# Complete your details and read the information below. Please attach this sheet to your online submission to confirm that you understand and accept the **ON**CAMPUS academic regulations regarding coursework.

|  |  |
| --- | --- |
| **Student Full Name:** | Matthew Brian Tahir |
| **Student ID Number:**  (e.g. 12345) | 18630 |
| **Date of Birth:**  (DD/MM/YYYY) | 03/04/2004 |
| **Programme:**  (e.g. UFP) | IY1 CS |
| **Programme Start Date:** (e.g. September 2020) | September 26, 2022 |

Please confirm your understanding that:

1. At **ON**CAMPUS, we recognise that English may not be your first language. Our assessments are designed to allow you to achieve while continuing to develop your academic English skills. We expect all submitted work to be your own words (apart from in-text quotations), written in a style that reflects your English language level.

2. At any time, you may be asked to attend an interview with at least two members of academic staff to discuss the content of your submitted coursework. It is likely that one interviewer will be the Centre Head or Deputy Centre Head. Interviews may be requested as part of a random check, or if there is suspicion that an academic offence has been committed. We expect all students to be able to discuss and/or explain the main ideas and vocabulary used in submitted coursework.

3. It is your responsibility to read the programme handbook, including the policy on late submission, and information on available support if you are finding your studies difficult.

4. If you are unsure about any part of your coursework, you must ask your tutors for help before the submission deadline.

By attaching this sheet, you confirm that that you understand and accept all of the information above. You understand that all submitted coursework must include this coursework submission sheet, and that submissions without the sheet included will be rejected.

Contents

I. [Complete your details and read the information below. Please attach this sheet to your online submission to confirm that you understand and accept the **ON**CAMPUS academic regulations regarding coursework. 1](#_Toc135133243)

[II. Ability to demonstrate the requirements assessment stage 3](#_Toc135133244)

[1. Analysing requirements 3](#_Toc135133245)

[2. The Activity timeline 4](#_Toc135133246)

[3. PERT Chart 5](#_Toc135133247)

[III. The design specification for the application 6](#_Toc135133248)

[The Storyboard 7](#_Toc135133249)

[2. UML diagrams (Class, use case, activity) 8](#_Toc135133250)

[IV. Testing & Implementation 11](#_Toc135133251)

[Testing Procedures Observed 11](#_Toc135133252)

[The code analyses with screenshots of results 18](#_Toc135133254)

# II. Ability to demonstrate the requirements assessment stage

# 1. Analysing requirements

This code is an implementation of a specialized calculator program that can calculate areas of various shapes, solve quadratic equations, and perform basic statistics. The program has a graphical user interface (GUI) that is created using the Swing library in Java. The GUI consists of a main frame, which has a menu bar with three menus - Areas of selected shapes, Solve quadratic equations, and Basic Statistics. The menus contain several items that correspond to the specific functionality of the program, such as calculating the area of a rectangle or solving a quadratic equation.

The target audience for this program is anyone who needs to perform calculations related to the areas of various shapes, solve quadratic equations, or calculate basic statistics. The program's purpose is to provide an easy-to-use and efficient tool for these types of calculations. For example, a student studying geometry could use the program to quickly calculate the area of various shapes for their homework assignments.

The code has been written in a modular and reusable way, with each item in the menu having its own action listener that handles the corresponding calculation. This design makes it easier to maintain and update the program as new features or functionality can be added without affecting the existing code. The code also provides a way for the user to specify the precision of the calculated result, which adds to the usability and accuracy of the program.

Overall, the code demonstrates the requirements assessment stage of the software development process by identifying the problem context, the purpose, the target audience, the objectives, and the significance of the project. It also provides a clear and organized implementation of the program's functionality, making it easy for users to perform the desired calculations. However, there may be potential limitations or areas for improvement in the program that could be addressed in future updates.

# 2. The Activity timeline

The figure below is an Activity timeline to Design a fully functional Specialised Calculator GUI.

A screenshot of a cell phone

Description automatically generated with medium confidence

# 3. PERT Chart

Within a limited timeframe of 14 days, my objective is to develop a specialized calculator app featuring a user-friendly graphical user interface (GUI) through this 6 process. I will begin by outlining the requirements, designing a mockup, and refining the interface layout. The app will encompass functionalities such as shape area calculations, quadratic equation solving, and basic statistical computations. Rigorous testing, user input, and comprehensive documentation will contribute to the final polished version. Despite the time constraints, I am confident in delivering a robust and intuitive calculator application.

A picture containing text, screenshot, diagram, circle

Description automatically generated

# III. The design specification for the application

1. The constructs & GUI design (inputs, processes and outputs)

User inputs are obtained through the GUI application, allowing users to provide coefficients for quadratic equations, dimensions for shapes, and other relevant parameters. These inputs serve as data for the calculations performed by the code.

