# **Object Oriented Programming**

# **COMP 2477 WA**

# **Assignment 4**

**Name:- Kirtan Sukhadiya**

**Student ID:- 1222983**

**Q1)**  
Code:-

import java.util.Scanner;  
  
class LetterGrade {  
 protected double score;  
  
 public LetterGrade() {  
 this.score = 0;  
 }  
  
 public LetterGrade(double score) {  
 this.score = score;  
 }  
  
 public void setScore(double score) {  
 this.score = score;  
 }  
  
 public double getScore() {  
 return score;  
 }  
  
 public char getLetterGrade() {  
 if (score >= 90) return 'A';  
 else if (score >= 80) return 'B';  
 else if (score >= 70) return 'C';  
 else if (score >= 60) return 'D';  
 else return 'F';  
 }  
}  
  
class Essay extends LetterGrade {  
 public Essay(double grammar, double spelling, double length, double content) {  
 super(grammar + spelling + length + content);  
 }  
}  
  
class PassFailExam extends LetterGrade {  
 public PassFailExam(int totalQuestions, int missedQuestions) {  
 double pointsEach = 100.0 / totalQuestions;  
 double score = 100 - (missedQuestions \* pointsEach);  
 this.setScore(score);  
 }  
  
 @Override  
 public char getLetterGrade() {  
 return score >= 70 ? 'P' : 'F';  
 }  
}  
  
class FinalExam extends LetterGrade {  
 public FinalExam(int totalQuestions, int missedQuestions) {  
 double pointsEach = 100.0 / totalQuestions;  
 double score = 100 - (missedQuestions \* pointsEach);  
 this.setScore(score);  
 }  
}  
  
class CourseGrades {  
 private final LetterGrade[] grades;  
  
 public CourseGrades() {  
 grades = new LetterGrade[4];  
 }  
  
 public void setLab(LetterGrade lab) {  
 grades[0] = lab;  
 }  
  
 public void setPassFailExam(PassFailExam exam) {  
 grades[1] = exam;  
 }  
  
 public void setEssay(Essay essay) {  
 grades[2] = essay;  
 }  
  
 public void setFinalExam(FinalExam finalExam) {  
 grades[3] = finalExam;  
 }  
  
 public void printGrades() {  
 System.*out*.println("\nCourse Grades:");  
 for (int i = 0; i < grades.length; i++) {  
 if (grades[i] != null) {  
 System.*out*.println("Assignment " + (i + 1) + " - Score: "  
 + String.*format*("%.2f", grades[i].getScore())  
 + ", Grade: " + grades[i].getLetterGrade());  
 } else {  
 System.*out*.println("Assignment " + (i + 1) + " - No grade assigned.");  
 }  
 }  
 }  
}  
  
public class Assignment4\_Q1 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 CourseGrades course = new CourseGrades();  
  
 // Get Lab Score  
 System.*out*.print("Enter the Lab Activity score (0-100): ");  
 double labScore = scanner.nextDouble();  
 course.setLab(new LetterGrade(labScore));  
  
 // Get Pass/Fail Exam Details  
 System.*out*.print("Enter total questions for the Pass/Fail exam: ");  
 int totalPassFailQuestions = scanner.nextInt();  
 System.*out*.print("Enter number of missed questions: ");  
 int missedPassFail = scanner.nextInt();  
 course.setPassFailExam(new PassFailExam(totalPassFailQuestions, missedPassFail));  
  
 // Get Essay Scores  
 System.*out*.print("Enter Grammar score (0-30): ");  
 double grammar = scanner.nextDouble();  
 System.*out*.print("Enter Spelling score (0-20): ");  
 double spelling = scanner.nextDouble();  
 System.*out*.print("Enter Length score (0-20): ");  
 double length = scanner.nextDouble();  
 System.*out*.print("Enter Content score (0-30): ");  
 double content = scanner.nextDouble();  
 course.setEssay(new Essay(grammar, spelling, length, content));  
  
 // Get Final Exam Details  
 System.*out*.print("Enter total questions for the Final Exam: ");  
 int totalFinalQuestions = scanner.nextInt();  
 System.*out*.print("Enter number of missed questions: ");  
 int missedFinal = scanner.nextInt();  
 course.setFinalExam(new FinalExam(totalFinalQuestions, missedFinal));  
  
 // Print results  
 course.printGrades();  
  
 scanner.close();  
 }  
}

Output:-  
A screenshot of a computer program

AI-generated content may be incorrect.

