



## Question - 1

### Staircase Printing

Write a program that prints a staircase of size  $n$ .

#### Input Format

A single integer,  $n$ , denoting the size of the staircase.

#### Output Format

Print a staircase of size  $n$  using `#` symbols and spaces.

**Note:** The last line must have 0 spaces in it.

#### Sample Input

6

#### Sample output

```
#
##
###
####
#####
#####
```

## Question - 2

### Lucky 7

Lucky Seven :

Write a program which takes in an array of  $n$  integers and prints Lucky 7 if any three consecutive elements sum to 7 else prints Better Luck Next

Example

In a array  $\{2,1,5,1,0\}$   $\Rightarrow 1 + 5 + 1 = 7$  .Hence Prints -  
Lucky 7

`lucky_sevens?([3,4,3,4])`  $\Rightarrow$  Prints Better Luck Next

Sample Input:

```
5
2
1
5
1
10
```

the array - n.  
Followed by n elements of the array.

### Question - 3

#### Lucky 4

---

Preethi likes the number 4 much. Of course! This number has such a lot of properties, like:

- Four is the smallest composite number;
- Four is the maximal degree of the equation that can be solved in radicals;
- There is four-color theorem that states that any map can be colored in no more than four colors in such a way that no two adjacent regions are colored in the same color;
- In bases 6 and 12, 4 is a 1-automorphic number;
- And there are a lot more cool stuff about this number!

Impressed by the power of this number, Preethi has begun to look for occurrences of four anywhere. She has a number, for which we need to calculate the number of occurrences of the digit 4 in the decimal representation. help her please.

#### Input

#### Example

```
SampleInput: 447474  
Sample Output:4
```

```
Sample Input: 228  
Sample Output:0
```

### Question - 4

#### Merge 2 Arrays

---

Complete the *mergeArrays* function which has two parameters- array, *a* and array *b*. Both arrays contains *m* elements in non-decreasing order. The *mergeArrays* function should merge arrays *a* and *b* into a single, *non-decreasing* array of size *2m* and then returns the merged array.

#### Input Format

The first line of the input is an integer *m*, total number of elements in arrays *a* and *b*. Each of the next subsequent *m* lines contains a single integer, denoting the elements of array *a*. Then the next line of input is the integer *m*. Each of the next *m* lines contains a single integer, denoting the elements of array *b*.

#### Constraints

- $1 < m < 5 \times 10^5$
- $0 \leq a_i, b_i \leq 10^9$ , where  $0 \leq i < m$

#### Output Format

Your function should return the merged array.

#### Sample Input 1

```
4
1
5
7
7
4
0
1
2
3
```

#### Sample Output 1

```
0
1
1
2
3
5
7
7
```

#### Sample Input 2

```
5
2
4
5
9
9
5
0
1
2
3
4
```

#### Sample Output 2

```
0
1
2
2
3
4
4
5
9
9
```

#### Explanation

##### Sample Case 1

The following arrays are passed to *mergeArrays* as arguments:

$a = \{ 1, 5, 7, 7 \}$

$b = \{ 0, 1, 2, 3 \}$

The *mergedArray* function returns the following merged, non-decreasing array:  $\{ 0, 1, 1, 2, 3, 5, 7, 7 \}$

##### Sample Case 2

The following arrays are passed to *mergeArrays* as arguments:

$a = \{ 2, 4, 5, 9, 9 \}$

$b = \{ 0, 1, 2, 3, 4 \}$

The *mergedArray* function returns the following merged, non-decreasing array:  $\{ 0, 1, 2, 2, 3, 4, 4, 5, 9, 9 \}$

## Question - 5

### Frequency of a word in a sentence

---

Write a code to count the Frequency of the word "the" in a sentence and print the frequency

## Question - 6

### Shape Classes

---

Implement the following classes and methods:

- A *Circle* class with:
  - A constructor having one floating-point number parameter, *radius*.
  - A *getArea()* method that returns a ceiling-rounded integer denoting the area of the *Circle* object, calculated using the formula:  
 $area(Circle(radius)) = 3.14159265 \times radius \times radius$ .
- A *Rectangle* class with:
  - A constructor having two floating-point number parameters: *width* and *height*.
  - A *getArea()* method that returns a ceiling-rounded integer denoting the area of the *Rectangle* object, calculated using the formula:  
 $area(Rectangle(width, height)) = width \times height$ .
- A *Square* class with:
  - A constructor having one floating-point number parameter, *width*.
  - A *getArea()* method that returns a ceiling-rounded integer denoting the area of the *Square* object, calculated using the formula:  
 $area(Square(width)) = width \times width$ .

