Test question

This problem will not count towards your final score.

PROBLEM STATEMENT

You are given a thousand pairs of integers a and b. Output the sum of each pair.

INPUT

There are 1000 sets of inputs.

Each set of inputs contains 2 lines.

- The first line is the integer a. The second line is the integer b.

OUTPUT

1000 lines, where each line is the output $a + b$ of the corresponding set of inputs.

CONSTRAINTS

$-10^{18} \leq a, b \leq 10^{18}$

BONUS

Since `grhkm` is feeling nice today, here is a solution written in `Python`:

```python
# Iterate 1000 times
for _ in range(1000):
    # Read the input
    a = int(input())
    b = int(input())
    # Output the answer
    print(a + b)
```

To run this on your machine, type `python3 q0.py < 1.in > 1.out`. Replace `q0.py` with your file name, `1.in` and `1.out` with the correct input and output files.

Here is the solution code in `C++`:

```cpp
#include <iostream>

int main() {
  long a, b;
  for (long T = 1; T <= 1000; T++) {
    std::cin >> a >> b;
    std::cout << a + b << std::endl;
```

```
  }
}
```

To run this on your machine, type `g++ q0.cpp -Wall -o q0 && ./q0 < 1.in > 1.out`. Replace `q0.cpp` with your file name, `1.in` and `1.out` with the correct input and output files.

CabbageMeet

PROBLEM STATEMENT

The UWCS exec are trying to organise a meeting! However, everyone is "busy" (right), so they are struggling to find a timeslot where everyone is free.

As the developer of **CabbageMeet**, a group scheduling software, you must help the exec. More specifically, there are n execs trying to schedule a meeting in the next 30 days. You are given each exec's availability for the next 30 days. Find the days on which the maximum number of exec are available.

INPUT

There are 1000 sets of inputs.

Each set of inputs begins with a line with an integer n, the number of exec scheduling a meeting.

Then, n line follows. Each line consists of 30 characters, each being either `.` or `X`.

- If the i-th character is `.`, it means the exec is **available** on the i-th day.
- Otherwise, the exec is **unavailable** on the i-th day.

OUTPUT

1000 lines, where each line consists of space-separated integers $i_1, \cdots, i_k$, representing the days of which the maximum number of execs can join the meeting.

CONSTRAINTS

For `1_small.in`, $n = 100$.

For `2_large.in`, $n = 1000$.

For `3_humongus.in`, $n = 10000$.

Painting (over) Hearts

TODO: Make this Warwick related?

PROBLEM STATEMENT

As we all know, Coventry is one of the most romantic cities in the UK. This was mostly caused by miles and miles of graffities on the wall. You, single af, has been tasked to remove these graffities.

Figure 1: wall

There is a N-meters-long wall with k graffities on it. To ease notation, we can identify it with the segment $[0, N]$, and divide it into segments $[0, 1], [1, 2] \cdots, [N - 1, N]$. Each graffiti can simply be described as a continuous interval $[L_i, R_i]$. Note that the intervals describing the graffities may intersect.

You also own a paint brush that is P-meters wide. With it, you can paint over *any* continuous interval $[x, x + P]$, removing any paintings underneath. Your task is to find how many paint strokes are required to paint over all paintings.

For example, say the wall is $N = 20$ meters long, with $k = 4$ graffities on it, located at $[L_i, R_i] = [1, 3], [4, 6], [10, 13]$ and $[11, 18]$, respectively. With a paint brush that is $P = 7$ meters wide, you are able to cover the first two graffities using a single stroke at $[1, 8]$. However, you *must* use two separate strokes to cover the remaining two graffities, for example at $[10, 17]$ and $[11, 18]$. This is demonstrated below. Hence the answer, is 3.

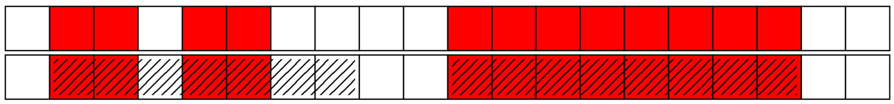

Figure 2: pic

INPUT

There are 100 sets of inputs.

Each set of inputs begins with a line with three spaced-separated integers N, k and P, as defined above.

Then, k lines follow, each containing the two integers L_i and R_i separated by space.

OUTPUT

100 lines, each containing a single integer, being the number of strokes required to cover all graffities on the wall with your paint brush.

CONSTRAINTS

For all test cases, $0 \leq L_i \leq R_i \leq N$ and $P \leq N$.

For `1_small.in`, $N \leq 1000$ and $k \leq 100$.

For `2_not_small.in`, $N \leq 10^4$ and $k \leq 10^3$.