**Q2)**

Code:-

import java.util.Scanner;  
  
public class Assignment4\_Q2 {  
 // Interface definition  
 public interface Analyzable {  
 double getAverage();  
 GradedActivity getHighest();  
 GradedActivity getLowest();  
 }  
  
 // GradedActivity class  
 public static class GradedActivity {  
 private double score;  
  
 public void setScore(double s) {  
 score = s;  
 }  
  
 public double getScore() {  
 return score;  
 }  
  
 public char getGrade() {  
 if (score >= 90) return 'A';  
 else if (score >= 80) return 'B';  
 else if (score >= 70) return 'C';  
 else if (score >= 60) return 'D';  
 else return 'F';  
 }  
 }  
  
 // CourseGrades class implementing Analyzable  
 public static class CourseGrades implements Analyzable {  
 private final GradedActivity[] grades;  
 private static final int *NUM\_GRADES* = 4;  
  
 public CourseGrades() {  
 grades = new GradedActivity[*NUM\_GRADES*];  
 }  
  
 public void setLab(GradedActivity lab) {  
 grades[0] = lab;  
 }  
  
 public void setPassFailExam(GradedActivity passFailExam) {  
 grades[1] = passFailExam;  
 }  
  
 public void setEssay(GradedActivity essay) {  
 grades[2] = essay;  
 }  
  
 public void setFinalExam(GradedActivity finalExam) {  
 grades[3] = finalExam;  
 }  
  
 @Override  
 public double getAverage() {  
 double total = 0;  
 for (GradedActivity g : grades) {  
 if (g != null) {  
 total += g.getScore();  
 }  
 }  
 return total / *NUM\_GRADES*;  
 }  
  
 @Override  
 public GradedActivity getHighest() {  
 GradedActivity highest = grades[0];  
 for (GradedActivity g : grades) {  
 if (g != null && g.getScore() > highest.getScore()) {  
 highest = g;  
 }  
 }  
 return highest;  
 }  
  
 @Override  
 public GradedActivity getLowest() {  
 GradedActivity lowest = grades[0];  
 for (GradedActivity g : grades) {  
 if (g != null && g.getScore() < lowest.getScore()) {  
 lowest = g;  
 }  
 }  
 return lowest;  
 }  
  
 @Override  
 public String toString() {  
 StringBuilder sb = new StringBuilder();  
 for (int i = 0; i < grades.length; i++) {  
 if (grades[i] != null) {  
 sb.append("Grade ").append(i + 1)  
 .append(": Score = ").append(grades[i].getScore())  
 .append(", Letter Grade = ").append(grades[i].getGrade())  
 .append("\n");  
 }  
 }  
 return sb.toString();  
 }  
 }  
  
 // Main method to take user input  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 CourseGrades courseGrades = new CourseGrades();  
  
 String[] gradeTypes = {"Lab", "Pass/Fail Exam", "Essay", "Final Exam"};  
  
 for (int i = 0; i < gradeTypes.length; i++) {  
 System.*out*.print("Enter score for " + gradeTypes[i] + ": ");  
 double score = scanner.nextDouble();  
 GradedActivity activity = new GradedActivity();  
 activity.setScore(score);  
  
 switch (i) {  
 case 0 -> courseGrades.setLab(activity);  
 case 1 -> courseGrades.setPassFailExam(activity);  
 case 2 -> courseGrades.setEssay(activity);  
 case 3 -> courseGrades.setFinalExam(activity);  
 }  
 }  
  
 scanner.close();  
  
 System.*out*.println("\n===== Course Grades =====");  
 System.*out*.println(courseGrades);  
 System.*out*.println("Average Score: " + courseGrades.getAverage());  
 System.*out*.println("Highest Score: " + courseGrades.getHighest().getScore());  
 System.*out*.println("Lowest Score: " + courseGrades.getLowest().getScore());  
 }  
}

Output:-

A screenshot of a computer

AI-generated content may be incorrect.