Assignment 4

Shambhawi Sharma

Roll no: 09

PRN:1132220067

1:-Create a class called Employee.

Make sure you write a description of your new Class in the comments, with your name as

author and date as the last date you worked on this exercise.

Add definitions for the following fields:

• A name field of type String

• An employeeId field of type int

• A wage field of type double

• A fullTime field of type boolean

Write a constructor for your Employee class that takes four parameters - the first of type

String called myName, the second of type int called myEmployeeId, the third of type double

called myWage, and the fourth of type boolean called isFullTime. Set the initial values of the

corresponding fields using the constructor.

Write an accessor method called getName that returns the value of the name field.

Write a mutator method called setEmployeeId that takes a single parameter of type int and

sets the value of the employeeId field.

Work out what other accessor and mutator methods would be useful for this Class and add

them. You should be able to get and set all fields in the Class.

Write a method called printDetails, which prints out all the details of an Employee object. You

must take into account the fullTime status and print a line saying either that the employee is

fulltime or the employee is not fulltime.

For example, if:

• The name field holds the value "John Smith"

• The employeeId field holds the value 123456

• The wage field holds the value 25.40

• The fullTime field holds the value false

Then the printDetails method would print out the following:

The name of the employee is John Smith. The employee id is 123456. The wage of the

employee is $25.40 per hour. The employee is fulltime.

If the fulltime field holds the value false, then the printDetails method would print out the

following:

The name of the employee is John Smith. The employee id is 123456. The wage of the

employee is $25.40 per hour. The employee is not fulltime.

Please Note: In the above examples, the name, employeeId, wage and whether the employee

is fulltime or not (in blue) will change based on the values the fields hold. However, you must

print the remainder of the statements exactly as in the above examples.

import java.util.Scanner**;**public class Employee {  
 private String myName**;** private int myEmployeeid**;** private double myWage**;** private boolean fullTime**;** Employee(String myName**,**int myEmployeeid**,**double myWage**,**boolean fullTime){  
 this.myName=myName**;** this.myEmployeeid=myEmployeeid**;** this.myWage=myWage**;** this.fullTime=fullTime**;** }  
 String Accessor(){  
 return getName()**;** }  
  
 private String getName() {  
 return null**;** }  
 int Mutator(){  
 return setEmployeeid()**;** }  
  
 private int setEmployeeid() {  
 return myEmployeeid**;** }  
  
 double Wage(){  
 return myWage**;** }  
 private double myWage(){  
 return myWage()**;** }  
  
  
 void printDetails(){  
 if(fullTime==true) {  
 System.*out*.println("Employee Name:- "+myName+"\tEmployee id:-"+myEmployeeid+"\tEmployee Wage:-"+myWage+"\tEmployee is full time")**;** }  
 else {  
 System.*out*.println("Employee Name:-"+myName+"\tEmployee id:-"+myEmployeeid+"\tEmployee Wage:-"+myWage+"\tEmployee is not a full time")**;** }  
 }  
  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*)**;** System.*out*.println("Enter the employee name ")**;** String myName =sc.next()**;** System.*out*.println("Enter the employee id ")**;** int myEmployeeid = sc.nextInt()**;** System.*out*.println("Enter the wage of employee")**;** double myWage= sc.nextDouble()**;** System.*out*.println("Enter True or False")**;** boolean fullTime=sc.hasNextBoolean()**;** Employee e = new Employee(myName**,**myEmployeeid**,**myWage**,**fullTime)**;** e.Accessor()**;** e.Mutator()**;** e.Wage()**;** e.printDetails()**;** }  
}

2:- Create a new BlueJ project called YourFirstNameLastName-A1Q3.

Create a class, Assignment, that contains the following four fields:

• A String called StudentName

• A double called assignmentMark (which will store the mark each assignment is worth

e.g. for this assignment that you are doing right now the value would be 20)

• A double called studentMark (stores the mark the student gets in the assignment e.g.

15)

• A String called grade

Make sure you write a description of your new Class in the comments, with your name as

author and give the version as the date you last worked on this exercise.

Define a constructor that takes and sets the studentName, studentMark and

assignmentMark.

