**PROGRAMMING ASSIGNMENT 1**

Class: S.E. Computer (Division A and B)

Date of Assignment: 26-09-2023

Date of Submission: 30-09-2023

| **CSL304.1** | **Demonstrate fundamental programming constructs** |
| --- | --- |
| **CSL304.2** | **Illustrate the concept of packages, classes, and objects** |
| **CSL304.3** | **Apply the concept of strings, arrays, and vectors** |

NOTE: Solve the following questions using Java. Submit code along with appropriate output cases.

| Q. | Description | CO mapping |
| --- | --- | --- |
| Q1 | Write a Java method to check whether a string is a valid password. Password rules: A password must have at least ten characters. A password consists of only letters and digits. A password must contain at least two digits.  Expected Output:  1. A password must have at least eight characters.  2. A password consists of only letters and digits.  3. A password must contain at least two digits  Input a password (You are agreeing to the above Terms and Conditions.): abcd1234  Password is valid: abcd1234 | CSL304.3 |
| Q2 | Write a Java method that accepts three integers and checks whether they are consecutive or not. Returns true or false.  Expected Output:  Input the first number: 15  Input the second number: 16  Input the third number: 17  Check whether the three said numbers are consecutive or not!true | CSL304.1 |
| Q3 | One class data member of another class  class Department{  }  class College {  private Department dt;  //Constructor  }  Department class can have the fields dept\_id, dept\_name and dept\_hod  College class can have the fields college\_id, college\_name, college\_principal, college\_departments as data members of college class.  Provide a constructor in College class which can receive object of Department class as parameter and assign it to the data member and also provide a method in College class which can print the data related to College and Department object  Consider to have Department class in com.p1 package, College class in com.p2 package, Main program in com.p3 package | CSL304.2 |
| Q4 | Write a Java program to remove duplicate elements from an array. | CSL304.3 |
| Q5 | Write a Java program that reads a floating-point number. If the number is zero it prints "zero", otherwise, print "positive" or "negative". Add "small" if the absolute value of the number is less than 1, or "large" if it exceeds 1,000,000. | CSL304.1 |

Q1. ⇒

Code:

*// #bug: the problem statement ask to have minimum length of 10 but the input and output indicate minimum 8 characters required. I went with minimum length of 8. To change to minimum 10 character go to line 9 and increase the condition from 8 to 10.*

import java.util.Scanner;

public class PasswordValidator {

*// return is boolean to tell whether the password is valid or not*

public static boolean isValid(String *password*) {

*// makes sure that the password is 8 characters long.*

if (*password*.length() < 8) {

System.out.println("Retry: The password must have at least ten characters.");

return false;

}

*// uses a regex expression to ensure that all the character belong to either [a-z] or [A-Z] or [0-9] character group regardless of their sequence*

if (!*password*.matches("[a-zA-Z0-9]+")) {

System.out.println("Retry: The password must consists of only letters and digits.");

return false;

}

*// atleast 2 digits*

int digitCount = 0;

for (char c : *password*.toCharArray()) {

*// makes sure that the char c is a digit or not*

if (Character.isDigit(c)) {

digitCount++;

if (digitCount >= 2) {

break; *// condition completed so no need to loop further*

}

}

}

if (digitCount < 2) {

System.out.println("3. The password must contain at least two digits.");

return false;

}

*// If all rules are satisfied, the password is valid.*

return true;

}

public static void main(String[] *args*) {

Scanner sc = new Scanner(System.in);

System.out.print("1. A password must have at least eight characters.\n2. A password consists of only letters and digits.\n3. A password must contain at least two digits\nInput a password (You are agreeing to the above Terms and Conditions.) : ");

String pass = sc.nextLine();

*// prints the statement based on the validity of the password*

if (isValid(pass)) {

System.out.println("Password is valid: " + pass);

