Greetings From Globussoft

- Given below are 5 Programming questions, you have to solve any 3 out of 5 questions.

- These 5 questions you can attempt in any technology like C/C++, java, .Net, PHP

- To solve these 3 questions you've max. 3 hours.

- While Solving these questions you are not allowed to use any Search Engine like Google, Yahoo, Bing …

All the best for your test

Globussoft

QUESTION - 1

Gergovia consists of one street, and every inhabitant of the city is a wine salesman. Everyone buys wine from other inhabitants of the city. Every day each inhabitant decides how much wine he wants to buy or sell. Interestingly, demand and supply is always the same, so that each inhabitant gets what he wants.

There is one problem, however: Transporting wine from one house to another results in work. Since all wines are equally good, the inhabitants of Gergovia don't care which persons they are doing trade with, they are only interested in selling or buying a specific amount of wine.

In this problem you are asked to reconstruct the trading during one day in Gergovia. For simplicity we will assume that the houses are built along a straight line with equal distance between adjacent houses. Transporting one bottle of wine from one house to an adjacent house results in one unit of work.

Input

The input consists of several test cases.

Each test case starts with the number of inhabitants N ($2 \leq N \leq 100000$).

The following line contains n integers a_i ($-1000 \leq a_i \leq 1000$).

If $a_i \geq 0$, it means that the inhabitant living in the i^{th} house wants to buy a_i bottles of wine. If $a_i < 0$, he wants to sell $-a_i$ bottles of wine.

You may assume that the numbers a_i sum up to 0.

The last test case is followed by a line containing 0.

Output

For each test case print the minimum amount of work units needed so that every inhabitant has his demand fulfilled.

Example

Input

```
5
5 -4 1 -3 1
6
-1000 -1000 -1000 1000 1000 1000
0
```

Output:

```
9
9000
```

QUESTION – 2

Consider two integer sequences **f(n) = n!** and **g(n) = a^n**, where **n** is a positive integer. For any integer **a > 1** the second sequence is greater than the first for a finite number of values. But starting from some integer **k**, **f(n)** is greater than **g(n)** for all **n >= k**. You are to find the least positive value of **n** for which **f(n) > g(n)**, for a given positive integer **a > 1**.

Input

The first line of the input contains number **t** – the amount of tests. Then **t** test descriptions follow. Each test consist of a single number **a**.

Constraints

1 <= **t** <= 100000
2 <= **a** <= 10^6

Output

For each test print the least positive value of **n** for which **f(n) > g(n)**.

Example

Input:

```
3
2
3
4
```

Output:

```
4
7
9
```

QUESTION – 3

Given a string S, find the longest substring that appears at least twice in S (occurrences may overlap).

Input

The first line contains an integer L ($1 \leq L \leq 200000$), the length of S.

The second line contains the string S, consisting of exactly L lowercase letters ('a'-'z').

Output

Output the length of the longest substring that appears at least twice in S. If there is no such substring, output 0.

Example

Input:
```
11
sabcabcfabc
```

Output:
```
3
```

Input:
```
18
trutrutiktiktappop
```

Output:
```
4
```

QUESTION – 4

Bob is sitting at home with his computer. He would like to experience more social interaction, so he is planning a trip to a coffee shop with his computer.

Bob has lots of data about wireless networks and coffee shops in the city. In Bob's city, there is one coffee shop at every intersection of streets. Specifically, Bob happens to live in a city with M streets ($1 \leq M \leq 30000$) that run east and west, and N streets ($1 \leq N \leq 1000$) that run north and south. As an added benefit, the distance between consecutive parallel streets is 1 metre (it is a very compact city).

It also turns out that inside K ($1 \leq K \leq 1000$) of the coffee shops, there is a wireless network station. Each wireless network station will have a particular bitrate B ($1 \leq B \leq 1000$) and can reach R metres ($1 \leq R \leq 30000$) from the coffee shop. In other words, a wireless network station from one coffee shop forms a circle with radius R centered at that particular coffee shop. Moreover, if someone is at distance R, the wireless network would be available, but if someone is at a distance larger than R, they cannot access that wireless point.

You can assume that each coffee shop has at most one wireless network stationed in it, but that multiple wireless networks may be available while sitting in that one coffee shop, due to the proximity of other wireless network stations.

Bob has a special device in his computer that can use all of the available bitrates of as many wireless networks as he can connect to.

Bob would like to find out the maximum bitrate he can obtain, and how many coffee shops would have that maximum capacity.

Input

On the first line of input, you will be given the integer *M*, the number of east-west streets. On the second line of input, you will be given the integer *N*, the number of north-south streets. On the third line of input, you will be given the integer *K*, the number of coffee shops with a wireless network. On the next *K* lines, you will have 4 integers per line. The first integer *x* on each line represents the north-south street on which the coffee shop is located, where $1 \leq x \leq N$. The second integer, *y*, on each line represents the east-west street on which the coffee shop is located, where $1 \leq y \leq M$. The third integer, *R*, on each line represents the radius of the wireless network from this particular coffee shop. The fourth integer, *B*, on each line represents the bitrate of the wireless network from this particular coffee shop.

Output

The output will be two lines long. The first line of output will be the integer representing the maximum bitrate that can be obtained over all coffee shops. The second line of output will be the number of coffee shops where this maximum bitrate can be obtained.

Sample Input:

```
3
5
3
1 3 2 5
3 1 2 7
5 1 1 5
```

Sample Output:

```
12
5
```

QUESTION – 5

Consider the following exercise, found in a generic linear algebra textbook.Let A be an n × n matrix. Prove that the following statements are equivalent:

- (a) A is invertible.

- (b) $Ax = b$ has exactly one solution for every $n \times 1$ matrix b.
- (c) $Ax = b$ is consistent for every $n \times 1$ matrix b.
- (d) $Ax = 0$ has only the trivial solution $x = 0$.

The typical way to solve such an exercise is to show a series of implications. For instance,one can proceed by showing that (a) implies (b), that (b) implies (c), that (c) implies (d),and finally that (d) implies (a). These four implications show that the four statements are equivalent. Another way would be to show that (a) is equivalent to (b) (by proving that (a) implies (b) and that (b) implies (a)), that (b) is equivalent to (c), and that (c) is equivalent to (d).

However, this way requires proving six implications, which is clearly a lot more work than just proving four implications! I have been given some similar tasks, and have already started proving some implications. Now I wonder, how many more implications do I have to prove? Can you help me determine this ?

Input

On the first line one positive number: the number of testcases, at most 100. After that per testcase:

- One line containing two integers n ($1 \leq n \leq 20\,000$) and m ($0 \leq m \leq 50\,000$): the number of statements and the number of implications that have already been proved.
- m lines with two integers s1 and s2 ($1 \leq s1, s2 \leq n$ and $s1 = s2$) each, indicating that it has been proved that statement s1 implies statement s2.

Output

Per testcase:
One line with the minimum number of additional implications that need to be proved in order to prove that all statements are equivalent.

Example

Input:

```
2
4 0
3 2
1 2
1 3
```

Output:

```
4
2
```