The code processes the user-provided coefficients using the quadratic formula to calculate the roots of the quadratic equation. Additionally, calculations are performed for various shapes like cones, cylinders, parallelograms, trapeziums, and triangles. Shape-specific formulas are applied to determine properties such as area, surface area, or volume.

The calculated roots of the quadratic equation are displayed to the user either as text within the GUI or visually represented. Similarly, the results of shape calculations, such as area, surface area, or volume, are presented to the user within the GUI components, either as text or visual representations.

In summary, the code allows users to provide inputs, performs necessary calculations based on those inputs, and presents the results in the GUI. This enables users to interact with the application, obtain the desired outputs, and visualize the performed calculations.

# The Storyboard

A notebook with writing on it

Description automatically generated with low confidence

Storyboard for specialised Calculator GUI

# 2. UML diagrams (Class, use case, activity)

A picture containing text, screenshot, design

Description automatically generated

UML Class Diagram

The UML class diagram provided depicts several classes within a Java program. The MainFrame class represents the main window of a graphical user interface (GUI) application. It encompasses various instance variables, such as JFrame, JMenuBar, JMenu, and JMenuItem objects, which are responsible for different components of the user interface. Alongside a constructor, the MainFrame class includes two methods: getRootsText() and actionPerformed(ActionEvent).

The QuadraticEquation class is depicted in the UML class diagram and represents the quadratic equation in the specialized calculator application. It includes three private instance variables: 'a', 'b', and 'c', which store the coefficients of the quadratic equation.

The class provides a public constructor, QuadraticEquation(double, double, double), which takes the values of 'a', 'b', and 'c' as parameters to initialize the instance variables. Additionally, the class features a public method, calculateRoots(), which calculates and returns the roots of the quadratic equation as an array of doubles.

The other classes in the diagram correspond to specific shapes and calculations employed in a specialized calculator application. These classes, namely Cone, Cylinder, Parallelogram, Trapezium, Triangle, and BasicStats, possess instance variables to store relevant properties of the shape or calculation. Furthermore, each class features methods that execute the respective calculations. For example, the Cone class incorporates instance variables for the radius and height of a cone, as well as a method to calculate its surface area.

Finally, The "BasicStats" class has five public methods: "calculateMean," "calculateMedian," "calculateStandardDeviation," "calculateVariance," and "calculateMode." They perform statistical calculations such as mean, median, standard deviation, variance, and mode on arrays of double numbers. These methods are accessible from other code and return double values as results.

A picture containing text, diagram, plan, technical drawing

Description automatically generated

User Flow Diagram

In addition to the UML class diagram, I have also utilized an activity diagram in the form of a user flow diagram. This activity diagram illustrates the flow of actions and decisions within the Calculator application from a user's perspective. It visually represents the sequence of steps and interactions involved in achieving specific tasks or goals. By employing the user flow diagram, I have captured the user's journey and the various paths they can take while using the application, enhancing the understanding of the application's functionality and usability.

A picture containing text, diagram, screenshot, font

Description automatically generated

Use case diagram for the specialised calculator

# IV. Testing & Implementation

## 1. Testing Procedures Observed

A screen shot of a computer

Description automatically generated with medium confidence

Figure 1

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 2

A screenshot of a computer

Description automatically generated

Figures 1-3: JUnit test result for geometric calculations in the MainFrame class

A screenshot of a computer

Description automatically generated

Figure 4

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 5

A screenshot of a computer

Description automatically generated

Figure 6

A picture containing text, software, multimedia software, computer icon

Description automatically generated

Figure 7: demonstrates the JUnit test result for quadratic equation calculation in the MainFrame class

The provided Figure 8-11 have written a collection of classes that encapsulate different geometric shapes and perform fundamental statistical calculations. These classes can be used for JUnit testing to verify the correctness of the calculations performed by the methods.

Each class has its own constructor and computation methods, and some classes also have getter and setter methods to manipulate the attributes associated with each shape. For example, the Cone class has a constructor that takes two arguments, radius and height, and a method called calculateSurfaceArea that calculates the surface area of a cone based on its radius and height.

The BasicStats class includes static methods to compute essential statistical measures such as mean, median, standard deviation, mode, and variance for a provided array of numbers. These methods can also be tested using JUnit to ensure that they produce the correct results.

# A screenshot of a computer program Description automatically generated with medium confidence

Figure 8

A screenshot of a computer

Description automatically generated with medium confidence

Figure 9

A screen shot of a computer program

Description automatically generated with medium confidence

Figure 10

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 11

# 2. The code analyses with screenshots of results

Figures 12 and 13 defines a MainFrame class that implements a GUI calculator with various functionalities. It sets up the main window, menu bar, menus, and menu items, and handles user actions through the actionPerformed method. The constructor initializes the GUI components, sets their properties, and establishes the layout of the calculator. This code forms the foundation of the GUI calculator, enabling user interaction and providing a structure for implementing additional functionalities.

