LOVELY PROFESSIONAL UNIVERSITY

School of Computer Application

II-Assignment On Java (CAP-680) Session 2022-2024

Submitted to:
Name: Dr. Md Irfan Alam
(Dept of. Computer Science)

Submitted by:
Name: Sanjeev Kumar
Roll no: RDOC14 A26
Course: MCA

Department of Computer Science
Lovely Professional University Jalandhar Punjab (144401)
India

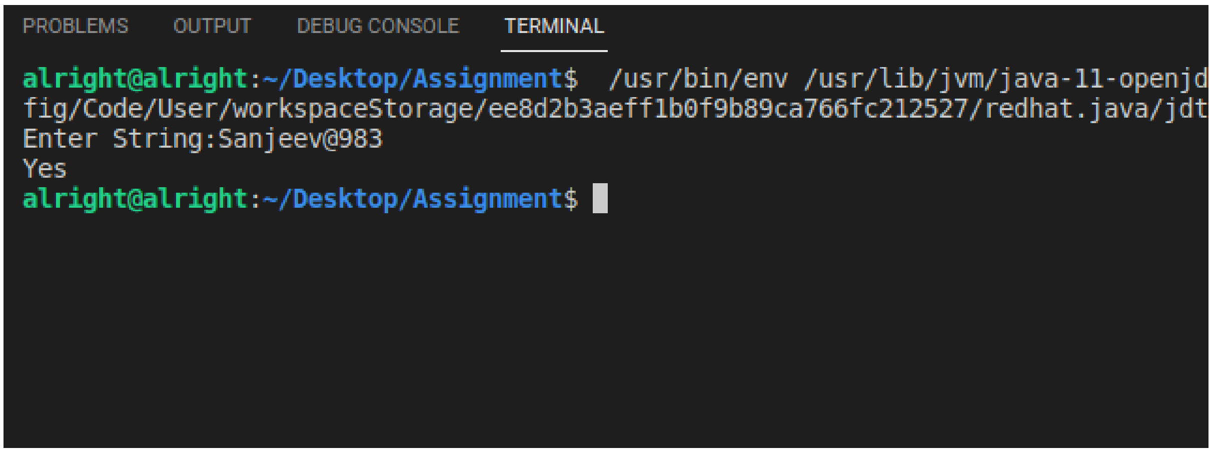
Smith is planning to setup a secure password for his customer account.For
a password to be secure the following conditions should be satisfied:

- Password must contain at least one lower case letter [a-z];
- Password must contain at least one upper case letter [A-Z] strictly inside, i.e. not as the first or the last character;
- Password must contain at least one digit [0-9] strictly inside;
- Password must contain at least one special character from the set {'@', '#', '%', '&', '?' } strictly inside:
- Password must be at least 10 characters in length, but it can be longer.

Input
First line will contain T, number of test cases. Then the test cases follow.
Each test case contains of a single line of input, string S.
Output
For each test case, output in a single line "YES" if the password is secure and "NO" if it is not.

```
import java.util.Scanner;

import java.util.regex.*;



class qu1 {



   public static void

   isAllPresent(String str)

   {



       String regex = "^(?=.*[a-z])(?=."

                  + "*[A-Z])(?=.*\\d)"

                  + "(?=.*[-+_!@#$%^&*., ?]).+$";





       Pattern p = Pattern.compile(regex);



       if (str == null) {
        System.out.println("No");

        return;

    }





            Matcher m = p.matcher(str);




    if (m.matches() && str.length()>9)

        System.out.println("Yes");

    else

        System.out.println("No");

}



public static void main(String args[])

{

    Scanner scan = new Scanner (System.in);

    System.out.print("Enter String:");

    String str = scan.nextLine();

    isAllPresent(str);

    scan.close();

}
```

Output:



```
alright@alright:~/Desktop/Assignment$  /usr/bin/env /usr/lib/jvm/java-11-openjd
fig/Code/User/workspaceStorage/ee8d2b3aeff1b0f9b89ca766fc212527/redhat.java/jdt
Enter String:Sanjeev@983
Yes
alright@alright:~/Desktop/Assignment$ 
```

Q2)Given an expression string x. Examine whether the pairs and the orders of {,},(,),[,] are correct in exp.

For example, the function should return 'true' for exp = [()]{}{[()()]()} and 'false' for exp = [(]).

