

Saiket Systems Internship tasks

Task1

Task 1: Calculator Application

Description:

Build a basic calculator application enabling arithmetic operations. Create a user-friendly menu for selecting operations, input numbers, and handle edge cases, such as division by zero, using exception handling. Strengthen your understanding of core Java concepts like methods and conditionals through this task.

Skills:

Core Java Concepts (Methods, Conditionals)

User Input Handling

Exception Handling

The screenshot shows the Microsoft Visual Studio Code (VS Code) interface. The left sidebar includes icons for Explorer, Tasks, Outline, Timeline, and Java Projects. The top navigation bar has File, Edit, Selection, and a back/forward icon. The main area displays a Java file named 'Calculator.java' with the following code:

```
public class Calculator {
    static double multiply(double a, double b) {
    }

    // Method for Division
    static double divide(double a, double b) {
        if (b == 0) {
            throw new ArithmeticException("Division by zero is not allowed.");
        }
        return a / b;
    }

    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int choice;
        double num1, num2;

        do {
            System.out.println("===== Calculator Menu =====");
            System.out.println("1. Addition");
            System.out.println("2. Subtraction");
            System.out.println("3. Multiplication");
            System.out.println("4. Division");
            System.out.println("5. Exit");
            System.out.print("Choose an operation: ");
            choice = sc.nextInt();

            if (choice >= 1 && choice <= 4) {
                System.out.print("Enter first number: ");
                num1 = sc.nextDouble();
                System.out.print("Enter second number: ");
                num2 = sc.nextDouble();

                try {
                    switch (choice) {
                        case 1:
```

The code implements a simple calculator with addition, subtraction, multiplication, and division. It also handles division by zero. The main method provides a menu and reads user input for each operation.

The screenshot shows a Java code editor interface with the following details:

- File Explorer:** Shows a file named "Calculator.java" with 1 change.
- Tasks:** Shows a task named "Calculator.java 1".
- Code Editor:** Displays the following Java code:

```
public class Calculator {
    public static void main(String[] args) {
        // ...
        switch (choice) {
            case 1:
                System.out.println("Result: " + (a + b));
                break;
            case 2:
                System.out.println("Result: " + (a - b));
                break;
            case 3:
                System.out.println("Result: " + (a * b));
                break;
            case 4:
                try {
                    if (b == 0) {
                        throw new ArithmeticException("Division by zero is not allowed");
                    }
                    System.out.println("Result: " + (a / b));
                } catch (ArithmeticException e) {
                    System.out.println("Error: " + e.getMessage());
                }
                break;
            default:
        }
    }
}
```
- Terminal:** Shows the command line output of the application:

```
PS C:\Users\Aaryan\Desktop\SaiKet Systems Internship\Tasks> cd "c:\Users\Aaryan\Desktop\SaiKet Systems Internship\Tasks" ; if ($?) { javac Calculator.java } ; if ($?) { java Calculator }
1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice: 1
Enter first number: 12345
Enter second number: 100
Result: 12445.0
PS C:\Users\Aaryan\Desktop\SaiKet Systems Internship\Tasks>
```
- Suggested Actions:** Includes "Build Workspace" and "Show Config".
- Bottom Status Bar:** Shows "ln 57, Col 1" and "Java".

Task 2: Temperature Converter

Description:

Create a temperature converter that handles Celsius to Fahrenheit and vice versa. Allow users to choose input and output scales. Implement conversion formulas and provide clear results. This task reinforces your Java programming skills and mathematical operation implementation.

