Homework #4 Increasing Pairs

(due: 11:59pm 11/27/2021)

Overview

In this homework, we define a special pair named increasing pair, in which the latter number is at least two times larger than the former one. The formal definition is as described below.

given index i, j where $0 \leq i < j <$ the length of the sequence

if $2 \times seq[i] \leq seq[j]$, we call ($seq[i]$, $seq[j]$) increasing pair.

You need to calculate the number of increasing pairs in the sequence. For instance, given sequence

1 6 13 4 2

There are three increasing pairs (1, 6), (1, 13), (1,4), (1, 2) and (6, 13). So, you need to print 5 in this case.

The format of `input.txt` and `output.txt` are given as follows.

Input.txt

In the first line, the number T means the number of test cases. Then, you will receive T lines of the test cases. There will be integer Ni in the head of each line which indicates the number of integers in the following sequence.

```
T
N1 sequence_1
...
Ni sequence_i
```

Output.txt

In the file output.txt, you need to print the number of increasing pairs in each sequence.

```
the number of the pairs in sequece1
…
the number of the pairs in sequece2
```

Example1

Input:

```
1
```

```
5 4 0 6 1 -1
```

The number 1 in the first line indicates that there is one test case in input.txt. The first number, 5 in the second line, means there are five integers in sequence 1 (4, 0, 6, 1, -1).

Result:

```
2
```

There are two pairs (0, 6) and (0, 1).

Constraints

1. 1 <= the number of integers in the sequence <= 500000
2. $-2^{31} \leq$ integer in the sequence $< 2^{31}$.

Preloaded Input Data

```
struct tTestData {
    int testcase_num;
    int seq_num[testcase_num];
    int seq[testcase_num][50000];
};
```

testcase_num: the number of test cases

seq_num: the number of integers in the sequence (for example: seq_num[0] means the number of integers in the sequence 1)

seq: each sequence (for example: seq[0][0] indicates the first integer in the sequence 1)

Hint: Be aware of integer overflow.

Submission of Homework

1. Download hw4.zip in the attachment.
2. Upload your code to Code Sensor.
3. If you choose c++ as your language, don't use the following library calls:
   mmap, mprotect, munmap, syscall, fork, vfork, clone, system, creat, open, mknod, mknodat, fopen
4. Use the “View” and “Scoreboard” function on CODE SENSOR to make sure your submission is successful.
5. Feel free to submit your questions to Forum on e3 or send e-mail to TA.