DTS102TC Coursework 1 Individual Assignment Marking Criteria

Tasks	100	Marking Criteria	Marks
Question 1	8	**Reports [3 marks]** • **Design Explanation**: Describe the logic for calculating the discriminant and determining the number of real roots (positive, zero, or negative discriminant). [1 mark] • **Test Cases**: Include at least two test cases (e.g., one with two real roots, one with no real roots) with input values, computed results, and verification against expected outputs. [1 mark] • **Code Analysis**: Explain how pow(x, 0.5) is used to compute square roots and how the output is formatted to match the sample runs. [1 mark] **Program execution by Gradescope [5 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) Two roots, i.e., x^2 - 3x + 2 = 0 [2 marks] 2) Single root, i.e., x^2 - 2x + 1 = 0 [2 marks] 3) No real roots, i.e., x^2 + 1 = 0 [1 mark]	
Question 2	8	**Reports [3 marks]** • **Formula Application**: Detail how the area formula is implemented in code, including the use of trigonometric functions. [1 mark] • **Test Cases**: Provide a test case with input values (number of sides and side length) and show the computed area, comparing it to the sample output. [1 mark] • **Precision Note**: Explain how floating-point precision is handled to ensure accurate results. [1 mark] **Program execution by Gradescope [5 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) Example polygon (sides = 5, the length of a side = 6.5) [1 mark] 2) Triangle (the length of a side = 1, 2, 10) [1 mark] 3) Pentagon (the length of a side = 1, 3.14, 6.5) [1 mark] 4) Large polygon (100 sides) [2 marks]	
Question 3	8	**Reports [3 marks]** • **Algorithm Description**: Explain the loop structure for reading input values until 0 is entered, how positive/negative counts are tracked, and how the total/average are calculated (excluding zeros). [1 mark] • **Edge Cases**: Address the scenario where the only input is 0,	

		explaining how the program outputs "No numbers are entered except 0." [1 mark] • **Test Cases**: Include a sample with multiple positive/negative numbers and a test case with only 0, showing counts, total, and average. [1 mark] **Program execution by Gradescope [5 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) Random generated 8 positive numbers [2 marks] 2) Random generated 20 positive and negative numbers [2 marks] 3) Empty set [1 mark]	
Question 4	8	**Reports [3 marks]** • **Conversion Logic**: Describe how the binary string is processed character by character to compute the decimal value (e.g., iterating from left to right, accumulating). [1 mark] • **Test Cases**: Provide at least one test case (e.g., binary string "10001" converting to 17) with step-by-step calculations to verify correctness. [1 mark] • **Code Details**: Explain how characters in the string are converted to integers (e.g., subtracting '0' from the character). [1 mark] **Program execution by Gradescope [5 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) Zero string [1 mark] 2) Simple string (10001) [2 marks] 3) Complex string (11011111101010010000000000) [2 marks]	
Question 5	8	**Reports [3 marks]** • **Data Structure Use**: Describe how an array is used to store distinct numbers (checking for duplicates before insertion) and how the order of input is preserved. [1 mark] • **Test Cases**: Include a test case with duplicate values (e.g., input "1 2 3 2 1 6 3 4 5 2") and show the output of distinct numbers in input order. [1 mark] • **Efficiency Note**: Briefly comment on how the duplicate check is implemented (e.g., linear search through the array). [1 mark] **Program execution by Gradescope [5 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) Example from sample run （1 2 3 2 1 6 3 4 5 2）[1 mark]	