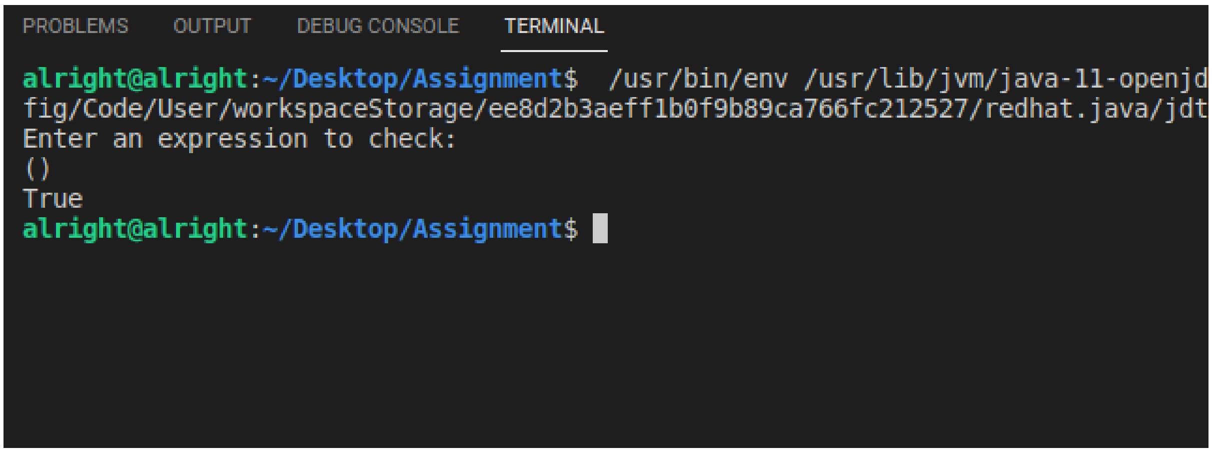
Note: The drive code prints "balanced" if the function returns true, otherwise it prints "not balanced".

Input:
()
Output:
true

```
import java.util.*;

public class qu_2 {
   public static void main(String[] args) {
       try (Scanner scanner = new Scanner(System.in)) {
           System.out.println("Enter an expression to check:");
           String exp = scanner.nextLine();

           if (isBalanced(exp)) {
               System.out.println("True");
           } else {
               System.out.println("False");
           }
       }
   }

   public static boolean isBalanced(String exp) {
```

```java
        Stack<Character> stack = new Stack<>();

        for (int i = 0; i < exp.length(); i++) {
            char ch = exp.charAt(i);

            if (ch == '(' || ch == '{' || ch == '[') {
                stack.push(ch);
            } else if (ch == ')' || ch == '}' || ch == ']') {
                if (stack.isEmpty()) {
                    return false;
                }

                char top = stack.pop();

                if ((ch == ')' && top != '(') || (ch == '}' && top != '{')
|| (ch == ']' && top != '[')) {
                    return false;
                }
            }
        }

        return stack.isEmpty();
    }
}
```

Output:

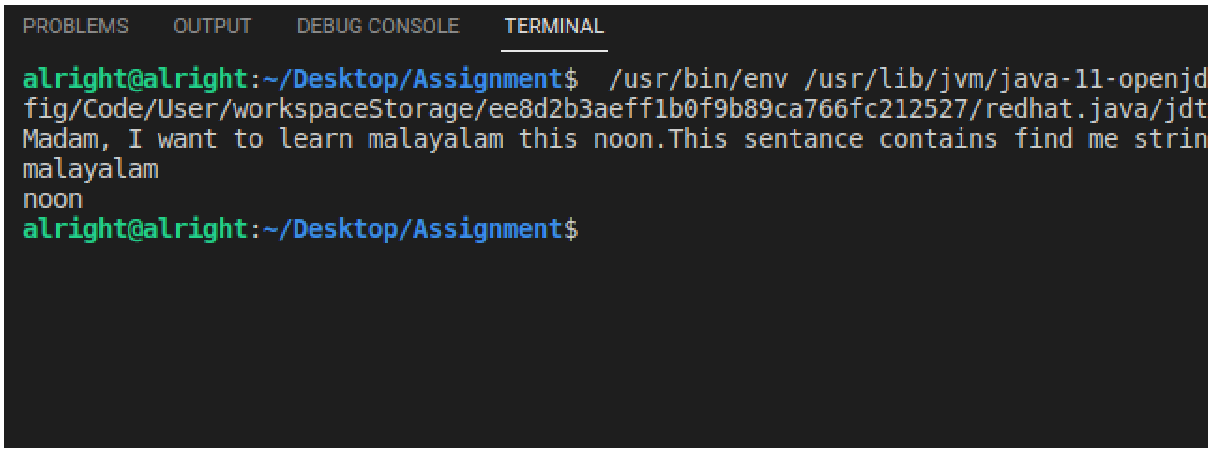


```
alright@alright:~/Desktop/Assignment$  /usr/bin/env /usr/lib/jvm/java-11-openjd
fig/Code/User/workspaceStorage/ee8d2b3aeff1b0f9b89ca766fc212527/redhat.java/jdt
Enter an expression to check:
()
True
alright@alright:~/Desktop/Assignment$ 
```