Also define a constructor that takes no parameters and sets the assignmentMark to 100.

Create an accessor and mutator for studentMark. The mutator should not let the

studentMark be set a value greater than the assignmentMark (as the student cannot get a

mark higher than the assignment is worth) or less than 0. If the user tries to set a value that

is not valid a suitable error message should be displayed.

Create a method that calculates the grade for the student. You will need to work out how

many percent the student scored in the assignment.

If the student scored:

• Less than 50% the grade will be fail

• 50% - 64% the grade will be pass

• 65% – 74% the grade will be credit

• 75% – 84% the grade will be distinction

• Greater than 85% the grade will be high distinction

For example, if assignmentMark is 30 and studentMark is 15, the percentage will be 50% so

the grade will be set to pass.

Define an accessor method to return the value of grade.

->public class Assignment {  
 String StudentName**;** double AssignmentMark**;** double StudentMark**;** String Grade**;** Assignment(String StudentName **,** double AssignmentMark **,** double StudentMark){  
 this.StudentName=StudentName**;** this.AssignmentMark=AssignmentMark**;** this.StudentMark=StudentMark**;** this.Grade=Grade**;** }  
 Assignment(){  
 AssignmentMark=100**;** }  
 double Accessor(){  
 return CalculateGrade()**;** }  
 private double getStudentMarK() {  
 return StudentMark**;** }  
  
 double Mutator(){  
 Assignment a = new Assignment()**;** a.getStudentMarK()**;** if(StudentMark < AssignmentMark || StudentMark>0){  
 return StudentMark**;** }else {  
 System.*out*.println("Enter the Valid Marks")**;** }  
 return -1**;** }  
  
  
 double CalculateGrade(){  
 double Grade= (StudentMark/AssignmentMark)\*100**;** if(Grade<50){  
 System.*out*.printf("fail")**;** } else if (Grade>=50 && Grade<=64) {  
 System.*out*.printf("Pass")**;** } else if (Grade>=65 && Grade<=74) {  
 System.*out*.printf("credit")**;** } else if (Grade>=75 && Grade<=84) {  
 System.*out*.printf("distinction")**;** }else {  
 System.*out*.printf("high distinction")**;** }  
 return Grade**;** }  
  
 public static void main(String[] args) {  
 Assignment a = new Assignment("Aditya"**,**30**,**30)**;** a.Mutator()**;** a.CalculateGrade()**;** }  
  
}

3:-Part A:

Imagine you need to write a program for a 24-hour clock with hours, minutes and seconds.

Write a Java program in BlueJ with a method that prints all possible times the clock could

display starting at 00:00:00 through to 23:59:59 when all the three numbers are the same

(e.g. 01:01:01, 02:02:02, 13:13:13 and so on)

->public class Time {  
 public static void main(String[] args) {  
  
  
  
 int hor = 23**;** int min = 59**;** int sec = 59**;** for (int i = 0**;** i <= hor**;** i++) **;** {  
 System.*out*.println(hor+":"+ hor+":"+ hor)**;** }  
  
 }  
}

Part B:

Write a second method that takes three (3) parameters – hours, minutes and seconds. This

method will print out all of the possible times the clock could display from one hour before

the time passed to the method till one hour after the time passed to the method when all the

three numbers are even (e.g. 12:20:00 or 12:20:02, NOT 12:20:01). For example:

If the method was passed the following values:

Hour = 11

Minutes = 23

Seconds = 44

The method would print all the times the clock could display from 10:23:44 until 12:23:44

when all the three numbers (hours, minutes and seconds) are even. The first printed time

would be 10:24:00, and the last one would be 12:22:58)

import java.util.Scanner**;**public class Time2 {  
 public static void main(String[] args) {  
 String first ="10:24:00"**;** String last = "12:22:58"**;** int hours **,**minutes**,**seconds**;** Scanner sc = new Scanner(System.*in*)**;** hours = sc.nextInt()**;** minutes = sc.nextInt()**;** seconds = sc.nextInt()**;** if(hours %2==0 && minutes%2==0 && seconds%2==0){  
 System.*out*.println("first printed time time:"+first+" last printed time:"+last)**;** }  
 else {  
 System.*out*.println("hours or minutes or second are not even")**;** }  
 }  
}

Create a class called ListOfNames, that has one ArrayList field called names, which holds a

collection of Strings (each string is a male or female name in Upper case e.g. PETER).