		2) A simple case （2 7 1 3 0 9 9 5 9 4 2 6 1 5 2）[2 marks] 3) Another case with negative numbers （-6 4 -8 -3 -10 9 9 1 9 -1 -5 2 -8 1 -6 -1 -7 -5 -8 10）[2 marks]	
Question 6	13	**Reports [3 marks]** • Function Design: Explain the sumColumn function, including how it iterates over rows to sum elements in the specified column of a 2D matrix. [1 mark] • Test Cases: Use the sample input (3x4 matrix) to show the sum of each column, verifying results (e.g., column 0 sum = 1.5 + 5.5 + 9.5 = 16.5). [1 mark] • Parameter Handling: Describe how rowSize, columnSize, and colIndex are used to avoid out-of-bounds errors. [1 mark] **Program execution by Gradescope [10 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) Example from sample run [4 marks, each column results 1 mark] 2) Unique matrix (random integers) [3 marks] 3) Unique matrix (random decimals) [3 marks]	
Question 7	13	**Reports [3 marks]** • Class Structure: Detail the Rectangle class members (data fields width and height, constructors, accessors/mutators, getArea(), getPerimeter()).[1 mark] • Test Program: Explain the test cases (e.g., first rectangle with width 4/height 40, second with 3.5/35.9) and show output for width, height, area, and perimeter. [1 mark] • OOP Principles: Comment on how encapsulation is achieved (private data fields with public accessors/mutators). [1 mark] **Program execution by Gradescope [10 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) A no-arg constructor that creates a rectangle with width 1 and height 1 [2 marks] 2) A constructor that creates a rectangle with the specified width and height. [2 marks] 3) The accessor and mutator functions for all the data fields.[2 marks] 4) A function named getArea() that returns the area of this rectangle. [2 marks] 5) A function named getPerimeter() that returns the perimeter of this rectangle. [2 marks]	

Question 8	18	**Reports [3 marks]** • **Class Design**: Describe the data fields (x, y, width, height) and methods (constructors, getArea(), getPerimeter(), contains(), overlaps()).[1 mark] • **Method Logic**: Explain the geometric checks for: A point being inside the rectangle (using bounds of x ± width/2 and y ± height/2). A rectangle being inside another (all corners of the smaller rectangle within the larger). Two rectangles overlapping (not fully separate). [1 mark] • **Test Cases**: Provide examples verifying contains(), contains(Rectangle2D),and overlaps().[1 mark] **Program execution by Gradescope [15 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) The double data fields width and height with constant get functions and set functions. [1 mark] 2) A no-arg constructor that creates a default rectangle with (0, 0) for (x, y) and 1 for both width and height. [2 marks] 3) A constructor that creates a rectangle with the specified x, y, width and height, i.e. r(3,4,5,6). [2 marks] 4) A constant function getArea() that returns the area of the rectangle, i.e., Area(r(3,4,5,6)=30) [2 marks] 5) A constant function getPerimeter() that returns the perimeter of the rectangle, i.e., perimeter(r(3,4,5,6)=22) [2 marks] 6) A constant function contains(double x, double y) that returns true if the specified point (x, y) is inside this rectangle.[2 marks] 7) A constant function contains(const Rectangle2D &r) that returns true if the specified rectangle is inside this rectangle. [2 marks] 8) A constant function overlaps(const Rectangle2D &r) that returns true if the specified rectangle overlaps with this rectangle. [2 marks]	
Question 9	13	**Reports [3 marks]** • **Algorithm**: Explain how the bounding rectangle is computed by finding the minimum/maximum x and y coordinates from the input points, then calculating center, width, and height. [1 mark] • **Function Comparison**: Briefly contrast getRectangle (returns a Rectangle2D object) and getRectanglePointer (returns a pointer to the object). [1 mark] • **Test Cases**: Use the sample input (five points) to show the computed center (5.0, 6.25), width 8.0, and height 7.5,	

		verifying correctness. [1 mark] **Program execution by Gradescope [10 marks]** Program testing and execution. Performs the calculation correctly and arrives at the correct answer for the given example 1) Test with sample run (1.0 2.5 3 4 5 6 7 8 9 10) [5 marks] 2) Test with an additional example (3 4 1 2.5 9 10 7 8 114 514) [5 marks]	
Report quality	3	Logical structure follows the template, adherence to 5-page limit, PDF format, no major grammar/spelling errors, etc.	