Cyclesheet- 1 Java Lab

Name: Sankalp Mukim

Registration Number: 20BDS0128

Subject: Java (Lab)

Slot: L21+L22

Link to all the code: https://github.com/sankalpmukim/jafva-lab-fall-sem-2021

Table of Contents

Question 1:

Write a Java program to read the number of students, ‘n’ as input from the user. For the ‘n’ students, the user may input either his registration number (an integer), name (String) or CGPA (float), randomly. The names may have multiple parts. Read those inputs and display the following.

Count of the registration numbers, count of the CGPA and count of the names

1. Average of all the cgpa values entered
2. The least and the greatest registration number entered so far
3. a single string that concatenates all the names with a comma between them.
4. Sample Input:

11 (n value)

6.8
14
25
8.3
7.9
Peter George
9.2
Dwarakesh
17
6
Ram Desai

Code:

```java
import java.util.*;

public class Q1 {
    private static boolean onlyDigits(String str) {
        for (int i = 0; i < str.length(); i++) {
            if (Character.isDigit(str.charAt(i)) == false) {
                return false;
            }
        }
        return true;
    }
```

```java
public static void main(String[] args) {
    int n;
    Scanner sc = new Scanner(System.in);
    n = Integer.parseInt(sc.nextLine());
    int numRegnos = 0, numCgpas = 0, numNames = 0;
    int[] regnos = new int[n];
    float[] cgpas = new float[n];
    String[] names = new String[n];
    String currenString;
    for (int i = 0; i < n; i++) {
        currenString = sc.nextLine();
        if (currenString.contains(".")) {
            cgpas[numCgpas++] = Float.parseFloat(currenString);
            continue;
        }
        if (Q1.onlyDigits(currenString)) {
            regnos[numRegnos++] = Integer.parseInt(currenString);
            continue;
        }
        names[numNames++] = currenString;
    }
    System.out.print(numRegnos);
    System.out.println(" (Count of reg.no)");
    System.out.print(numCgpas);
    System.out.println(" (Count of CGPA)");
    System.out.print(numNames);
    System.out.println(" (Count of names)");
    float avg = 0;
    for (int i = 0; i < numCgpas; i++) {
        avg += cgpas[i];
    }
    avg /= numCgpas;
    System.out.print(avg);
    System.out.println(" (Average of CGPA)");
    int leastRegNo = 100000, greatestRegNo = -1;

    for (int i = 0; i < numRegnos; i++) {
        if (regnos[i] < leastRegNo) {
            leastRegNo = regnos[i];
        }
        if (regnos[i] > greatestRegNo) {
            greatestRegNo = regnos[i];
        }
    }
    System.out.print(leastRegNo);
    System.out.println(" (Least reg.no)");
    System.out.print(greatestRegNo);
    System.out.println(" (Greatest reg.no)");
```

```
        for (int i = 0; i < numNames; i++) {
            if (i != numNames - 1) {
                System.out.print(names[i] + ", ");
            } else {
                System.out.println(names[i]);
            }
        }
        sc.close();
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> & 'c:\Users\
sanka\.vscode\extensions\vscjava.vscode-java-debug-0.36.0\scripts\launcher.bat' 'C:\Program Files\Ecli
pse Foundation\jdk-11.0.12.7-hotspot\bin\java.exe' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\sanka\AppDa
ta\Roaming\Code\User\workspaceStorage\9bf9376e625620dffc29874d8f3e6e4f\redhat.java\jdt_ws\CyclesheetOn
e_5bc1aaa2\bin' 'Q1'
11
6.8
14
25
8.3
7.9
Peter George
9.2
Dwarakesh
17
6
Ram Desai
4 (Count of reg.no)
4 (Count of CGPA)
3 (Count of names)
8.05 (Average of CGPA)
6 (Least reg.no)
25 (Greatest reg.no)
Peter George, Dwarakesh, Ram Desai
```

Question 2:

The details of a list of transactions carried out on a single day for a particular bank are given as command line inputs in the order of customer name, total amount, transacted amount. Read these inputs. If the transacted amount is negative, it can be considered as withdrawal. If it is positive, it can be considered as deposit. For withdrawal, either if the total amount is less than the |transacted amount| or if the withdrawal amount exceeds the maximum limit of 25000, display the message, “Failed Transaction”. Withdrawal charges are as follows.

