ASSIGNMENT 1

Note:

1. *Assignment to be submitted on A4 size sheets ruled or blank. A description has to be* ***handwritten.*** *Printouts of the programs with* ***typical output*** *are required.*
2. *Put them in plastic file with covering sheet detailing your name, branch , semester, section, Roll no etc.*
3. *Assignment should be submitted to CR on time. CR has to arrange them roll number wise and submit them to my office as per the date mentioned.*
   It will be the responsibility of the students to submit the assignments well in time to the CR

1. Write a detailed description of Java's Just in time compilation.

Implement following programs

2. **Natural Language Processing***

   Natural Language Understanding is the subdomain of Natural Language Processing where people used to design AI based applications have ability to understand the human languages. HashInclude Speech Processing team has a project named Virtual Assistant. For this project they appointed you as a data engineer (who has good knowledge of creating clean datasets by writing efficient code). As a data engineer your first task is to make vowel recognition dataset. In this task you have to find the presence of vowels in all possible substrings of the given string. For each given string you have to print the total number of vowels.

   Input
   First line contains an integer T**,** denoting the number of test cases. (1 to 3)
   Each of the next lines contains a string, string contains both lower case and upper case (ignore case) .
   Output
   Print the vowel **sum**
   Answer for each test case should be printed in a new line.

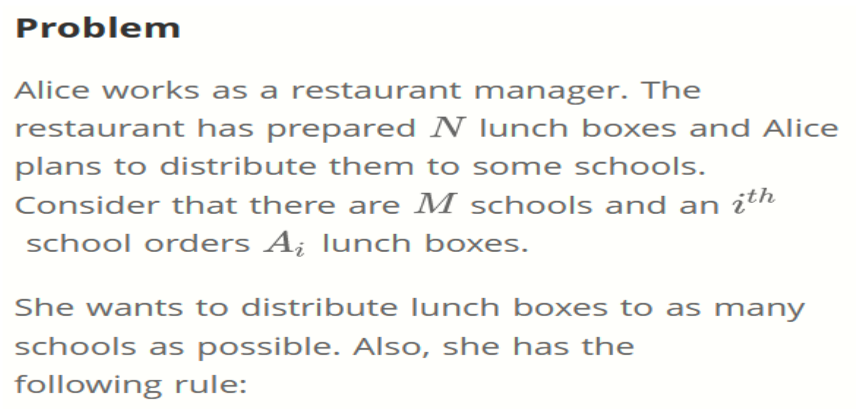
   Sample Input:
   1
   baceb

   Sample Output
   16

   Explanation

   First line is number of input string, In given example, string is "baceb" so the substrings will be like -"b, ba, bac, bace, a, ac, ace, aceb, c, ce, ceb, e, eb, baceb" now the number of vowels in each substring will be 0, 1, 1, 2, 1, 1, 2, 2, 0, 1, 1, 1, 1, 2 and the total number will be sum of all presence which is 16.

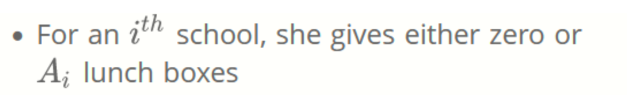
3. Lunch boxes *



Problem

Alice works as a restaurant manager. The restaurant has prepared N lunch boxes and Alice plans to distribute them to some schools. Consider that there are M schools and an i^{th} school orders A_i lunch boxes.

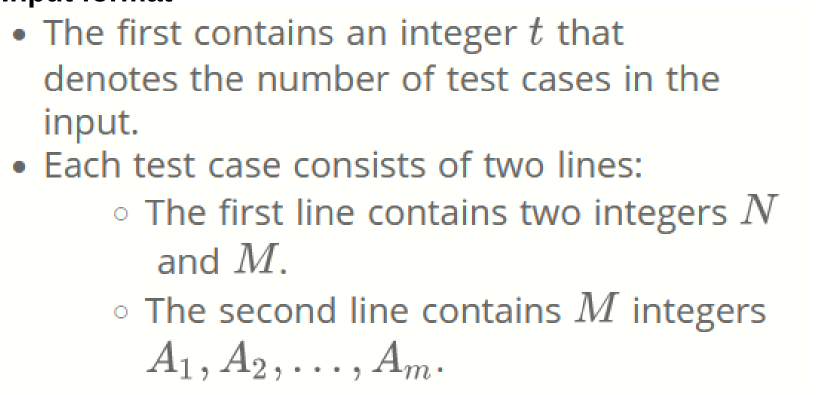
She wants to distribute lunch boxes to as many schools as possible. Also, she has the following rule:



- For an i^{th} school, she gives either zero or A_i lunch boxes

Your task is to help Alice to determine the maximum number of schools that can get lunch boxes.