Skills:

Core Java Concepts (Methods, User Input)

Mathematical Operations

A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows the Explorer, Tasks, Outline, Timeline, and Java Projects sections. The center pane displays Java code for a Temperature Converter application. The code uses a Scanner to get user input and an if-else block to perform Celsius to Fahrenheit or Fahrenheit to Celsius conversion. The right sidebar includes a 'Build with Agent' section and a 'SUGGESTED ACTIONS' panel with 'Build Workspace' and 'Show Config' buttons.

```
1 import java.util.Scanner;
2
3 public class TemperatureConverter {
4
5     Run | Debug
6     public static void main(String[] args) {
7
8         Scanner sc = new Scanner(System.in);
9         int choice;
10        double temperature, result;
11
12        System.out.println("==> Temperature Converter ==>");
13        System.out.println("1. Celsius to Fahrenheit");
14        System.out.println("2. Fahrenheit to Celsius");
15        System.out.print("Choose conversion type: ");
16
17        choice = sc.nextInt();
18
19        if (choice == 1) {
20            System.out.print("Enter temperature in Celsius: ");
21            temperature = sc.nextDouble();
22
23            result = (temperature * 9 / 5) + 32;
24            System.out.println("Result: " + result + " °F");
25
26        } else if (choice == 2) {
27            System.out.print("Enter temperature in Fahrenheit: ");
28            temperature = sc.nextDouble();
29
30            result = (temperature - 32) * 5 / 9;
31            System.out.println("Result: " + result + " °C");
32
33        } else {
34            System.out.println("Invalid choice!");
35        }
36
37        sc.close();
38    }
39}
```

A screenshot of the Visual Studio Code (VS Code) interface, similar to the first one but with a terminal window at the bottom. The terminal shows the execution of the Java code. It starts with the command 'PS C:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks> cd "c:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks"' followed by 'javac TemperatureConverter.java'. Then it shows the output of the program, which asks for a choice (1 for Celsius to Fahrenheit) and prints the result (640.888888888889). The right sidebar includes a 'Build with Agent' section and a 'SUGGESTED ACTIONS' panel with 'Build Workspace' and 'Show Config' buttons.

```
PS C:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks> cd "c:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks"
PS C:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks> javac TemperatureConverter.java
PS C:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks> java TemperatureConverter
==> Temperature Converter ==>
1. Celsius to Fahrenheit
2. Fahrenheit to Celsius
Choose conversion type: 2
Enter temperature in Fahrenheit: 1200
Result: 640.888888888889
```

Task3

Task 3: To-Do List Application

Description:

Develop a to-do list application where users can manage tasks. Use classes to represent tasks, incorporating attributes like title and status. Implement functions for adding tasks, marking them as complete, and displaying the task list.

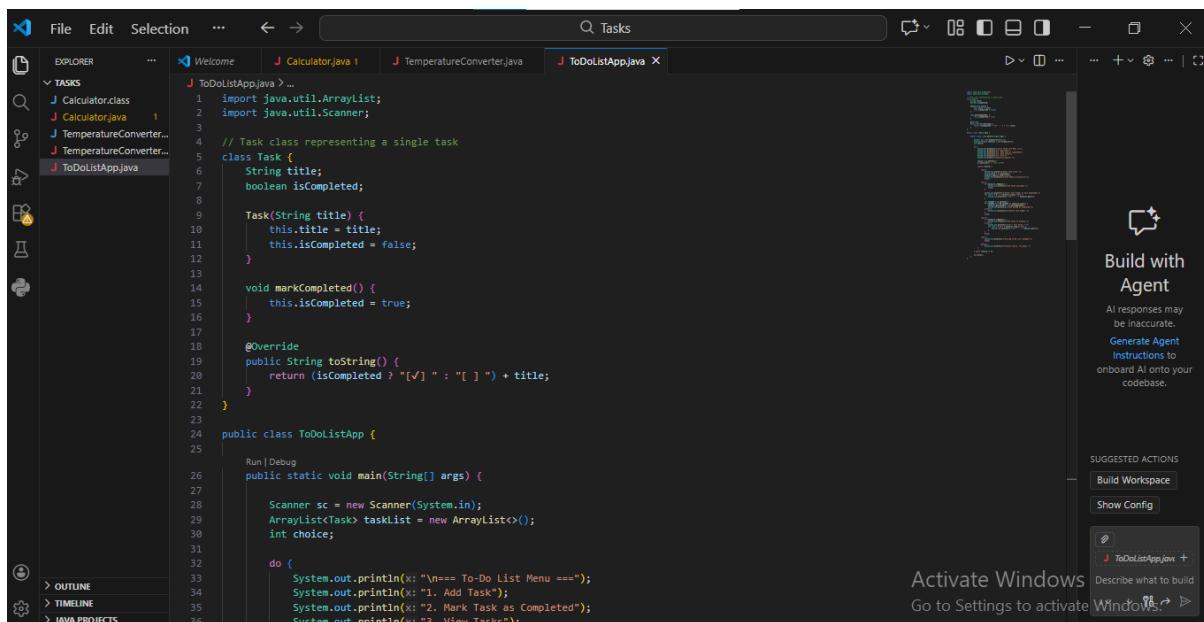
Enhance your knowledge of object-oriented programming and data structure manipulation.

