



The CENTRE for EDUCATION  
in MATHEMATICS and COMPUTING

*2015  
Canadian  
Computing  
Competition:  
Senior  
Division*

Sponsor:

**WATERLOO**  
**MATHEMATICS**

## Problem J5: $\pi$ -day

### Problem Description

You may know that March 14 is known as “ $\pi$ -day”, since 3.14 (which is the third month and fourteenth day) is a good approximation of  $\pi$ .

Mathematicians celebrate this day by eating pie.

Suppose that you have  $n$  pieces of pie, and  $k$  people who are lined up for pieces of pie. All  $n$  pieces of pie will be given out. Each person will get at least one piece of pie, but mathematicians are a bit greedy at times. So, they always get at least as many of pieces of pie as the person in front of them.

For example, if you have 8 pieces of pie and 4 people in line, you could give out pieces of pie in the following five ways (with the first person in line being the first number in the list):  $[1, 1, 1, 5]$ ,  $[1, 1, 2, 4]$ ,  $[1, 1, 3, 3]$ ,  $[1, 2, 2, 3]$ ,  $[2, 2, 2, 2]$ .

Notice that if  $k = n$ , there is only one way to give out the pieces of pie: every person gets exactly one piece. Also, if  $k = 1$ , there is only one way to give out the pieces of pie: that single person gets all the pieces.

Write a program that determines the number of ways that the pieces of pie can be given out.

### Input Specification

The first line of input is the integer number of pieces of pie,  $n$  ( $1 \leq n \leq 250$ ).

The second line of input is the integer  $k$  which is the number of people in line ( $1 \leq k \leq n$ ).

For at least 20% of the marks for this problem,  $n \leq 9$ . For at least 50% of the marks for this problem,  $n \leq 70$ . For at least 85% of the marks for this problem,  $n \leq 120$ .

### Output Specification

The output will consist of a single integer which is the number of ways that the pieces of pie can be distributed. The output is guaranteed to be less than  $2^{31}$ .

### Sample Input 1

8  
4

### Output for Sample Input 1

5

### Sample Input 2

6  
2

### Output for Sample Input 2

3

# Problem S1: Zero That Out

## Problem Description

Your boss has asked you to add up a sequence of positive numbers to determine how much money your company made last year.

Unfortunately, your boss reads out numbers incorrectly from time to time.

Fortunately, your boss realizes when an incorrect number is read and says “zero”, meaning “ignore the current last number.”

Unfortunately, your boss can make repeated mistakes, and says “zero” for each mistake.

For example, your boss may say “One, three, five, four, zero, zero, seven, zero, zero, six”, which means the total is 7 as explained in the following chart:

Boss statement(s)	Current numbers	Explanation
“One, three, five, four”	1, 3, 5, 4	Record the first four numbers.
“zero, zero”	1, 3	Ignore the last two numbers.
“seven”	1, 3, 7	Record the number 7 at the end of our list.
“zero, zero”	1	Ignore the last two numbers.
“six”	1, 6	We have read all numbers, and the total is 7.

At any point, your boss will have said at least as many positive numbers as “zero” statements. If all positive numbers have been ignored, the sum is zero.

Write a program that reads the sequence of boss statements and computes the correct sum.

## Input Specification

The first line of input contains the integer  $K$  ( $1 \leq K \leq 100\,000$ ) which is the number of integers (including “zero”) your boss will say. On each of the next  $K$  lines, there will either be one integer between 1 and 100 (inclusive), or the integer 0.

## Output Specification

The output is one line, containing the integer which is the correct sum of the integers read, taking the “zero” statements into consideration. You can assume that the output will be an integer in the range 0 and 1 000 000 (inclusive).

## Sample Input 1

4  
3  
0  
4  
0

**Output for Sample Input 1**

0

**Sample Input 2**

10

1

3

5

4

0

0

7

0

0

6

**Output for Sample Input 2**

7

## Problem S2: Jerseys

### Problem Description

A school team is trying to assign jerseys numbered  $1, 2, 3, \dots, J$  to student athletes. The size of each jersey is either small (S), medium (M) or large (L).

Each athlete has requested a specific jersey number and a preferred size. The athletes will not be satisfied with a jersey that is the wrong number or that is smaller than their preferred size. They will be satisfied with a jersey that is their preferred size or larger as long as it is the right number. Two students cannot be given the same jersey.

Your task is to determine the maximum number of requests that can be satisfied.