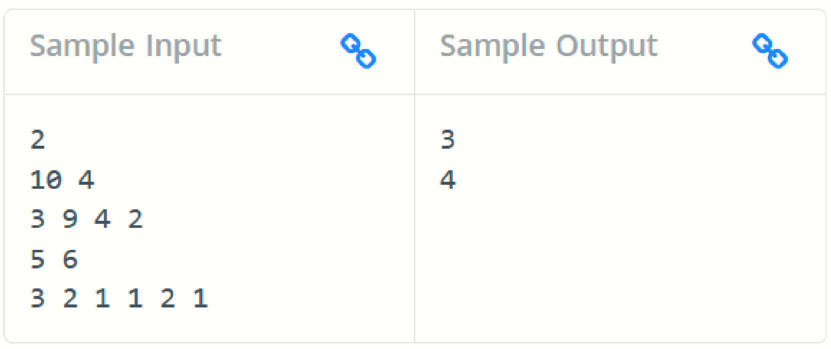
- **Input format**



  - The first contains an integer t that denotes the number of test cases in the input.
  - Each test case consists of two lines:
    - The first line contains two integers N and M.
    - The second line contains M integers $A_1, A_2, \ldots, A_m$.

- **Output format**

For each test case, you are required to print one integer that represents the maximum number of schools that can receive lunch boxes.



Sample Input	Sample Output

2
10 4
3 9 4 2
5 6
3 2 1 1 2 1
	3
	4

Explanation

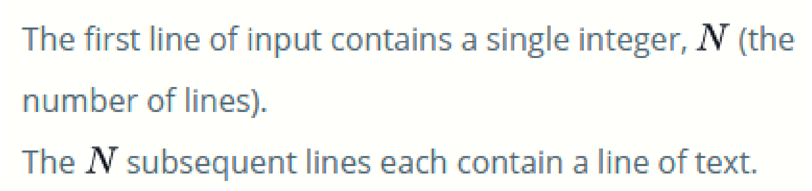
In first test case 1,3,4 schools got lunch-boxes.

In second test case 3,4 and 2(or 5) schools got lunch boxes.

4. **Tagged Language**:**
In a tag-based language like *XML* or *HTML*, contents are enclosed between a *start tag* and an *end tag* like `<tag>contents</tag>`. Note that the corresponding *end tag* starts with a `/`.
Given a string of text in a tag-based language, parse this text and retrieve the contents enclosed within sequences of well-organized tags meeting the following criterion:

   i. The name of the *start* and *end* tags must be same. The HTML code `<h1>Hello World</h2>` is *not valid*, because the text starts with an `h1` tag and ends with a non-matching `h2` tag.
   ii. Tags can be nested, but content between nested tags is considered *not valid*. For example, in `<h1><a>contents</a>invalid</h1>`, `contents` is *valid* but `invalid` is *not valid*.
   iii. Tags can consist of any printable characters.

- **Input Format**


The first line of input contains a single integer, N (the number of lines).
The N subsequent lines each contain a line of text.

- **Output Format**

For each line, print the content enclosed within valid tags. If a line contains multiple instances of valid content, print out each instance of valid content on a new line; if no valid content is found, print `None`.

- **Sample Input**

```
 4

<h1>Nayeem loves counseling</h1>

<h1><h1>Sanjay has no watch</h1></h1><par>So wait for a while</par>

<Amee>safat codes like a ninja</amee>

<SA premium>Imtiaz has a secret crush</SA premium>
```

- **Sample Output**

```
Nayeem loves counseling
Sanjay has no watch
So wait for a while
None
Imtiaz has a secret crush
```

**HackerRank

==