Recursion 1

Calculate Power
Send Feedback
Write a program to find x to the power n (i.e. x^n). Take x and n from the user. You need to return the answer.
Do this recursively.
Input format :

Two integers x and n (separated by space)

Output Format :

x^n (i.e. x raise to the power n)

Constraints :

0 <= x <= 30
0 <= n <= 30

Sample Input 1 :

3 4

Sample Output 1 :

81

Sample Input 2 :

2 5

Sample Output 2 :

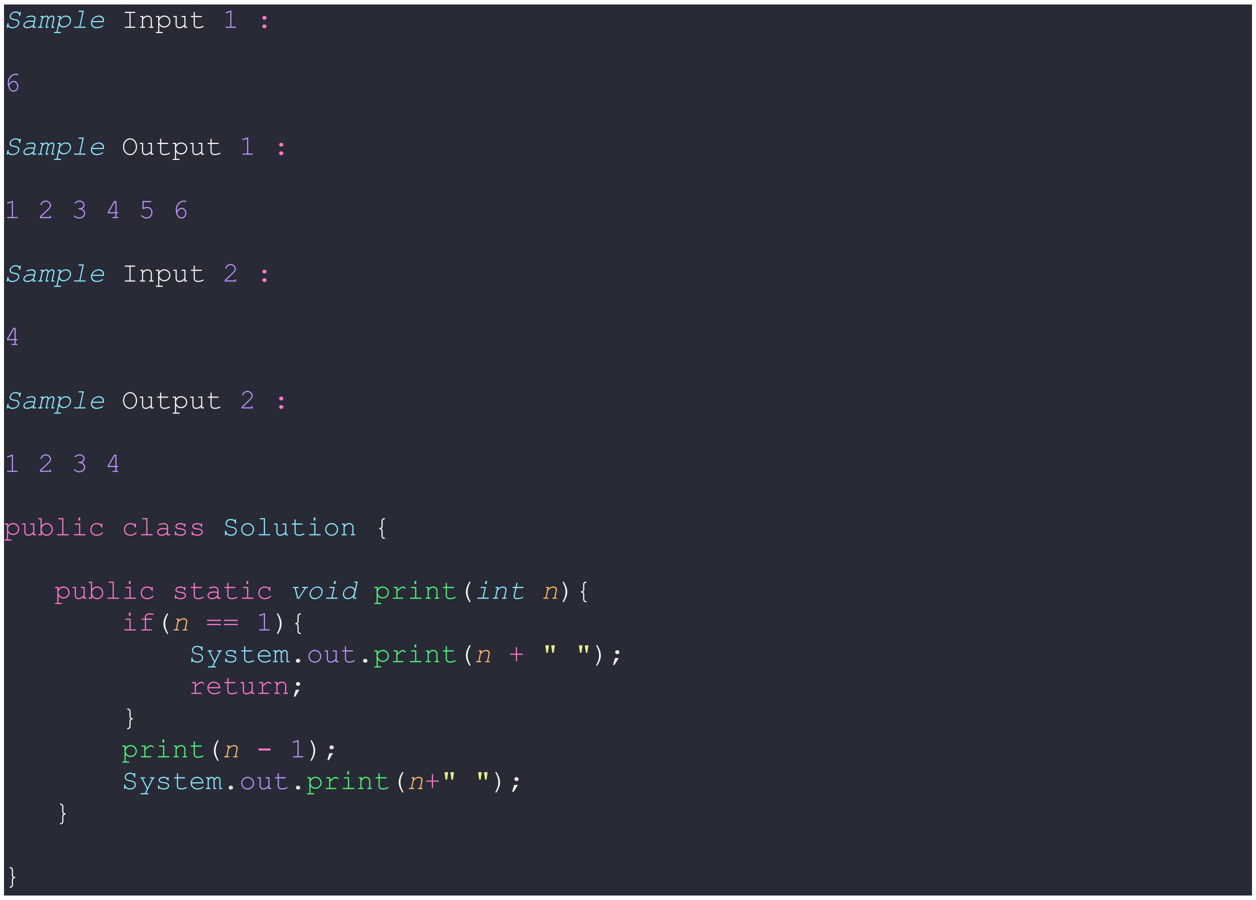
32

```
public class Solution {

   public static int power(int x, int n) {
        if(n==0)
            return 1;
        int smalloutput = power(x, n-1);
        int output = x * smalloutput;
        return output;
   }
}
```