For `3_hard_disjoint.in`, $N \leq 10^6$ and $k \leq 10^5$. It is guaranteed that the intervals $[L_i, R_i]$ are disjoint, i.e. the graffities do not overlap.

For `4_hard.in`, $N \leq 10^9$ and $k \leq 10^5$.

Magic Tricks (easy ver.)

Our Lord Chancellor of the Computers, Mat, loves magic tricks. His favourite magic trick involves shuffling a deck of n cards. However, his obsession with a magic number k forces him to shuffle in a very specific way.

PROBLEM STATEMENT

Each time he shuffles, he performs a specific sequence of *Riffle shuffles* and *cuts*:

- When Mat performs a *Riffle shuffle*, he separates the deck into two halves, **top** and **bottom**. He then interlace them in alternating order, starting with the bottom card of the **top** deck. Counting the cards from the bottom, the resulting deck of cards start with the 1st card of the **top** deck, then the 1st card of the **bottom** deck, then the 2nd card of the **top** deck, etc.
- When Mat performs a *cut*, he takes the top k cards and place it under the rest while preserving the order.

For example, say there are $n = 8$ cards at the start, numbered $1, 2, \cdots, 8$ starting from the bottom. A Riffle shuffle will transform the deck into $5, 1, 6, 2, 7, 3, 8, 4$. If Mat's magic number is $k = 3$ and he performs a cut, then the deck will become $3, 8, 4, 5, 1, 6, 2, 7$.

To ensure that the deck is well-shuffled, Mat performs each operation multiple times. For example, a possible shuffling operation would be "shuffle 20 times, then cut 7 times, and shuffle 13 times." See INPUT and SAMPLES below.

Write a program that simulates this process.

INPUT

There are 1000 sets of input.

Each set of inputs begins with a line with three integers n, k and q. n is the number of cards in the deck, and k is Mat's magic number.

Then, q lines follow. Each line contains an integer x and a string s, either `shuffle` or `cut`. You shall repeat the operation s a total of x times.

OUTPUT

1000 lines, where each line is n space-separated integers, being the numbers of each card in the shuffled deck from bottom to top.

CONSTRAINTS

$x, k \leq n \leq 10^5$, $\sum n \leq 10^6$, $q \leq 10^3$.

Smashing Balls

`grhkm` watched Wrecking Ball and read a copypasta on the UWCS discord:

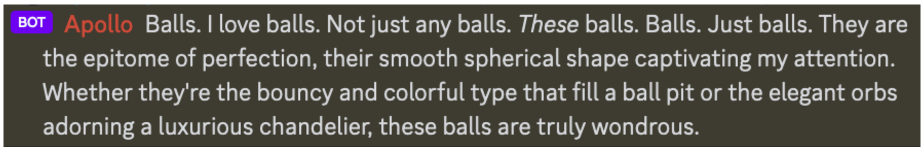

Figure 1: balls

He was very inspired, so he decided to write a problem about it.

PROBLEM STATEMENT

There is a line of n balls in front of you. The balls are massive - they have masses $m_1, \cdots, m_n$ tonnes respectively! It is difficult to carry so many balls around, so `grhkm` grants you the power to do the following operation any number of times:

- Take two adjacent balls and smash them together. They will combine into a bigger ball with the mass added.

`grhkm` loves seeing people smash big balls, so he decided to award you with coins each time you smash balls. If you smash two balls with masses m_1 and m_2 tonnes, then you are awarded with $m_1 + m_2$ coins!

For example, if you start with $n = 4$ balls with masses $2, 5, 3, 1$ tonnes:

- First, you can smash the balls with masses $5, 3$ together, leaving three balls with $2, 8, 1$ tonnes respectively and gaining 8 coins.
- Next, you can smash the balls with masses 2 and 8 tonnes together, gaining 10 coins and leaving two balls with 10 and 1 tonnes respectively.
- Finally, you smash the remaining two balls, resulting with a ball of 11 tonnes and gaining 11 coins.

You end up with $8 + 10 + 11 = 29$ coins. Of course, with a different order of smashes, you will end up with different amount of coins. However, it is possible to show that 29 coins is the maximum attainable.

Your task is to compute the minimum and maximum coins you can attain!

INPUT

There are 1000 sets of input.

Each set of inputs consists of two lines.

The first line of the set is a single integer n, the number of balls in the line.

The next line contains n space-separated integers $m_1, \cdots, m_n$, the masses of the balls.

OUTPUT

1000 lines, where each line is two integers separated by spaces, being the minimum and maximum coins you can attain.

CONSTRAINTS

For `1.in`, $n \leq 10$.

For `2.in`, `3.in` and `4.in`, $n \leq 10^4$.

For `5.in`, $n \leq 3 \times 10^5$.