} else {

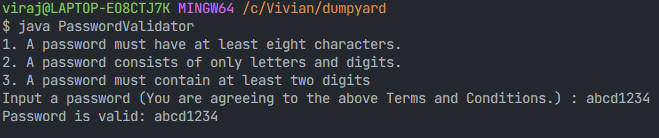
System.out.println("Password is invalid.");

}

}

}

OUTPUT:



Q2. ⇒

Code:

import java.util.Scanner;

public class ConsecutiveNumberChecker {

public static void main(String[] *args*) {

Scanner sc = new Scanner(System.in);

System.out.print("Input the first number: ");

int num1 = sc.nextInt();

System.out.print("Input the second number: ");

int num2 = sc.nextInt();

System.out.print("Input the third number: ");

int num3 = sc.nextInt();

boolean result = areConsecutiveNumbers(num1, num2, num3);

System.out.println("Check whether the three said numbers are consecutive or not!" + result);*//*

}

public static boolean areConsecutiveNumbers(int *num1*, int *num2*, int *num3*) {

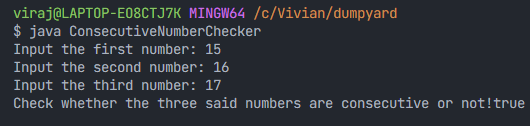
*// checks if absolute diff between two numbers is 1. absolute diff taken cause input can be ascending or descending.*

return (Math.abs(*num2* - *num1*) == 1 && Math.abs(*num3* - *num2*) == 1);

}

}

OUTPUT:



Q3. ⇒

Code:

**In com>p1>Department.java**

package com.p1;

public class Department {

private int dept\_id;

private String dept\_name;

private String dept\_hod;

public Department(int *dept\_id*, String *dept\_name*, String *dept\_hod*) {

this.dept\_id = *dept\_id*;

this.dept\_name = *dept\_name*;

this.dept\_hod = *dept\_hod*;

}

public int getDept\_id() {

return dept\_id;

}

public void setDept\_id(int *dept\_id*) {

this.dept\_id = *dept\_id*;

}

public String getDept\_name() {

return dept\_name;

}

public void setDept\_name(String *dept\_name*) {

this.dept\_name = *dept\_name*;

}

public String getDept\_hod() {

return dept\_hod;

}

public void setDept\_hod(String *dept\_hod*) {

this.dept\_hod = *dept\_hod*;

}

}

**In com>p2>College.java**

**package com.p2;**

**import com.p1.Department; *// import dept***

**public class College {**

**private int college\_id;**

**private String college\_name;**

**private String college\_principal;**

**private Department college\_departments; *// Reference to Department object***

**public College(int *college\_id*, String *college\_name*, String *college\_principal*, Department *college\_departments*) {**

**this.college\_id = *college\_id*;**

**this.college\_name = *college\_name*;**

**this.college\_principal = *college\_principal*;**

**this.college\_departments = *college\_departments*;**

**}**

**public void printInfo() {**

**System.out.println("College ID: " + college\_id);**

**System.out.println("College Name: " + college\_name);**

**System.out.println("College Principal: " + college\_principal);**

**System.out.println("Department ID: " + college\_departments.getDept\_id());**

**System.out.println("Department Name: " + college\_departments.getDept\_name());**

**System.out.println("Department HOD: " + college\_departments.getDept\_hod());**

**}**

**}**

**In com>p3>Main.java**

**package com.p3;**

**import com.p1.Department;**

**import com.p2.College;**

**public class Main {**

**public static void main(String[] *args*) {**

***// Department object***

**Department department = new Department(1, "Computer Engineering", "Dr. Sujata Deshmukh");**

***// Create a College object and pass Department object in it***

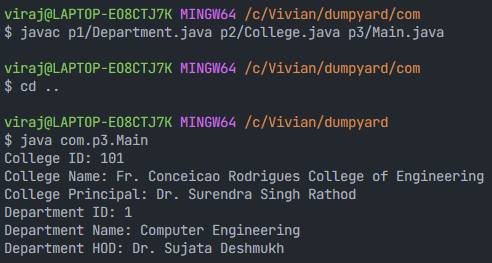
**College college = new College(101, "Fr. Conceicao Rodrigues College of Engineering", "Dr. Surendra Singh Rathod", department);**