Q3) A text mining system accepts a sentence as an input. It tries to extract those words which read the same backwards or forwards. This system is interested in extracting only the largest and smallest possible words. Develop a java program which can help this text mining system for the extraction of such words from an input sentence.

```java
public class qu_3 {

   public static void main(String[] args){

       String str = "Madam, I want to learn malayalam this noon.This sentence
contains find me string";

       System.out.println(str);

       String find = "malayalam";

       String find2 = "noon";

       int i = str.indexOf(find);

       int i2 = str.indexOf(find2);

       if(i>0)

           System.out.println(str.substring(i, i+find.length()));

       else

           System.out.println("string not found");

           if(i2>0)

           System.out.println(str.substring(i2, i2+find2.length()));

       else

           System.out.println("string not found");

   }

}
```

Output:



PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

```
alright@alright:~/Desktop/Assignment$  /usr/bin/env /usr/lib/jvm/java-11-openjd
fig/Code/User/workspaceStorage/ee8d2b3aeff1b0f9b89ca766fc212527/redhat.java/jdt
Madam, I want to learn malayalam this noon.This sentance contains find me strin
malayalam
noon
alright@alright:~/Desktop/Assignment$
```

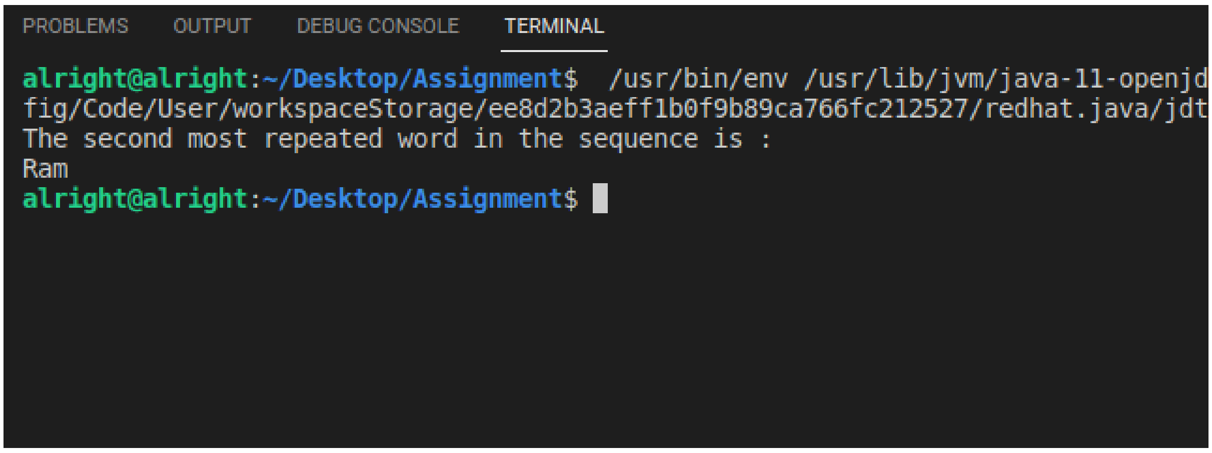
Q4): Given all the students, Mr. XYZ wants to find that student name which is second most frequent in this section. Develop a java program which can help Mr. XYZ to locate such student name with its frequency.