### Input Specification

The first line of input is the integer  $J$  which is the number of jerseys.

The second line of input is the integer  $A$  which is the number of athletes.

The next  $J$  lines are each the character S, M or L. Line  $j$  gives the size of jersey  $j$  ( $1 \leq j \leq J$ ).

The last  $A$  lines are each the character S, M or L followed by a space followed by an integer. Line  $a$  ( $1 \leq a \leq A$ ) gives the requested size and jersey number for athlete  $a$  where the athletes are numbered  $1, 2, 3, \dots, A$ .

For 50% of the test cases,  $1 \leq J \leq 10^3$  and  $1 \leq A \leq 10^3$ .

For the remaining 50% of the test cases,  $1 \leq J \leq 10^6$  and  $1 \leq A \leq 10^6$ .

### Output Specification

The output will consist of a single integer which is the maximum number of requests that can be satisfied.

### Sample Input

```
4
3
M
S
S
L
L 3
S 3
L 1
```

### Output for Sample Input

```
1
```

### Explanation Sample Output

Jersey 1 cannot be assigned because it is medium and athlete 3 requested large. No athlete requested jersey 2 or 4. Jersey 3 (small) can be assigned athlete 2 (small) but not athlete 1 (large).

## Problem S3: Gates

### Problem Description

For your birthday, you were given an airport.

The airport has  $G$  gates, numbered from 1 to  $G$ .

$P$  planes arrive at the airport, one after another. You are to assign the  $i$ th plane to permanently dock at any gate  $1, \dots, g_i$  ( $1 \leq g_i \leq G$ ), at which no previous plane has docked. As soon as a plane cannot dock at any gate, the airport is shut down and no future planes are allowed to arrive.

In order to keep the person who gave you the airport happy, you would like to maximize the number of planes starting from the beginning that can all dock at different gates.

### Input Specification

The first line of input contains  $G$  ( $1 \leq G \leq 10^5$ ), the number of gates at the airport.

The second line of input contains  $P$  ( $1 \leq P \leq 10^5$ ), the number of planes which will land.

The next  $P$  lines contain one integer  $g_i$  ( $1 \leq g_i \leq G$ ), such that the  $i$ th plane must dock at some gate from 1 to  $g_i$ , inclusive.

Note that for at least 40% of the marks for this question,  $P \leq 2000$  and  $G \leq 2000$ .

### Output Specification

Output the maximum number of planes that can land starting from the beginning.

### Sample Input 1

```
4
3
4
1
1
```

### Output for Sample Input 1

```
2
```

### Explanation of Output for Sample Input 1

The first plane can go anywhere, but it is best to not put it into Gate 1. Notice that planes 2 and 3 both want to dock into Gate 1, so plane 3 is unable to dock.

### Sample Input 2

```
4
6
2
```

2  
3  
3  
4  
4

### **Output for Sample Input 2**

3

### **Explanation of Output for Sample Input 2**

The first two planes will dock in gates 1 and 2 (in any order). The third plane must dock at Gate 3. Thus, the fourth plane cannot dock anywhere, and the airport is closed, even though plane 5 would have been able to dock.

## Problem S4: Convex Hull

### Problem Description

You are travelling on a ship in an archipelago. The ship has a convex hull which is  $K$  centimetres thick. The archipelago has  $N$  islands, numbered from 1 to  $N$ . There are  $M$  sea routes amongst them, where the  $i$ th route runs directly between two different islands  $a_i$  and  $b_i$  ( $1 \leq a_i, b_i \leq N$ ), takes  $t_i$  minutes to travel along in either direction, and has rocks that wear down the ship's hull by  $h_i$  centimetres. There may be multiple routes running between a pair of islands.

You would like to travel from island  $A$  to a different island  $B$  ( $1 \leq A, B \leq N$ ) along a sequence of sea routes, such that your ship's hull remains intact – in other words, such that the sum of the routes'  $h_i$  values is strictly less than  $K$ .

Additionally, you are in a hurry, so you would like to minimize the amount of time necessary to reach island  $B$  from island  $A$ . It may not be possible to reach island  $B$  from island  $A$ , however, either due to insufficient sea routes or the having the ship's hull wear out.

### Input Specification

The first line of input contains three integers  $K$ ,  $N$  and  $M$  ( $1 \leq K \leq 200$ ,  $2 \leq N \leq 2000$ ,  $1 \leq M \leq 10000$ ), each separated by one space.