***// call print from college***

**college.printInfo();**

**}**

**}**

OUTPUT:

Q4. ⇒

Code:

import java.util.\*;

public class Duplicate {

public static void **main**(String *args*[]) {

Scanner sc = new **Scanner**(System.in);

System.out.**println**("Enter the number of array elements: ");

int n = sc.**nextInt**();

int[] arr = new int[n];*// array to hold the elements*

System.out.**println**("Enter " + n + " array element");

for (int i = 0; i < n; i++) {

arr[i] = sc.**nextInt**();

}

*// array of n integers filled*

*// more efficient solution would be to store the elements in a set.*

for (int i = 0; i < n; i++) {

int comp = arr[i];

for (int j = i + 1; j < n;) {

if (comp == arr[j]) {

*// since last element is shifted loop only till n-2*

for (int k = j; k < n - 1; k++) {

arr[k] = arr[k + 1];

}

n--;*// reduce n by one since the duplicate is removed*

}

else {

j++;

}

}

}

*// the size of the array will not change it will still remain the same even after deletion of the element*

System.out.**println**("After removing duplicate elements: ");

for (int i = 0; i < n; i++) {

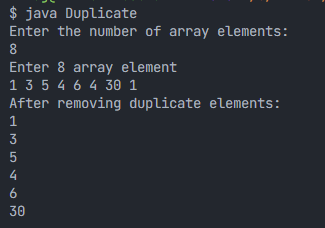
System.out.**println**(arr[i]);

}

}

}

OUTPUT:



Q5. ⇒

Code:

import java.util.Scanner;

public class FloatBasedOutput {

public static void **main**(String[] *args*) {

Scanner sc = new **Scanner**(System.in);

*// double is also a floating point number*

System.out.**print**("Enter a floating-point number: ");

double number = sc.**nextDouble**();

if (number == 0) {

System.out.**println**("Zero");

} else if (number > 0) {

System.out.**print**("Positive");

if (Math.**abs**(number) < 1) {

System.out.**println**(" and Small");

} else if (Math.**abs**(number) > 1000000) {

System.out.**println**(" and Large");

} else {

*// if no conditions match then print empty line ot end*

System.out.**println**();

}

} else {

System.out.**print**("Negative");

if (Math.**abs**(number) < 1) {

System.out.**println**(" and Small");

} else if (Math.**abs**(number) > 1000000) {

System.out.**println**(" and Large");

} else {

System.out.**println**();

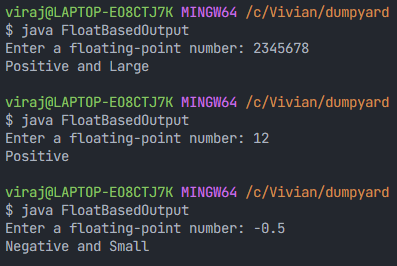
}

}

}

}

OUTPUT:



**RUBRICS for Programming Assignment Grading:**

| Sr. No | Performance Indicator | Below average | Average | Good | Excellent | Marks |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | On time Submission (2) | - | Submitted after deadline (1) | Early or on time submission(2) |  |  |
| 2 | Test cases and output  (4) | Incorrect output  (1) | Expected output is verified only for few test cases (2) | Expected output is Verified for all test cases but is not presentable (3) | Expected output is obtained for all test cases. Presentable and easy to follow (4) |  |
| 3 | Coding efficiency (2) | The code is not structured at all.(0) | The code is structured but not efficient (1) | The code is structured and efficient. (2) | - |  |
| 4 | Knowledge(2) | Basic concepts not clear  (0) | Understood the basic concepts (1) | Could explain the concept with suitable example (1.5) | Could relate the theory with real world application(2) |  |
| Total Marks | | | | | | 10 |

\*\*\*\*\*\*\*\*\*\*\*