If the withdrawal amount <=500, charge=5

If the withdrawal amount <=1000, charge=8

If the withdrawal amount >1000 and <=5000, charge=10

If the withdrawal amount >5000 and <=15000, charge =12

If the withdrawal amount >15000 and <=25000, charge=15

For every customer, print the name and his balance amount after the transaction.

Sample Input:

java MainClass Vinay 7000 1500 Andrea 46000 -28000 Venba 18500 -11800 Mithil 78000 3000 Kavin 8600 -10000

Output:

Vinay

8500

Andrea

Failed Transaction

46000

Venba

6688

Mithil

81000

Kavin

Failed Transaction

8600

Code:

```
public class Q2 {
    public static int withdrawalCharge(int amount) {
        if (amount <= 500) {
            return 5;
        }
        if (amount <= 1000) {
            return 8;
        }
        if (amount <= 5000) {
            return 10;
        }
        if (amount <= 15000) {
            return 12;
        }
        if (amount <= 25000) {
            return 15;
        }
        return 0;
    }

    public static void main(String[] args) {
        for (int i = 0; i < args.length; i += 3) {
            String name = args[i];
```

```
            int amount = Integer.parseInt(args[i + 1]);
            int transaction = Integer.parseInt(args[i + 2]);
            System.out.println(name);
            if (transaction < 0) {
                // Withdrawal
                transaction = Math.abs(transaction);
                if (amount < transaction || transaction > 25000) {
                    System.out.println("Failed Transaction");
                    System.out.println(amount);
                    continue;
                }
                int withdrawalCharge;
                if (transaction <= 5000) {
                    withdrawalCharge = 5;
                } else if (transaction <= 1000) {
                    withdrawalCharge = 8;
                } else if (transaction <= 5000) {
                    withdrawalCharge = 10;
                } else if (transaction <= 15000) {
                    withdrawalCharge = 12;
                } else {
                    withdrawalCharge = 15;
                }
                System.out.println((amount - transaction - withdrawalCharge));
            } else {
                System.out.println(amount + transaction);
            }
        }
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> javac .\Q2.ja
va
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> java Q2 Vinay
 7000 1500 Andrea 46000 -28000 Venba 18500 -11800  Mithil 78000 3000 Kavin 8600 -10000
Vinay
8500
Andrea
Failed Transaction
46000
Venba
6688
Mithil
81000
Kavin
Failed Transaction
8600
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne>
```

Question 3:

Part 1:

Write a Java program to print the sum of the series

1-22+333-4444+ … upto n terms (without using string functions)

Sample Input I:
4 (number of terms)
Sample Output I:
-4132

Sample Input II:
3 (number of terms)
Sample Output II:
312

Code:

```java
package Q3;

import java.util.Scanner;

public class Q31 {
    private static int numGenerator(int n) {
        int x = 0;
        for (int i = 0; i < n; i++) {
            x += n * Math.pow(10, i);
        }
        return x;
    }

    public static void main(String[] args) {
        int sum = 0;
        int n;
        Scanner sc = new Scanner(System.in);
        n = sc.nextInt();
        for (int i = 0; i < n; i++) {
            if (i % 2 == 0) {
                sum += numGenerator(i + 1);
            } else {
                sum -= numGenerator(i + 1);
            }
        }
        System.out.println(sum);
        sc.close();
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q3> javac Q31.java
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q3> java Q31
4
-4132
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q3> java Q31
3
312
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q3>
```

Part 2:

Given the number of rows, n, as input, write a program to print the following pattern.