Make sure you write a description of your new Class in the comments, with your name as

author and give the version as the date you last worked on this exercise.

5

Define a constructor that initialises the ArrayList. Note that you can also add any other

initialisations that you feel are relevant.

Create methods to add elements, remove elements and get the number of elements in the

collection. Add a test for all three of these methods to check whether the operation was

successful and print a message letting the user know if it was or was not.

Create a method called printNames. This method should loop through the collection and print

out the elements (each String on a new line) as determined by the following rules:

• If the string contains any vowels (A, E, I, O and U), the method should print "The name

" + the value of the String + " contains vowels, and the vowels are:" + list of the vowels

in the string. The string may have more than one vowels. For example, if the name is

EMMA it would print: The name EMMA contains vowels, and the vowels are: E, A

• If the string contains duplicate characters, the method should print "The name " + the

value of the String + " has the following duplicate character(s):" + list of the duplicate

characters in the string. The string may have one or more duplicate character. For

example, if the name is ANNABELLA, it would print: The name ANNABELLA has the

following duplicate character(s): A, N, L

• If the string contains any vowels (A, E, I, O and U), and it has duplicate characters, the

method should print "The name " + the value of the String + " contains vowels and has

duplicate characters". For example, if the name is LARISSA it would print: The name

LARISSA contains vowels and has duplicate characters.

• If none of the above criteria is met, then the method should print the String element

in lower case. For example, if the name is SKY, it would print: sky

Once you have finished your project, open the terminal window in BlueJ and turn on record

method calls. Create a new ListOfNames object, and then add at least ten (10) Strings using

the add method you wrote. You must have:

• A String that contains vowels

• A String that has duplicate characters

• A String that contains vowels and has duplicate characters

Demonstrate removing an element using the remove method you wrote, and then find the

number of elements using the method you wrote that gets the number of elements. Finally,

run your printNames method.

SOLUTION:

import java.util.ArrayList;

import java.util.Scanner;

public class Q4 {

    ArrayList<String> names = new ArrayList<String>();

    Scanner sc = new Scanner(System.in);

    // add method to take the values upto n values

    public void add(int n) {

        for (int i = 0; i < n; i++) {

            System.out.print("Enter the name:");

            String s = sc.nextLine();

            System.out.println();

            names.add(s);

            names.replaceAll(String::toUpperCase);

        }

        System.out.println(names);

    }

    // removing element from a particular location

    public void remove(int pos) {

        // removing element from n-1 location

        names.remove(pos);

        System.out.println(names);

    }

    // public void get(int pos) {

    //     System.out.println(names.get(pos));

    // }

    public void printNames(){

        String str;

        int i,j;

        for ( i = 0; i < names.size(); i++)

      {

        str = names.get(i);

        for(j=0;j<str.length();j++){

        if(str.charAt(j)=='A' || str.charAt(j)=='E' || str.charAt(j)=='I' || str.charAt(j)=='U' || str.charAt(j)=='O')

        {

            System.out.println("The name "+str+" contains vowels "+"and the vowels are "+str.charAt(j));

        }

}

            int count;

            //Converts given string into character array

            char string[] = str.toCharArray();

            System.out.println("Duplicate characters in a given string: ");

            //Counts each character present in the string

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

                count = 1;

                for(j = i+1; j <string.length; j++) {

                    if(string[i] == string[j] && string[i] != ' ') {

                        count++;

                        //Set string[j] to 0 to avoid printing visited character

                        string[j] = '0';

                    }

                }

                //A character is considered as duplicate if count is greater than 1

                if(count > 1 && string[i] != '0')

                    System.out.println(string[i]);

            }

        }

        }

// System.out.println(name);

    public static void main(String args[]) {

        Q4 q = new Q4();

        q.add(1);

        // q.remove(2);

        // q.get(2);

        q.printNames();

    }

}