**Hacker Rank Certification Answers**

**Question and Answers**

**-----------------------shape -----------------------**

class Shape

{

  int length, breadth;

    Shape(int l,int b)

    {

        length=l;

        breadth=b;

    }

    public void area()

    {

        System.out.println(length\*breadth);

    }

}

class Rectangle extends Shape

{

  public Rectangle(int l,int b)

    {

        super(l,b);

    }

    @Override

    public void area()

    {

        System.out.println(length\*breadth);

    }

}

public class Solution {

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

        /\* Enter your code here. Read input from STDIN. Print output to STDOUT \*/

        Scanner sc=new Scanner(System.in);

        int l=sc.nextInt();

        int b=sc.nextInt();

        System.out.println(l+" "+b);

        Rectangle obj=new Rectangle(l,b);

        obj.area();

    }

}

2.

**-----Type counter---**

class Result {

    /\*

     \* Complete the 'typeCounter' function below.

     \*

     \* The function accepts STRING sentence as parameter.

     \*/

  public static void typeCounter(String sentence) {

           int dbl = 0;

        int str = 0;

        int integer =0;

        String[] arr = String.valueOf(sentence).split("\\s");

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

            try {

                   Integer.parseInt(arr[i]);

                    integer++;

                }catch (Exception e){

                try{

                    Double.parseDouble(arr[i]);

                    dbl++;

                }catch (Exception err){

                    str++;

                }

            }

//            if(arr[i].matches("[0-9]+")){

//                ++integer;

//             }

//           else if(arr[i].matches("[a-zA-Z]+")){

//                ++str;

//            }

//            else{

//                ++dbl;

//            }

        }

        System.out.println("string " + str);

        System.out.println("integer " + integer);

        System.out.println("double " + dbl);

  }

}

public class Solution {

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

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

        String sentence = bufferedReader.readLine();

        Result.typeCounter(sentence);

        bufferedReader.close();

    }

}

3. 15

4. compiliation error due to line no 6

5. 9.0

6. The overriding method can have derived type as the return type instead of the base type.

**-----------------------IS BALANCED--------------------**

  static String isBalanced(String s)

    {

        Stack<Character> openBracket = new Stack<Character>();

        // 'true' to start, if s == ""

        // use to break from loop, instead of having multi return statements

        boolean isBalanced = true;

        // "YES" to start, if s == ""

        String isBalancedStr = "true";

        int i = 0;

        while (isBalanced && i < s.length()){

            if (s.charAt(i) == '{' || s.charAt(i) == '(' || s.charAt(i) == '['){

                openBracket.push(s.charAt(i));

            } else if (s.charAt(i) == '}') {

                if (openBracket.isEmpty() || openBracket.pop() != '{'){

                    isBalanced = false;

                    isBalancedStr = "false";

                }

            } else if (s.charAt(i) == ')') {

                if (openBracket.isEmpty() || openBracket.pop() != '('){

                    isBalanced = false;

                    isBalancedStr = "false";

                }

            } else if (s.charAt(i) == ']') {

                if (openBracket.isEmpty() || openBracket.pop() != '['){

                    isBalanced = false;

                    isBalancedStr = "false";

                }

            }

            i++;

        }

        // if open bracket is left, 's' is not balanced

        // ex. ([{}])(

        if (!openBracket.isEmpty()){

            isBalancedStr = "false";

        }

        return isBalancedStr;

    }

**-------------------------------------Encrypt and Decrypt------------------------------------------**

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;

class Result {

    /\*

     \* Complete the 'decryptMessage' function below.

     \*

     \* The function is expected to return a STRING.

     \* The function accepts STRING encryptedMessage as parameter.

     \*/

   public static String decryptMessage(String encryptedMessage) {

        // Split the message into words

        String[] words = encryptedMessage.split(" ");

        // Reverse the order of words

        StringBuilder reversedSentence = new StringBuilder();

        for (int i = words.length - 1; i >= 0; i--) {

            reversedSentence.append(words[i]);

            if (i > 0) {

                reversedSentence.append(" ");

            }

        }

        // Split the reversed sentence into words again

        String[] reversedWords = reversedSentence.toString().split(" ");

        // Decompress each word

        StringBuilder decryptedMessage = new StringBuilder();

        for (String word : reversedWords) {

            decryptedMessage.append(decompressWord(word)).append(" ");