Example:

4 (number of rows for the pattern)

```
   1
  1  2
 1    3
1      4
 1    3
  1  2
   1
```

Sample Input II:

3

```
   1
  1  2
 1    3
  1  2
   1
```

Code:

```java
package Q3;

import java.util.Scanner;

public class Q32 {
    private static String stringMultiplier(String x, int n) {
        String initString = "";
        for (int i = 0; i < n; i++) {
            initString += x;
        }
        return initString;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.close();
        // First Half
        for (int i = 1; i <= n; i++) {
```

```
            System.out.print(stringMultiplier(" ", (n - i)) + "1");
            if (i > 1) {
                System.out.println(stringMultiplier(" ", ((i - 1) * 2) - 1) +
i);
            } else {
                System.out.println("");
            }
        }
        // Second Half
        for (int i = n - 1; i >= 1; i--) {
            System.out.print(stringMultiplier(" ", (n - i)) + "1");
            if (i > 1) {
                System.out.println(stringMultiplier(" ", ((i - 1) * 2) - 1) +
i);
            } else {
                System.out.println("");
            }
        }
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q3> javac .\Q32.java
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q3> java Q32
4
   1
  1 2
 1   3
1     4
 1   3
  1 2
   1
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q3> java Q32
3
  1
 1 2
1   3
 1 2
  1
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q3>
```

Question 4:

Given a list of ‘n’ integers, build a 2-D ragged array with n-2 rows. Store the following in each row.
Row 0: sum of the individual 2-element subarrays formed by taking 2 contiguous elements each time
Row 1: sum of the individual 3-element subarrays formed by taking 3 contiguous elements each time and so on
…….
Row n-3: sum of the individual n-element subarrays formed by taking all the ‘n’ elements of the given list. Print the ragged array.
For each element x, in the 2-D array, perform OR and Ex-OR operations with every other element in the same row and print the pair of elements along with their results. If for any pair of values, the OR and Ex-OR results are found to be the same, stop further calculations for that row and move to the next row.
Sample Input: [2,-1,4,3,-1,0,5]

Sample Output:
(1,3) OR=3 XOR=2
(1,7) OR=7 XOR=6
(1,2) OR=3 XOR=3
Row 0 is abruptly terminated
(5,6) OR=7 XOR=3
(5,6) OR=7 XOR=3
(5,2) OR=7 XOR=7
Row 1 is abruptly terminated
(8,5) OR=13 XOR=13
Row 2 is abruptly terminated
(7,5) OR=7 XOR=2
(7,11) OR=15 XOR=12
(5,11) OR=15 XOR=14
Row 3 is processed entirely
(7,10) OR=15 XOR=13
Row 4 is processed entirely

Code:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class Q4 {
    private static int summer(int[] arr, int start, int end) {
        int sum = 0;
        for (int i = start; i <= end; i++) {
            sum += arr[i];
        }
        return sum;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < arr.length; i++) {
            arr[i] = sc.nextInt();
        }
        List<int[]> jaggedArray = new ArrayList<int[]>();
        for (int i = 2; i < arr.length; i++) {
            int[] sumArray = new int[arr.length - i + 1];
            for (int j = 0; j < sumArray.length; j++) {
                sumArray[j] = summer(arr, j, j + i - 1);
            }
            jaggedArray.add(sumArray.clone());
        }
        for (int i = 0; i < jaggedArray.size(); i++) {
            boolean terminate = false;
```

```
            boolean manuallyTerminated = false;
            manuallyTerminated = false;
            for (int j = 0; j < jaggedArray.get(i).length; j++) {
                for (int k = 0; k < jaggedArray.get(i).length; k++) {
                    if (j <= k) {
                        int or = jaggedArray.get(i)[j] | jaggedArray.get(i)[k]
;
                        int xor = jaggedArray.get(i)[j] ^ jaggedArray.get(i)[k
];
                        if (jaggedArray.get(i)[j] == jaggedArray.get(i)[k]) {
                            continue;
                        }
                        if (or != xor) {
                            System.out.println("(" + jaggedArray.get(i)[j] + "
," + jaggedArray.get(i)[k] + ")" + " OR="
                                    + or + " XOR=" + xor);
                        } else {
                            System.out.println("(" + jaggedArray.get(i)[j] + "
," + jaggedArray.get(i)[k] + ")" + " OR="
                                    + or + " XOR=" + xor);
                            System.out.println("Row " + i + " abruptly termina
ted");
                            terminate = true;
                            break;
                        }
                    }
                }
                if (terminate) {
                    terminate = false;
                    manuallyTerminated = true;
                    break;
                }
            }