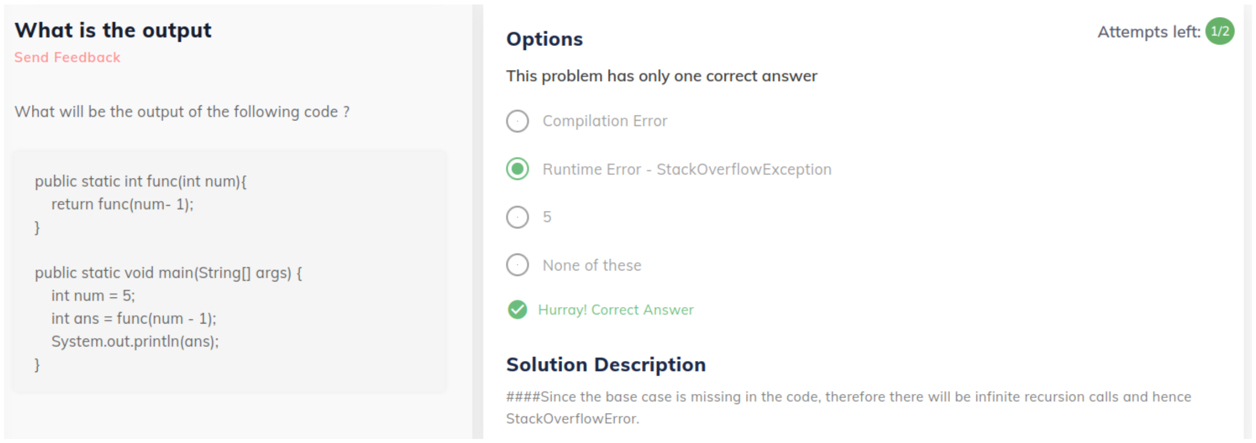
Number of Digits
Send Feedback
Given the number 'n', find out and return the number of digits present in a number recursively.
Input Format :

```
Integer n

Output Format :

Count of digits

Constraints :

1 <= n <= 10^6

Sample Input 1 :

156

Sample Output 1 :

3

Sample Input 2 :

7

Sample Output 2 :

1

public class Solution {

   public static int count(int n){
       if(n == 0){
           return 0;
       }
       int smallAns = count(n / 10);
       return smallAns + 1;
   }

}
```

```
Print First N Natural Numbers - Code
Send Feedback
Given the number 'n', write a code to print numbers from 1 to n in
increasing order recursively.
Input Format :

Integer n

Output Format :

Numbers from 1 to n (separated by space)

Constraints :

1 <= n <= 10000
```



```
Sample Input 1 :

6

Sample Output 1 :

1 2 3 4 5 6

Sample Input 2 :

4

Sample Output 2 :

1 2 3 4

public class Solution {

   public static void print(int n){
       if(n == 1){
           System.out.print(n + " ");
           return;
       }
       print(n - 1);
       System.out.print(n+" ");
   }

}
```



What is the output

Send Feedback

What will be the output of the following code ?

```
public static int func(int num){
   return func(num- 1);
}

public static void main(String[] args) {
   int num = 5;
   int ans = func(num - 1);
   System.out.println(ans);
}
```

Options

Attempts left: 1/2

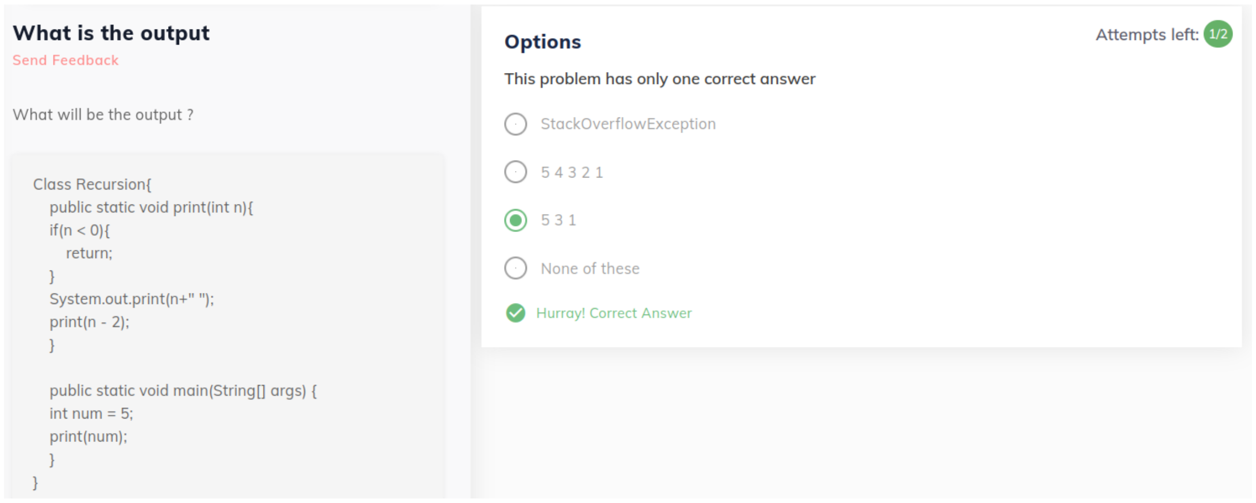
This problem has only one correct answer

- Compilation Error
- [✓ Runtime Error - StackOverflowException]
- 5
- None of these

✓ Hurray! Correct Answer

Solution Description

####Since the base case is missing in the code, therefore there will be infinite recursion calls and hence StackOverflowError.



