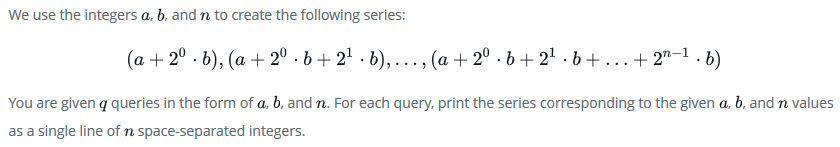
# Java Loops II(Easy)



Case link: https://www.hackerrank.com/challenges/java-loops-i/problem

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class Solution {

private static final Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {

        int N = scanner.nextInt();

        scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

        for(int i=1; i<=10; i++) {

            System.out.println(N+" x "+i+" = "+N\*i);

        }

        scanner.close();

    }

}

# Java Output Formatting(Easy)

Java's System.out.printf function can be used to print formatted output. The purpose of this exercise is to test your understanding of formatting output using printf.

To get you started, a portion of the solution is provided for you in the editor; you must format and print the input to complete the solution.

Case link: <https://www.hackerrank.com/challenges/java-output-formatting/problem>

import java.util.Scanner;

public class Solution {

    public static void main(String[] args) {

            Scanner sc=new Scanner(System.in);

            System.out.println("================================");

            for(int i=0;i<3;i++){

                String s1=sc.next();

                int x=sc.nextInt();

                //Complete this line

                System.out.printf("%-15s%03d\n",s1,x);

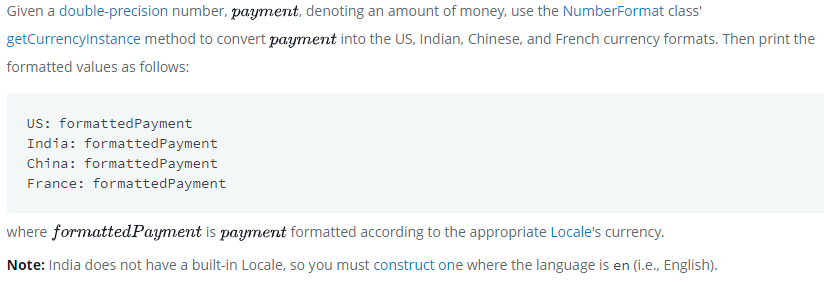
            }

            System.out.println("================================");

    }

}

# Java Currency Formatter(Easy)



Case link: <https://www.hackerrank.com/challenges/java-currency-formatter/problem>

import java.util.\*;

import java.text.\*;

public class Solution {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        double payment = scanner.nextDouble();

        scanner.close();

        // Write your code here.

        String us,india,china,france;

        us = NumberFormat.getCurrencyInstance(Locale.US).format(payment);

        india = NumberFormat.getCurrencyInstance(new Locale("en","in")).format(payment);

        china = NumberFormat.getCurrencyInstance(Locale.CHINA).format(payment);

        france = NumberFormat.getCurrencyInstance(Locale.FRANCE).format(payment);

        System.out.println("US: "+us);

        System.out.println("India: "+india);

        System.out.println("China: "+china);

        System.out.println("France: "+france);

    }

}

# Java Date and Time(Easy)

# 

# Case link: <https://www.hackerrank.com/challenges/java-date-and-time/problem>

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.function.\*;

import java.util.regex.\*;

import java.util.stream.\*;

import static java.util.stream.Collectors.joining;

import static java.util.stream.Collectors.toList;

import java.time.LocalDate;

class Result {

    /\*

     \* Complete the 'findDay' function below.

     \*

     \* The function is expected to return a STRING.

     \* The function accepts following parameters:

     \*  1. INTEGER month

     \*  2. INTEGER day

     \*  3. INTEGER year

     \*/

    public static String findDay(int month, int day, int year) {

Calendar rightNow = Calendar.getInstance();

        rightNow.set(Integer.valueOf(year), (Integer.valueOf(month) - 1), Integer.valueOf(day));

        return rightNow.getDisplayName(Calendar.DAY\_OF\_WEEK, Calendar.LONG, Locale.getDefault()).toUpperCase() ;

    }

}

public class Solution {

    public static void main(String[] args) throws IOException {

        BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));

        BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

        String[] firstMultipleInput = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

        int month = Integer.parseInt(firstMultipleInput[0]);

        int day = Integer.parseInt(firstMultipleInput[1]);

        int year = Integer.parseInt(firstMultipleInput[2]);