            if (!manuallyTerminated) {
                System.out.println("Row " + (i) + " is processed entirely");
            }
        }
        sc.close();
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> javac .\Q4.java
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> java Q4
7
2 -1 4 3 -1 0 5
(1,3) OR=3 XOR=2
(1,7) OR=7 XOR=6
(1,2) OR=3 XOR=3
Row 0 abruptly terminated
(5,6) OR=7 XOR=3
(5,6) OR=7 XOR=3
(5,2) OR=7 XOR=7
Row 1 abruptly terminated
(8,5) OR=13 XOR=13
Row 2 abruptly terminated
(7,5) OR=7 XOR=2
(7,11) OR=15 XOR=12
(5,11) OR=15 XOR=14
Row 3 is processed entirely
(7,10) OR=15 XOR=13
Row 4 is processed entirely
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne>
```

Question 5:

Write a program to read the size and the elements of a square matrix. Rotate the i^{th} row elements 'i' times towards the left and then rotate the j^{th} column elements 'j' times upwards. Print the resulting matrix.

Example:

4

Input matix:

A B C D
E F G H
I J K L
M N O P

Output:

A G I O
F L N D
K M C E
P B H J

Sample Input 2:
3
X Y Z
A B C
P Q R
Sample Output 2:
X C Q
B P Z

R Y A

Code:

```java
import java.util.Scanner;

public class Q5 {
    private static void rotateLeft(String[][] arr, int i) {
        int k = i;
        while (k > 0) {
            String leftMost = arr[i][0];
            for (int j = 0; j < arr[i].length - 1; j++) {
                arr[i][j] = arr[i][j + 1];
            }
            arr[i][arr[i].length - 1] = leftMost;
            k--;
        }
    }

    private static void rotateUpwards(String[][] arr, int j) {
        int k = j;
        while (k > 0) {
            String leftMost = arr[0][j];
            for (int m = 0; m < arr[j].length - 1; m++) {
                arr[m][j] = arr[m + 1][j];
            }
            arr[arr[j].length - 1][j] = leftMost;
            k--;
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        String[][] arr = new String[n][n];
        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j < arr[i].length; j++) {
                arr[i][j] = sc.next();
            }
        }
        sc.close();

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

        }

        for (int i = 0; i < arr.length; i++) {
            rotateUpwards(arr, i);
        }
```

```
        System.out.println("Output:");

        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j < arr[i].length; j++) {
                System.out.print(arr[i][j] + " ");
            }
            System.out.println("");
        }
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> javac .\Q5.java
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> java Q5
4
A B C D
E F G H
I J K L
M N O P
Output:
A G I O
F L N D
K M C E
P B H J
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> java Q5
3
X Y Z
A B C
P Q R
Output:
X C Q
B P Z
R Y A
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne>
```

Question 6:

// This question is marked Q5 in the cyclesheet

Write a Java program to read a list of ‘n’ words. Mark each letter in the word as a vowel or consonant. Replace a sequence of consecutive consonants by a single ‘C’ and a sequence of consecutive vowels by a single ‘V’. For each word, print the resulting “CV” sequence and the count of the number of occurrences of the pattern “VC”.

Eg., if the word is “CARROT”

C	A	R	R	O	T
↓	↓	↓	↓	↓	↓
C	V	C	C	V	C

=CVCVC Therefore, count of the pattern “VC” is 2

Sample Input:

5 (the value of n)

CLASS

GLUE

COMPATIBILITY

ASSESSMENT

PROGRAM

Sample Output:

CVC
1
CV
0
CVCVCVCVCVC
5
VCVCVC
3
CVCVC
2

Code:

```
import java.util.Scanner;

public class Q6 {
    private static boolean isVowel(char ch) {
        if (ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {
            return true;
        }
        return false;
    }

    private static int countCV(String vowelString) {
        int output = 0;
        for (int i = 0; i < vowelString.length() - 1; i++) {
            if (vowelString.charAt(i) == 'V' && vowelString.charAt(i + 1) == '
C') {
                output++;
            }
        }
        return output;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        String[] words = new String[n];
        for (int i = 0; i < words.length; i++) {
            words[i] = sc.next();
        }
        for (int i = 0; i < words.length; i++) {
```

```
            String vowelString = "";
            for (int j = 0; j < words[i].length(); j++) {
                if (isVowel(words[i].charAt(j))) {
                    if (vowelString.length() == 0 || vowelString.charAt(vowelS
tring.length() - 1) == 'C') {
                        vowelString += "V";
                    }
                } else {
                    if (vowelString.length() == 0 || vowelString.charAt(vowelS
tring.length() - 1) == 'V') {
                        vowelString += "C";
                    }
                }
            }
            System.out.println(vowelString);
            System.out.println(countCV(vowelString));
        }
        sc.close();
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> javac .\Q6.java
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> java Q6
5
CLASS
GLUE
COMPATIBILITY
ASSESSMENT
PROGRAM
CVC
1
CV
0
CVCVCVCVCVC
5
VCVCVC
3
CVCVC
2
```

Question 7:

// This is marked as Q6 in the cyclesheet

Given a Python dictionary definition statement as an input string like the one shown below, write a Java program to create two arrays – one with keys as its elements and the other with the sum of the tuple elements.

Sample Input:

Eg., Input string = “ mydict={‘A’: (1,-2,3), ‘B’: (4,8), ‘C’: (3,6,-4,5), ‘D’: (1,7,8,-2,-6), ‘E’: (9,10)}”

Sample Output:

Array 1 =[“A”,”B”,”C”,”D”,”E”]

Array 2= [2,12,10,8,19]

Code:

```java
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

// mydict={'A':(1,-2,3),'B':(4,8),'C':(3,6,-4,5),'D':(1,7,8,-2,-6),'E':(9,10)}
public class Q7 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String inp = sc.nextLine();
        sc.close();
        List<Character> keys = new ArrayList<Character>();
        List<Integer> sums = new ArrayList<Integer>();
        int i = 0;
        while (inp.charAt(i) != '{') {
            i++;
        }
        while (inp.charAt(i) != '}') {
            if (Character.isAlphabetic(inp.charAt(i))) {
                keys.add(inp.charAt(i));
            }
            if (inp.charAt(i) == '(') {
                i++;
                int sum = 0;
                String nums = inp.substring(i, inp.indexOf(")", i));
                String[] numbers = nums.split(",");
                for (String string : numbers) {
                    sum += Integer.parseInt(string);
                }
                sums.add(sum);
            }
            i++;
        }
        System.out.println("[");
        for (int j = 0; j < keys.size(); j++) {
            System.out.print("\t\"" + keys.get(j) + "\", ");
        }
        System.out.println("\n]");
        System.out.println("[");
        for (int j = 0; j < keys.size(); j++) {
            System.out.print("\t" + sums.get(j) + ", ");
        }
        System.out.println("\n]");
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> javac .\
Q7.java
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne> java Q7
mydict={'A':(1,-2,3),'B':(4,8),'C':(3,6,-4,5),'D':(1,7,8,-2,-6),'E':(9,10)}
[
        "A",    "B",    "C",    "D",    "E",
]
[
        2,      12,     10,     8,      19,
]
```

Question 8:

Define a class 'Encoding' with the instance variables – *inputtext* (a string), SA (a String array to store the individual words of the inputtext)and IA (an integer array to store the encoded values). [Note: Each object of this class will have a copy of inputtext, SA and IA]. Define a constructor, to split the inputtext into words and store them in the String array, SA. In the main class, read 'n' such input texts and store their details in an array of 'n' objects. Define a method, *sort()* in the 'Encoding' class to sort the individual words of all the 'n' input texts, in alphabetical order. Define another method, *encodeText()* to encode the individual words of each text based on the index of the word in the sorted list and store them in the integer array. Overload this method to make the integer array of each inputtext to have equal number of elements by padding with the value, -1.

Invoke all these methods from the main class.

Sample Input:

3

this is an example for classes

students attend their classes online

online classes are also equally effective

Output:

Sorted List of Words:

also an are attend classes effective equally example for is online students their this

Encoded List from the first encodeText()

[13 9 1 7 8 4]

[11 3 12 4 10]

[10 4 2 0 6 5]

Encoded List from the second encodeText()

[13 9 1 7 8 4]

[11 3 12 4 10 -1]

[10 4 2 0 6 5]

Code:

```
package Q8;

public class Encoding {
    public static String[] overAllStrings = new String[0];
    public String inputtext;
    public String[] SA;
    public int[] IA;

    private static String[] removeTheElement(String[] arr, int index) {
        String[] newArr = new String[arr.length - 1];
        int nextInput = 0;
        for (int i = 0; i < arr.length; i++) {
            if (i != index) {
                newArr[nextInput] = arr[i];
                nextInput++;
            }
        }
        return newArr;
    }

    public static int indexOf(String[] arr, String item) {
        for (int i = 0; i < arr.length; i++) {
            if (arr[i].equals(item)) {
                return i;
            }
        }
        return -1;
    }

    private static String[] concatenate(String[] s1, String[] s2) {
        String[] s3 = new String[s1.length + s2.length];
        for (int i = 0; i < s1.length; i++) {
            s3[i] = s1[i];
        }
        for (int j = 0; j < s2.length; j++) {
            s3[s1.length + j] = s2[j];
        }
        return s3;
    }

    private static int[] concatenate(int[] s1, int[] s2) {
        int[] s3 = new int[s1.length + s2.length];
        for (int i = 0; i < s1.length; i++) {
            s3[i] = s1[i];
        }
        for (int j = 0; j < s2.length; j++) {
            s3[s1.length + j] = s2[j];
```

```
        }
        return s3;
    }

    private static int max(int[] nums) {
        int max = nums[0];
        for (int i = 0; i < nums.length; i++) {
            if (max < nums[i]) {
                max = nums[i];
            }
        }
        return max;
    }

    public Encoding(String inptt) {
        inputtext = inptt;
        SA = inputtext.split(" ");
        overAllStrings = concatenate(overAllStrings, SA);
        IA = new int[SA.length];
    }

    public static void sort() {
        for (int i = 0; i < overAllStrings.length; i++) {
            for (int j = i + 1; j < overAllStrings.length; j++) {
                if (overAllStrings[i].compareTo(overAllStrings[j]) > 0) {
                    String temp = overAllStrings[i];
                    overAllStrings[i] = overAllStrings[j];
                    overAllStrings[j] = temp;
                }
            }
        }
        for (int i = 1; i < overAllStrings.length; i++) {
            if (overAllStrings[i - 1].equals(overAllStrings[i])) {
                overAllStrings = removeTheElement(overAllStrings, i);
            }
        }

    }

    public void encodeText() {
        sort();
        for (int i = 0; i < SA.length; i++) {
            IA[i] = indexOf(overAllStrings, SA[i]);
        }
    }

    public static void displayFullSorted() {
        sort();
```

```
        for (int i = 0; i < overAllStrings.length; i++) {
            System.out.print(overAllStrings[i] + " ");
        }
        System.out.println("");
    }

    public static void encodeText(Encoding[] e) {
        int[] lengths = new int[e.length];
        for (int i = 0; i < e.length; i++) {
            lengths[i] = e[i].IA.length;
        }
        int maxLen = max(lengths);
        for (int i = 0; i < e.length; i++) {
            while (e[i].IA.length < maxLen) {
                int[] x = new int[] { -1 };
                e[i].IA = concatenate(e[i].IA, x);
            }
        }
    }

    public void displayIntegerArray() {
        encodeText();
        System.out.print("[");
        for (int i = 0; i < IA.length; i++) {
            System.out.print(IA[i] + ",");
        }
        System.out.println("]");
    }
}
```