What is the output

Send Feedback

What will be the output ?

```
Class Recursion{
   public static void print(int n){
   if(n < 0){
      return;
   }
   System.out.print(n+" ");
   print(n - 2);
   }

   public static void main(String[] args) {
   int num = 5;
   print(num);
   }
}
```

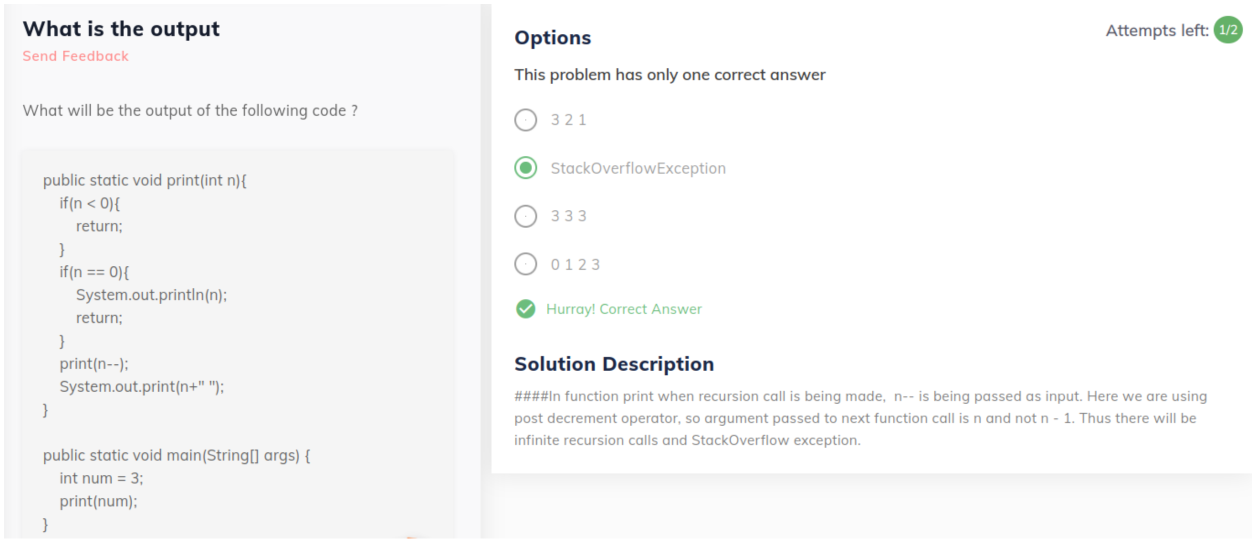
Options

Attempts left: 1/2

This problem has only one correct answer

- StackOverflowException
- 5 4 3 2 1
- [✓ 5 3 1]
- None of these

Hurray! Correct Answer



What is the output

Send Feedback

What will be the output of the following code ?

```
public static void print(int n){
   if(n < 0){
      return;
   }
   if(n == 0){
      System.out.println(n);
      return;
   }
   print(n--);
   System.out.print(n+" ");
}

public static void main(String[] args) {
   int num = 3;
   print(num);
}
```