The next  $M$  lines each contain 4 integers  $a_i$   $b_i$   $t_i$  and  $h_i$  ( $1 \leq a_i, b_i \leq N$ ,  $1 \leq t_i \leq 10^5$ ,  $0 \leq h_i \leq 200$ ), each separated by one space. The  $i$ th line in this set of  $M$  lines describes the  $i$ th sea route (which runs from island  $a_i$  to island  $b_i$ , takes  $t_i$  minutes and wears down the ship's hull by  $h_i$  centimetres). Notice that  $a_i \neq b_i$  (that is, the ends of a sea route are distinct islands).

The last line of input contains two integers  $A$  and  $B$  ( $1 \leq A, B \leq N$ ;  $A \neq B$ ), the islands between which we want to travel.

For 20% of marks for this question,  $K = 1$  and  $N \leq 200$ . For another 20% of the marks for this problem,  $K = 1$  and  $N \leq 2000$ .

### Output Specification

Output a single integer: the integer representing the minimal time required to travel from  $A$  to  $B$  without wearing out the ship's hull, or  $-1$  to indicate that there is no way to travel from  $A$  to  $B$  without wearing out the ship's hull.

### Sample Input 1

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



```
1 4 6 12
1 4
```

### **Output for Sample Input 1**

7

### **Explanation of Output for Sample Input 1**

The path of length 1 from 1 to 4 would wear out the hull of the ship. The three paths of length 2 ([1, 2, 4] and [1, 3, 4] two different ways) take at least 8 minutes. The path [1, 2, 3, 4] takes 7 minutes and only wears down the hull by 7 centimetres, whereas the path [1, 3, 2, 4] takes 13 minutes and wears down the hull by 5 centimetres.

### **Sample Input 2**

```
3 3 3
1 2 5 1
3 2 8 2
1 3 1 3
1 3
```

### **Output for Sample Input 2**

-1

### **Explanation of Output for Sample Input 2**

The direct path [1, 3] wears down the hull to 0, as does the path [1, 2, 3].

## Problem S5: Greedy For Pies

### Problem Description

The local pie shop is offering a promotion - all-you-can-eat pies! Obviously, you can't pass up this offer.

The shop lines up  $N$  pies from left to right - the  $i$ th pie contains  $A_i$  grams of sugar. Additionally, another  $M$  pies are provided - the  $i$ th of these contains  $B_i$  grams of sugar.

You are first allowed to insert each of the  $M$  pies from the second group anywhere into the first list of  $N$  pies, such as at its start or end, or in between any two pies already in the list. The result will be a list of  $N + M$  pies with the constraint that the initial  $N$  pies are still in their original relative order.

Following this, you are allowed to take one walk along the new line of pies from left to right, to pick up your selection of all-you-can-eat pies! When you arrive at a pie, you may choose to add it to your pile, or skip it. However, because you're required to keep moving, if you pick up a certain pie, you will not be able to also pick up the pie immediately after it (if any). In other words, you cannot eat consecutive pies in this combined list.

Being a pie connoisseur, your goal is to maximize the total amount of sugar in the pies you pick up from the line. How many grams can you get?

### Input Specification

The first line of input contains the integer  $N$  ( $1 \leq N \leq 3000$ ). The next  $N$  lines contain one integer  $A_i$  ( $1 \leq A_i \leq 10^5$ ), describing the integer number of grams of sugar in pie  $i$  in the group of  $N$  pies.

The next line contains  $M$  ( $0 \leq M \leq 100$ ), the number of pies in the second list. The next  $M$  lines contain one integer  $B_i$  ( $1 \leq B_i \leq 10^5$ ), describing the integer number of grams of sugar in pie  $i$  in the group of  $M$  pies.

For 20% of the marks for this question,  $M = 0$ . For another 20% of the marks for this question  $M = 1$ . For another 20% of the marks for this question  $M \leq 10$ .

### Output Specification

Output the maximum number of grams of sugar in all the pies that you are able to pick up.

### Sample Input

```
5
10
12
6
14
7
```

3  
1  
8  
2

### **Output for Sample Input**

4 4

### **Explanation of Output for Sample Input**

Place the pies in the order

10, 1, 12, 2, 8, 6, 14, 7

(that is, insert the pie with 1 gram of sugar between 10 and 12, and insert pies with 2 and 8 grams of sugar, in that order, between pies 12 and 6). Then, we can grab  $10 + 12 + 8 + 14 = 44$  grams of sugar, which is maximal.