A picture containing text, screenshot, software, font

Description automatically generated

Figure 12

A screenshot of a computer screen

Description automatically generated with medium confidence

Figure 13

The code from figure 14-15 demonstrates a method called actionPerformed that is responsible for handling user actions in a graphical calculator interface. It determines which menu item was selected by the user and performs the appropriate calculations based on the chosen option.

When a menu item is selected, a dialog box appears and asks the user to input the necessary values for the calculation. For example, if the user selects the "Area of Rectangle" option, they will be prompted to enter the length, width, and precision. Once the user provides valid inputs, the code uses the given formula and the entered values to perform the calculation. The result is then formatted according to the specified precision and displayed in another dialog box.

To handle any potential errors, the code includes exception handling. If the user enters non-numeric values instead of numbers, an error message is displayed to let them know that their input is invalid.

Overall, this code demonstrates how to handle user interactions in a graphical calculator. It shows how to link menu items to specific actions and gather user input using dialog boxes. The code also highlights the importance of validating user input and handling potential errors gracefully.

A screen shot of a computer program

Description automatically generated with low confidence

Figure 14

A screen shot of a computer program

Description automatically generated with low confidence

Figure 15

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 16

A screenshot of a computer error message

Description automatically generated with medium confidence

Figure 17

A screen shot of a computer screen

Description automatically generated with low confidence

Figure 18-19: prompts the user to enter the radius, height and precision for the surface area of a cone

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Figure 20

A screenshot of a calculator

Description automatically generated

Figure 21

A screenshot of a computer

Description automatically generated with medium confidence

Figure 22

A picture containing text, software, font, multimedia software

Description automatically generated

Figure 20-22: prompts the user to enter the radius and precision to calculate the area of a circle

A screenshot of a calculator

Description automatically generated

Figure 23

A screenshot of a computer

Description automatically generated with medium confidence

Figure 24

A screenshot of a computer error

Description automatically generated with medium confidence

Figure 25

A screen shot of a computer

Description automatically generated with low confidence

Figure 23-26: prompts the user to enter the base, height and precision to find the area of the triangle

A screenshot of a computer

Description automatically generated

Figure 27

A screenshot of a calculator

Description automatically generated

Figure 28

A screenshot of a computer

Description automatically generated

Figure 29

A picture containing text, software, multimedia software, font

Description automatically generated

Figure 27-30: prompts the user to enter the radius and height of the cylinder, as well as the precision for the calculation.

A screenshot of a computer error message

Description automatically generated with medium confidence

Figure 31

A screenshot of a calculator

Description automatically generated

Figure 32

A picture containing text, font, screenshot

Description automatically generated

Figure 33: 31-33 illustrates the section where the user is prompted to enter the base and height of the parallelogram, as well as the precision for the calculation.

A screenshot of a computer error

Description automatically generated with medium confidence

Figure 34

A screenshot of a computer

Description automatically generated

Figure 35

A screenshot of a calculator

Description automatically generated

Figure 36

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Figure 37: 34-37 prompts the user to enter the lengths of the two bases, the height and the precision of the calculation to calculate the area of a trapezium.

A screenshot of a computer error

Description automatically generated with medium confidence

Figure 38

A screenshot of a computer

Description automatically generated with medium confidence

Figure 39

A screenshot of a calculator

Description automatically generated

Figure 40

As far as the actionPerformed method is overridden to handle different user actions, It calculates the areas of selected shapes based on user input.

Using if and else if statements, on the other hand, identifies the action source and executes the appropriate code for each shape. Users are prompted to enter shape parameters, and the area is calculated accordingly. The calculated area is formatted and shown to the user.

Separate conditions handle quadratic equation and basic statistics calculations. These sections prompt for inputs, perform the calculations, and display the results.

The use of if and else if statements ensures that the program responds correctly to the user's chosen shape or action by executing the corresponding code.

A screen shot of a computer

Description automatically generated with medium confidence

Figure 41

Following the images of 42-44, the provided code calculates the mean, variance, standard deviation, median, and mode of the provided data. It adjusts the variance based on whether the data represents a sample or a population. The results are displayed in a message dialog with options to calculate again or close.

A screenshot of a calculator

Description automatically generated

Figure 42

A screenshot of a computer

Description automatically generated

Figure 43: Asks the user to input a sample or population for the result

A screenshot of a calculator

Description automatically generated

Figure 44: Result of Basic Statistics that calculate the Sample

A screenshot of a computer

Description automatically generated

Figure 45: Result of Basic Statistics that calculate the Population

A screen shot of a computer program

Description automatically generated with low confidence

Figure 46 code to launch the Specialized Calculator GUI