Example : Input – [“Ram”, “Aryan”, “Sumit”, “Ram”, “Sumit”, “Akshay”, “Moni”, “Sumit”]

```java
import java.util.*;

public class qu_4 {
   static String second_repeated(Vector<String> my_seq) {
       HashMap<String, Integer> my_map = new HashMap<String,
Integer>(my_seq.size()) {
           @Override
           public Integer get(Object key) {
               return containsKey(key) ? super.get(key) : 0;
           }
       };
       for (int i = 0; i < my_seq.size(); i++)
           my_map.put(my_seq.get(i), my_map.get(my_seq.get(i)) + 1);
       int first_val = Integer.MIN_VALUE;
       int sec_val = Integer.MIN_VALUE;
       Iterator<Map.Entry<String, Integer>> my_iter =
my_map.entrySet().iterator();
       while (my_iter.hasNext()) {
           Map.Entry<String, Integer> ent = my_iter.next();
           int v = ent.getValue();
           if (v > first_val) {
               sec_val = first_val;
               first_val = v;
           } else if (v > sec_val && v != first_val)
               sec_val = v;
       }
       my_iter = my_map.entrySet().iterator();
       while (my_iter.hasNext()) {
           Map.Entry<String, Integer> ent = my_iter.next();
           int v = ent.getValue();
           if (v == sec_val)
               return ent.getKey();
       }
       return null;
   }

   public static void main(String[] args) {
       String arr[] = { "Ram", "Aryan", "Sumit", "Ram", "Sumit", "akshay",
"Moni", "Sumit" };
       List<String> my_seq = Arrays.asList(arr);
```

```
        System.out.println("The second most repeated word in the sequence is
: ");
        System.out.println(second_repeated(new Vector<>(my_seq)));
    }
}
```

Output:

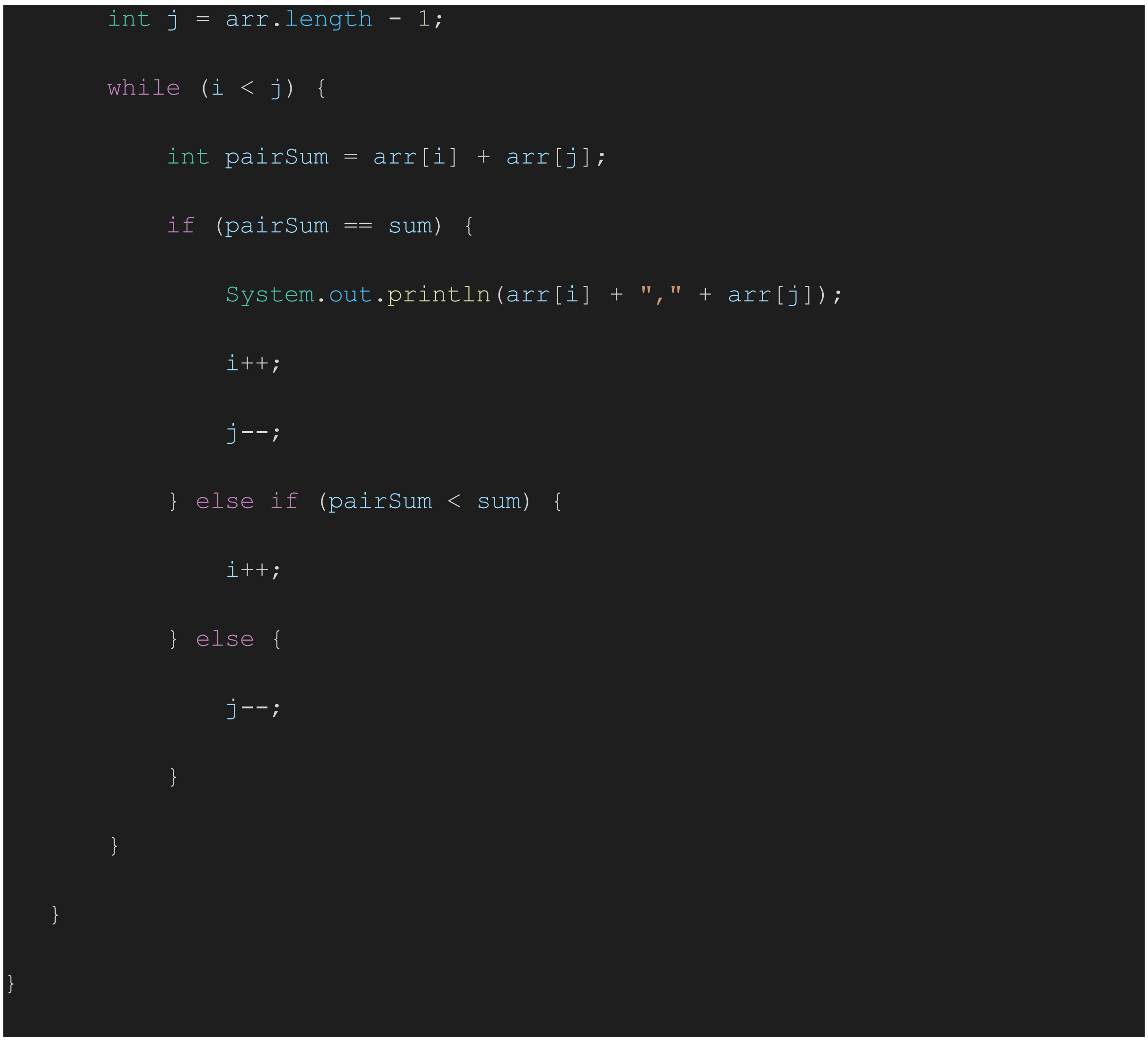


```
alright@alright:~/Desktop/Assignment$  /usr/bin/env /usr/lib/jvm/java-11-openjd
fig/Code/User/workspaceStorage/ee8d2b3aeff1b0f9b89ca766fc212527/redhat.java/jdt
The second most repeated word in the sequence is :
Ram
alright@alright:~/Desktop/Assignment$ 
```

