 = 1731$.

Input (sumfac.in)

The first line contains a single integer N, the number of values. The second line contains the sum to be calculated. The sum will include N integers. Each integer, except for the last, will be followed by an operation, either a '+' or a '*'. All the integers and operations will be space separated and will all appear on a single line of the input file.

Sample input

```
3
5 * 345 + 6
```

Output (sumfac.out)

The only line of the output file should contain a single integer, the solution to the sum. The solution must be output in the format shown below, with a comma separating each set of three digits.

Sample output

```
1,731
```

Constraints

- $1 \leq N \leq 3000$
- $0 \leq X_i < 1000000$

Time limit

2 seconds.

Scoring

- You will score 0% for an incorrect answer or incorrect formatting of output, and
- 100% for a correct answer.

South African Computer Olympiad
Training camp 3, 2004
Day 1

Road Block

Author

Richard Starfield

Introduction

A small island community has recently been enjoying an upsurge in technological and urban growth. Unfortunately the spread of cities has lead to the spread of crime, and allowed one crimelord in particular to rise to notoriety.

This crimelord, known only as 'The Ijug', is rumoured to be hiding in one of several villages; biding his time until a certain secret project is completed. This project, codenamed L.O.Y.E.R. will supposedly render The Ijug impervious to justice.

The police have therefore decided that The Ijug must be prevented from reaching this secret project at all costs. Fortunately the secret project is being developed in another village; unfortunately, the police have only managed to narrow this down to one of several possibilities. It is assumed that The Ijug will shortly attempt to travel to one of these villages. The police would like to place the minimal amount of roadblocks necessary to guarantee his apprehension.

Task

Since they have limited resources, the police have turned to you to help them decide which roads to block. You will be given the number of villages N, the set of R roads between villages and two lists of villages: L_I where the Ijug may be hiding and L_P, where it is suspected the Project is being developed. You must find the minimal set of roads whose blockage will prevent access from any of the villages in L_I to any of the villages in L_P.

Example

Suppose there are villages numbered 1–4 connected clockwise in a circle. If the Ijug may be in villages 1 or 2 and the Project is suspected to be in village 3, then blocking the road between villages 2 and 3, and 1 and 4 would be sufficient.

Input (roads.in)

The first line of the input contains one integer, N, the number of villages. All villages will be numbered from 1–N. The second line contains one integer, I, the number of different villages in the set L_I. The next line contains I space-separated integers; the id numbers of these villages. The next line contains one integer, P, the number of villages in the set L_P. The next line contains P space-separated integers; the id numbers of these villages.

The next line contains one integer, R, the number of roads between villages. The next R lines each contain two space separated integers, V_1 and V_2, the villages connected by that road. All roads are bidirectional.

Sample input

```
4
2
1 2
1
3
4
1 2
2 3
3 4
4 1
```

Output (roads.out)

The first line of the output should contain one integer, B, the number of roads which must be blocked. The next B lines should contain two space-separated integers, V_1 and V_2, which are the villages whose connecting road needs to be blocked. Each road should be mentioned at most once. The order of the roads is immaterial, as is the order in which villages are mentioned when describing a road. Extraneous whitespace will not affect the validity of a solution.

Sample output

```
2
1 4
2 3
```

Constraints

- $2 \leq N \leq 250$
- $1 \leq I \leq 100$
- $1 \leq P \leq 100$

- $1 \leq R \leq N^2$
- $1 \leq V_1, V_2 \leq N$
- $0 \leq B \leq R$

50% constraints

- $2 \leq N \leq 25$
- $1 \leq I \leq 10$
- $1 \leq P \leq 10$
- $1 \leq R \leq 50$

Time limit

2 seconds.

Scoring

- If after removing the set of roads specified in the output, a path exists between any of the villages in L_I and any of the villages in L_P, you will score 0.
- Otherwise you will score $10 * (M/B)$, rounded down where M is the minimal number of roadblocks required.

Primes

Author

Linsen Loots

Introduction

Peter was given a list of very important prime numbers by his dying father, the code to the family vault. Concerned about forgetting them, he saved the list on his computer. Unfortunately the hard drive crashed, and all he was able to recover was the digits it contained.

Task

Peter has asked your help in finding out the values of the prime numbers. Luckily he can remember that they all had 5 digits and that they were consecutive. Given the statistics about them, namely how many 0s there were, how many 1s, how many 2s, and so on, find out what the prime numbers were.

Example

Peter knows that there were four 1s, six 3s, three 6s and two 9s. The original prime numbers were thus 16319 16333 and 16339.

Input (primes.in)

The input file consists of a single line containing ten space-separated integers. The first is the number of zeroes, the second the number of ones, and so on.

Sample input

```
0 4 0 6 0 0 3 0 0 2
```

Output (primes.out)

The first line of the output file should contain a single integer N, the number of primes in the list. The next line should contain a single integer P, the smallest of these primes.

Sample output

```
3
16319
```

Constraints

There will never be more than 250 of a single digit.

50% constraints

There will not be more than 40 of a single digit.

Time limit

1 second.

Scoring

- 1 for the correct value of N
- 9 for the correct smallest prime