        }

        // Remove the trailing space and return the result

        return decryptedMessage.toString().trim();

    }

    private static String decompressWord(String word) {

        StringBuilder decompressedWord = new StringBuilder();

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

            char currentChar = word.charAt(i);

            // Check if the next character is a digit (frequency)

            if (i + 1 < word.length() && Character.isDigit(word.charAt(i + 1))) {

                int frequency = Character.getNumericValue(word.charAt(i + 1));

                for (int j = 0; j < frequency; j++) {

                    decompressedWord.append(currentChar);

                }

                // Skip the frequency digit

                i++;

            } else {

                decompressedWord.append(currentChar);

            }

        }

        return decompressedWord.toString();

    }

}

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 encryptedMessage = bufferedReader.readLine();

        String result = Result.decryptMessage(encryptedMessage);

        bufferedWriter.write(result);

        bufferedWriter.newLine();

        bufferedReader.close();

        bufferedWriter.close();

    }

}

**---------------------------------------------Adder Class---------------------------------**

**class Adder extends Calculator{**

**public int add(int a, int b){**

**return a + b;**

**}**

**}**

**----How will you compare ---------------------**

import java.util.\*;

/\*Write your code here. DO NOT use access modifiers (e.g.: 'public') in your class declarations.\*/

class Solution {

    public static void main(String []args)

    {

    Scanner scan = new Scanner(System.in);

        Comparator comp = new Comparator();

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

        while(testCases-- > 0){

            int condition = Integer.parseInt(scan.nextLine());

            switch(condition){

                case 1:

                    String s1 = scan.nextLine().trim();

                    String s2 = scan.nextLine().trim();

                    System.out.println(comp.compare(s1, s2) ? "Same" : "Different");

                    break;

                case 2:

                    int num1 = scan.nextInt();

                    int num2 = scan.nextInt();

                    System.out.println(comp.compare(num1, num2) ? "Same" : "Different");

                    if(scan.hasNext()){ // avoid exception if this is the last test case

                        scan.nextLine(); // consume the remaining newline

                    }

                    break;

                case 3:

                    // create and fill arrays

                    int[] array1 = new int[scan.nextInt()];

                    int[] array2 = new int[scan.nextInt()];

                    for(int i = 0; i < array1.length; i++){

                        array1[i] = scan.nextInt();

                    }

                    for(int i = 0; i < array2.length; i++){

                        array2[i] = scan.nextInt();

                    }

                    System.out.println(comp.compare(array1, array2) ? "Same" : "Different");

                    if(scan.hasNext()){ // avoid exception if this is the last test case

                        scan.nextLine(); // consume the remaining newline

                    }

                    break;

                default:

                    System.err.println("Invalid input.");

            } // end switch

        } // end while

        scan.close();

    }

}

class Comparator {

    // Method to compare two strings

    public boolean compare(String s1, String s2) {

        return s1.equals(s2);

    }

    // Method to compare two integers

    public boolean compare(int num1, int num2) {

        return num1 == num2;

    }

    // Method to compare two integer arrays

    public boolean compare(int[] array1, int[] array2) {

        if(array1.length != array2.length) {

            return false;

        }

        for(int i = 0; i < array1.length; i++) {

            if(array1[i] != array2[i]) {

                return false;

            }

        }

        return true;

    }

}

**--------java Multi sum --**public class Solution { public int sum(Integer[] numbers) { int sum = 0; for (int num : numbers) { sum += num; } return sum; } public String sum(String[] strings) { String result = ""; for (String str : strings) { result += str; } return result; } public static void main(String args[] ) throws Exception { Scanner sc = new Scanner(System.in); String input = sc.nextLine(); Solution solution = new Solution(); // Check if input contains any digits if (input.matches(".\*\\d.\*")) { // Assume input is a list of integers String[] parts = input.split("\\s+"); Integer[] numbers = new Integer[parts.length]; for (int i = 0; i < parts.length; i++) { numbers[i] = Integer.parseInt(parts[i]); } int result = solution.sum(numbers); System.out.println(result); } else { // Assume input is a list of strings String[] parts = input.split("\\s+"); String result = solution.sum(parts); System.out.println(result); } sc.close(); } }

**=adder class—**

 class Adder extends Calculator

 {

     int add (int a, int b)

     {

         return (a+b);

     }

 }