Q-5) A company XYZ is storing its employee's salary in an array of size N. This company is interested to find out those pairs of employees whose sum of salary is equal to a given number K. Given this array, develop a java program which can help the company to extract such pairs.

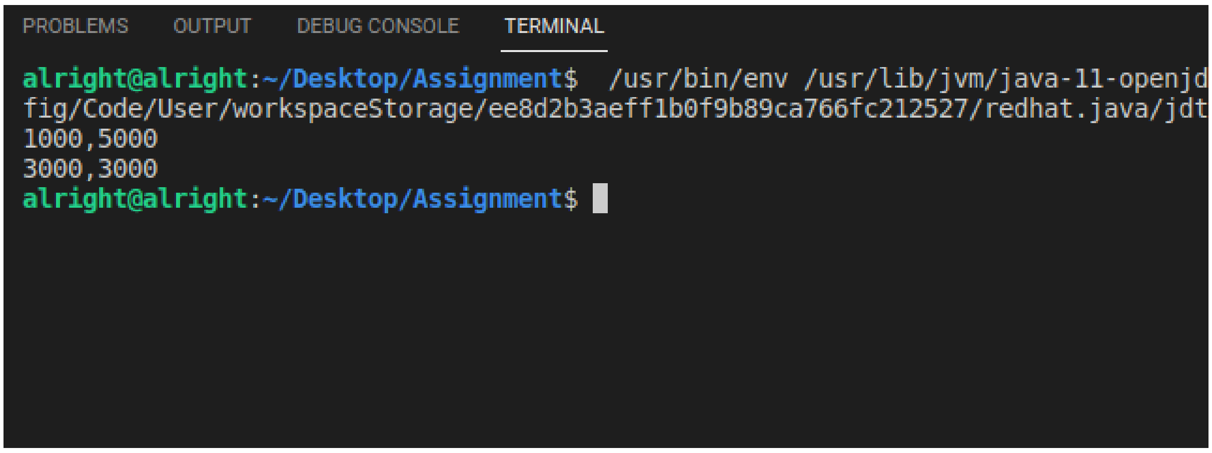
Example: Input – N = 5, K = 6000, Salary [] = {1000, 5000, 1000, 7000, 6000}

```
import java.util.Arrays;


public class qu_5 {


    public static void main(String[] args) {

        int[] arr = { 1000, 5000, 3000, 3000, 6000 };

        int sum = 6000;

        Arrays.sort(arr);

        // Find pairs

        int i = 0;
```



```java
        int j = arr.length - 1;

        while (i < j) {

            int pairSum = arr[i] + arr[j];

            if (pairSum == sum) {

                System.out.println(arr[i] + "," + arr[j]);

                i++;

                j--;

            } else if (pairSum < sum) {

                i++;

            } else {

                j--;

            }

        }

    }

}
```

Output:



```
alright@alright:~/Desktop/Assignment$  /usr/bin/env /usr/lib/jvm/java-11-openjd
fig/Code/User/workspaceStorage/ee8d2b3aeff1b0f9b89ca766fc212527/redhat.java/jdt
1000,5000
3000,3000
alright@alright:~/Desktop/Assignment$ 
```