### Input Format

Locked stub code in the editor reads the following input from stdin and passes it to your constructors and methods:

The first line contains a single floating-point number denoting the *radius* of a circle.

The second line contains two space-separated floating-point numbers denoting the respective *width* and *height* of a rectangle.

The third line contains a single floating-point number denoting the *radius* of another circle.

The fourth line contains a single floating-point number denoting the *width* of a square.

The fifth line contains two space-separated floating-point numbers denoting the respective *width* and *height* of another rectangle.

### Output Format

The *getArea* method must return a ceiling-rounded integer denoting the area of the shape object it's called on.

Locked stub code in the editor tests your code by calling your constructors and methods.

### Sample Input 0

```
5
3 4
2
3.3
5 7.5
```

### Sample Output 0

```
79
12
13
11
38
```

### Explanation 0

Area of *Circle*(5) is  $3.14159265 \times 5.0 \times 5.0 =$

$78.53981625 \Rightarrow 79$

Area of *Rectangle*(3, 4) is  $3.0 \times 4.0 = 12.0 \Rightarrow 12$

Area of *Circle*(2) is  $3.14159265 \times 2.0 \times 2.0 = 12.5663706$   
 $\Rightarrow 13$

Area of *Square*(3.3) is  $3.3 \times 3.3 = 10.89 \Rightarrow 11$

Area of *Rectangle*(5, 7.5) is  $5.0 \times 7.5 = 37.5 \Rightarrow 38$

## Question - 7

### How Will You Compare?

Write a *Comparator* class with the following 3 overloaded *compare* methods:

1. *boolean compare(int a, int b)*: Return **true** if *int a = int b*, otherwise return **false**.
2. *boolean compare(string a, string b)*: Return **true** if *string a = string b*, otherwise return **false**.
3. *boolean compare(int[] a, int[] b)*: Return **true** if both of the following conditions hold *true*:
  - Arrays *a* and *b* are of equal length.
  - For each index *i* (where  $0 \leq i < |a|, |b|$ ),  $a[i] = b[i]$ .Otherwise, return **false**.

**Note:** For C++, both parameters are of type *Vector<int>*.

### Input Format

You *do not* need to read anything from stdin. The following input from stdin is handled by the locked stub code in your editor:

The first line contains an integer, *T*, the number of test cases. Each test case is described as follows:

- If the first line contains the integer *1*, the next 2 lines contain *strings a* and *b*. The stub code then passes them to *compare(string, string)* and prints the result.
- If the first line contains the integer *2*, the next 2 lines contain *integers a* and *b*. The stub code then passes them to *compare(int, int)* and prints the result.
- If the first line contains the integer *3*, the next 3 lines contain the following:
  1. Two space-separated integers describing the

respective lengths of arrays  $a$  and  $b$ .

2. A line of  $|a|$  space-separated integers describing the elements in array  $a$ .
3. A line of  $|b|$  space-separated integers describing the elements in array  $b$ .

The stub code then assembles the *two* array arguments and passes them to `compare(int[], int[])`, then prints the result.

### Output Format

You *do not* need to print anything to stdout. This is handled for you by the locked stub code in your editor.

### Sample Input 1

```
3
1
hello world
hello world
2
3
4
3
3 3
1 2 3
1 2 3
```

### Sample Output 1

```
Same
Different
Same
```

### Sample Input 2

```
2
3
3 4
1 2 3
1 2 3 4
1
HackerRank
hackerRank
```

### Sample Output 2

```
Different
Different
```

### Explanation

*Sample Case 1:*

There are 3 test cases:

T e s t C a s e	co n di ti on	a	b	O ut p ut	Explanation
1	1	"hello"	"hello"	Same	Both strings are the same.

		<i>world</i>	<i>world</i>	<i>me</i>	
2	2	3	4	"Different"	The two integers are different ( $3 \neq 4$ ).
3	3	{1, 2, 3}	{1, 2, 3}	"Same"	Both arrays have the same number of elements and each element $a[i] = b[i]$

Sample Case 2:

There are 2 test cases.

T e s t C a s e	co n di ti on	a	b	O ut p ut	Explanation
1	3	{1, 2, 3}	{1, 2, 3, 4}	"Different"	Both arrays are different.
2	1	HackerRank	hackerRank	"Different"	The two strings are different.