```
package Q8;

public class Execution {
    public static void main(String[] args) {
        Encoding e1 = new Encoding("this is an example for classes");
        Encoding e2 = new Encoding("students attend their classes online");
        Encoding e3 = new Encoding("online classes are also equally effective"
);
        Encoding[] arr = new Encoding[] { e1, e2, e3 };
        Encoding.displayFullSorted();
        for (Encoding encoding : arr) {
            encoding.displayIntegerArray();
        }
        // System.out.println(Encoding.indexOf(new String[] { "ABC", "DEF", "G
HI" }, new
        // String("GHI")));
        Encoding.encodeText(arr);
```

```
        for (Encoding encoding : arr) {
            encoding.displayIntegerArray();
        }
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q8> javac .\Execution.java
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q8> java Execution
also an are attend classes classes effective equally example for is online students their this
[13,9,1,7,8,4,]
[11,3,12,4,10,]
[10,4,2,0,6,5,]
[13,9,1,7,8,4,]
[11,3,12,4,10,-1,]
[10,4,2,0,6,5,]
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q8>
```

Question 9:

There is a class 'Faculty' with two methods - *findClassAverage()* to calculate the internal marks average of the entire class and *findMaxScore(),* a private method to print the highest internal marks score. In the main class, read *n*, the number of students and instantiate the 'Faculty' class by passing *n* and invoke its two methods. The 'Faculty' class alone knows the existence of another class 'Student' and it should create an array of *n* 'Student' objects. The 'Student' class should have two instance variables – *sum* and *marks[]* (an integer array of size 5 to store CAT-1,CAT-2,DA-1,DA-2 and DA-3 marks). Define a method *getIndividualTotal()* in 'Student' class that calculates and returns the total marks. Use the individual total returned by this method in *findClassAverage()*.

Sample Input:

3 (n, the number of students)
1 (marks of student 1)
2 (marks of student 1)
3 (marks of student 1)
4 (marks of student 1)
5 (marks of student 1)
6 (marks of student 2)
7 (marks of student 2)
8 (marks of student 2)
9 (marks of student 2)
10 (marks of student 2)
11 (marks of student 3)
12 (marks of student 3)
13 (marks of student 3)
14 (marks of student 3)
15 (marks of student 3)
Sample Output:
40 (class average)
65 (highest internal marks)

Code:

```java
package Q9;

public class Student {
    public int sum;
    public int[] marks;

    public Student(int[] marksOfStudents) {
        marks = marksOfStudents;
        sum = 0;
    }

    public int getIndividualTotal() {
        sum = 0;
        for (int i = 0; i < marks.length; i++) {
            sum += marks[i];
        }
        return sum;
    }
}
```

```java
package Q9;

import java.util.Scanner;

public class Faculty {
    int n;
    Student[] arr;

    public Faculty(int N) {
        n = N;
        Scanner sc = new Scanner(System.in);
        arr = new Student[N];
        for (int i = 0; i < N; i++) {
            int cat1, cat2, da1, da2, da3;
            cat1 = sc.nextInt();
            cat2 = sc.nextInt();
            da1 = sc.nextInt();
            da2 = sc.nextInt();
            da3 = sc.nextInt();
            arr[i] = new Student(new int[] { cat1, cat2, da1, da2, da3 });
        }
        sc.close();
    }

    public double findClassAverage() {
        double sum = 0;
```

```
        for (Student student : arr) {
            sum += student.getIndividualTotal();
        }
        return sum / (double) n;
    }

    public int findMaxScore() {
        int max = arr[0].getIndividualTotal();
        for (Student student : arr) {
            if (max < student.getIndividualTotal()) {
                max = student.getIndividualTotal();
            }
        }
        return max;
    }
}
```

```
package Q9;

public class Execution {
    public static void main(String[] args) {
        Faculty obj = new Faculty(3);
        System.out.println(obj.findClassAverage());
        System.out.println(obj.findMaxScore());
    }
}
```

Output:

```
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q9> javac
 .\Execution.java
PS C:\Users\sanka\OneDrive\Documents\.SEM3\CSE1007 - Java Programming\LAB\CyclesheetOne\Q9> java
Execution
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
40.0
65
```