        String res = Result.findDay(month, day, year);

        bufferedWriter.write(res);

        bufferedWriter.newLine();

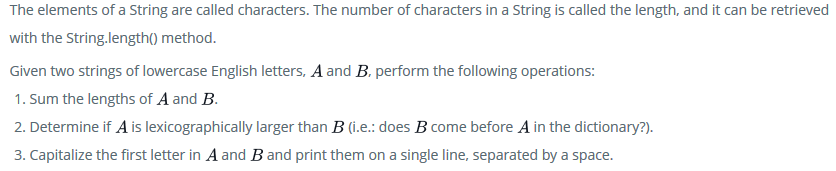
        bufferedReader.close();

        bufferedWriter.close();

    }

}

# Java Strings Introduction(Easy)



Case link: <https://www.hackerrank.com/challenges/java-strings-introduction/problem>

import java.io.\*;

import java.util.\*;

public class Solution {

    public static void main(String[] args) {

        Scanner sc=new Scanner(System.in);

        String A=sc.next();

        String B=sc.next();

        /\* Enter your code here. Print output to STDOUT. \*/

        System.out.println(A.length()+B.length());

        int com = A.compareTo(B);

        if(com>0){

            System.out.println("Yes");

        }else{

            System.out.println("No");

        }

        String outA = A.substring(0, 1).toUpperCase()+A.substring(1);

        String outB = B.substring(0, 1).toUpperCase()+B.substring(1);

        System.out.println(outA+" "+outB);

    }

}

# Java Substring(Easy)

# 

Case link: <https://www.hackerrank.com/challenges/java-substring/problem?h_r=next-challenge&h_v=zen>

import java.io.\*;

import java.util.\*;

import java.text.\*;

import java.math.\*;

import java.util.regex.\*;

public class Solution {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

        String S = in.next();

        int start = in.nextInt();

        int end = in.nextInt();

        System.out.println(S.subSequence(start, end));

    }

}

# Java Substring Comparisons(Easy)

# 

# Case link: <https://www.hackerrank.com/challenges/java-string-compare/problem?h_r=next-challenge&h_v=zen&h_r=next-challenge&h_v=zen>

import java.util.Scanner;

public class Solution {

    public static String getSmallestAndLargest(String s, int k) {

        // Complete the function

        // 'smallest' must be the lexicographically smallest substring of length 'k'

        // 'largest' must be the lexicographically largest substring of length 'k'

        String subWord = s.substring(0, k);

        String smallest = subWord;

        String largest = subWord;

        for(int i=1; i<=s.length()-k; i++){

            subWord = s.substring(i, i+k);

           if(subWord.compareTo(largest) > 0){

               largest=subWord;

           }if(subWord.compareTo(smallest) < 0){

               smallest=subWord;

           }

        }

        return smallest + "\n" + largest;

    }

    public static void main(String[] args) {

        Scanner scan = new Scanner(System.in);

        String s = scan.next();

        int k = scan.nextInt();

        scan.close();

        System.out.println(getSmallestAndLargest(s, k));

    }

}

# Tag Content Extractor(Medium)

# 

# Case link: <https://www.hackerrank.com/challenges/tag-content-extractor/problem>

import java.io.\*;

import java.util.\*;

import java.text.\*;

import java.math.\*;

import java.util.regex.\*;

public class Solution{

    public static void main(String[] args){

        Scanner in = new Scanner(System.in);

        int testCases = Integer.parseInt(in.nextLine());

        while(testCases>0){

            String line = in.nextLine();

            boolean same = false;

            Pattern regex = Pattern.compile("<(.+)>([^<]+)</\\1>");

            Matcher match = regex.matcher(line);

            while(match.find()){

                System.out.println(match.group(2));

                same = true;

            }

            if(!same){

                System.out.println("None");

            }

            //Write your code here

            testCases--;

        }in.close();

    }

}

# Java 1D Array(Easy)

# 

# Case link: <https://www.hackerrank.com/challenges/java-1d-array-introduction/problem>

import java.util.\*;

public class Solution {

    public static void main(String[] args) {

        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();

        int[] a = new int[n];

        for(int i = 0; i < n; i++)

        {

            a[i] = scan.nextInt();

        }

        scan.close();

        // Prints each sequential element in array a

        for (int i = 0; i < a.length; i++) {

            System.out.println(a[i]);

        }

    }

}