Options

Attempts left: 1/2

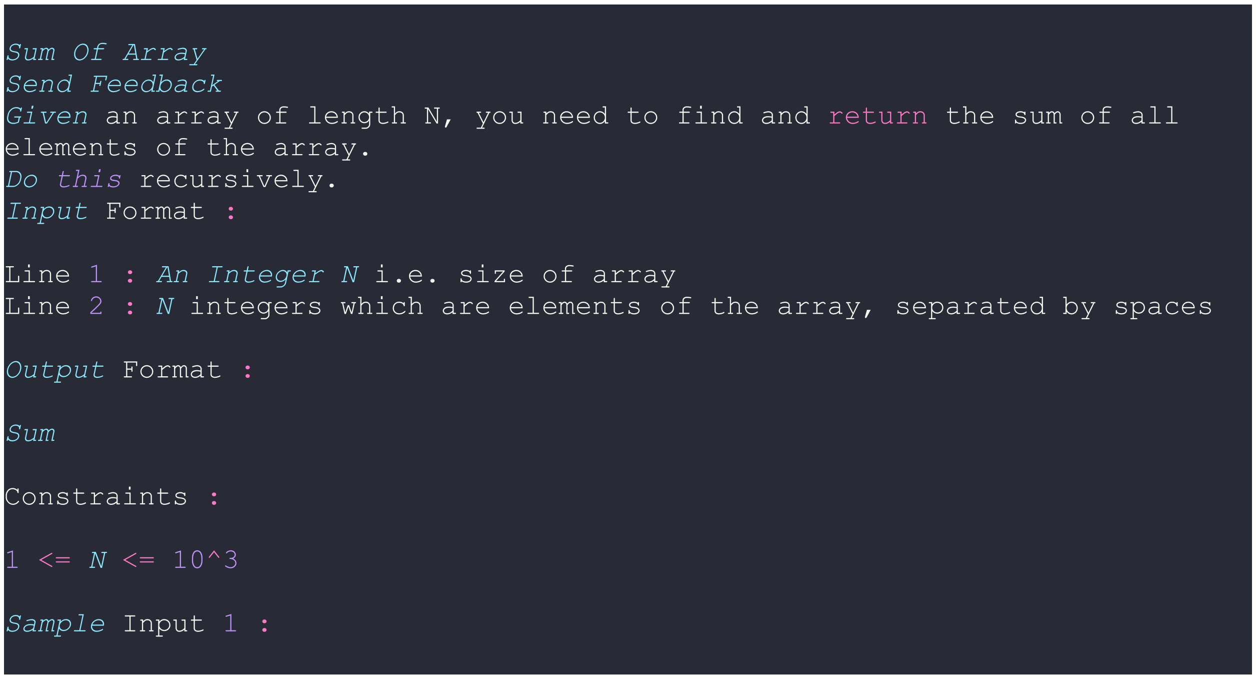
This problem has only one correct answer

- 3 2 1
- [✓ StackOverflowException]
- 3 3 3
- 0 1 2 3

Hurray! Correct Answer

Solution Description

####In function print when recursion call is being made, n-- is being passed as input. Here we are using post decrement operator, so argument passed to next function call is n and not n - 1. Thus there will be infinite recursion calls and StackOverflow exception.



```
Sum Of Array
Send Feedback
Given an array of length N, you need to find and return the sum of all
elements of the array.
Do this recursively.
Input Format :

Line 1 : An Integer N i.e. size of array
Line 2 : N integers which are elements of the array, separated by spaces

Output Format :

Sum

Constraints :

1 <= N <= 10^3

Sample Input 1 :
```

```
3
9 8 9

Sample Output 1 :

26

Sample Input 2 :

3
4 2 1

Sample Output 2 :

7

public class Solution {

   public static int sum(int input[]) {

       if(input.length == 1)
           return input[0];
       int arr[] = new int[input.length-1];

       for(int i=1; i<input.length; i++){
           arr[i-1] = input[i];
       }
       return input[0]+sum(arr);
   }
}
```

```
Check Number in Array
Send Feedback
Given an array of length N and an integer x, you need to find if x is
present in the array or not. Return true or false.
Do this recursively.
Input Format :

Line 1 : An Integer N i.e. size of array
Line 2 : N integers which are elements of the array, separated by spaces
Line 3 : Integer x

Output Format :

'true' or 'false'

Constraints :

1 <= N <= 10^3

Sample Input 1 :

3
```

```
9 8 10
8

Sample Output 1 :

true

Sample Input 2 :

3
9 8 10
2

Sample Output 2 :

false

public class Solution {

   public static boolean checkNumber(int input[], int x) {
       if(input[0]==x)
           return true;
       if(input.length==1)
           return false;
       int arr[] = new int[input.length-1];
       for(int i=1; i<input.length; i++){
           arr[i-1] = input[i];
       }
       return checkNumber(arr,x);
   }
}
```

```
First Index Of a Number in an Array - Question
Send Feedback
Given an array of length N and an integer x, you need to find and return
the first index of integer x present in the array. Return -1 if it is not
present in the array.
First index means, the index of first occurrence of x in the input array.
Do this recursively. Indexing in the array starts from 0.
Input Format :

Line 1 : An Integer N i.e. size of array
Line 2 : N integers which are elements of the array, separated by spaces
Line 3 : Integer x

Output Format :

first index or -1

Constraints :
1 <= N <= 10^3
Sample Input :
```

```
4
9 8 10 8
8

Sample Output :

1


public class Solution {
   static int startindex =0;
   public static int firstIndex(int input[],  int x) {
       if(startindex==input.length)
           return -1;
       if(x==input[startindex])
           return startindex;
       startindex++;
       // if(startindex==input.length-1)
       //     return -1;

       return firstIndex(input, x);


   }

}
```