Skills :

Object-Oriented Programming (Classes, Objects)

Data Structures (Lists)

Conditional Statements



The screenshot shows a Java IDE interface with the following details:

- File Bar:** File, Edit, Selection, ...
- Toolbars:** Standard toolbar with icons for file operations.
- Left Sidebar:** EXPLORER (Tasks), OUTLINE, TIMELINE, JAVA PROJECTS.
- Central Area:** Code editor for `ToDoListApp.java`. The code defines a `Task` class with attributes `title` and `isCompleted`, and methods `markCompleted` and `toString`. It also defines a `ToDoListApp` class with a `main` method that prints a menu and handles user input.
- Right Sidebar:**
 - Build with Agent:** A panel with instructions to onboard AI onto the codebase.
 - SUGGESTED ACTIONS:** Buttons for "Build Workspace" and "Show Config".
 - Activate Windows:** A message prompting to activate Windows.

The screenshot shows a Java IDE interface with the following details:

- File Bar:** File, Edit, Selection, ...
- Toolbar:** Standard icons for file operations.
- Explorer:** Shows files: Welcome, Calculator.java, TemperatureConverter.java, and ToDoListApp.java.
- Code Editor:** Displays the ToDoListApp.java code. The code implements a simple command-line application for managing a to-do list. It includes a switch statement for user choices, task addition, marking tasks as completed, displaying tasks, and exiting the application.
- Right Panel:**
 - Build with Agent:** A sidebar with AI-related features.
 - Suggested Actions:** Buttons for "Build Workspace" and "Show Config".
 - A tooltip for "Activate Windows" with the message: "Go to Settings to activate Windows".
- Bottom Status Bar:** Shows the current line (Ln 96), column (Col 1), spaces (Spaces: 4), and encoding (UTF-8). It also includes Java and Go Live buttons.

The screenshot shows the same Java IDE interface, but the ToDoListApp.java code has been modified to include a loop for handling multiple user inputs. The changes are as follows:

```
24 public class ToDoListApp {  
25     public static void main(String[] args) {  
26         // ... existing code ...  
27         > while (choice != 4);  
28         sc.close();  
29     }  
30 }
```

The rest of the code remains the same as in the first screenshot, implementing the to-do list functionality.

```
PS C:\Users\Varyan\Desktop\Saliket Systems Internship\Tasks> cd "c:\Users\Varyan\Desktop\Saliket Systems Internship\Tasks\" & if ($?) { javac ToDoListApp.java } & if ($?) { java ToDoListApp }

== To-Do List Menu ==
1. Add Task
2. Mark Task as Completed
3. View Tasks
4. Exit
Choose an option: 1
Enter task title: Play
Task added successfully!

== To-Do List Menu ==
1. Add Task
2. Mark Task as Completed
3. View Tasks
4. Exit
Choose an option: 2
Select task number to mark completed:
1. [ ] Play
```

Task4

Task 4: Number Guessing Game Description: Create a number-guessing game where users guess a random number. Provide feedback for high or low guesses and track attempts. Implement loops for multiple attempts and reinforce your skills in handling random numbers and conditional statements.

Skills :

Core Java Concepts (Random Numbers, Loops) User Input Handling Conditional Statements

```

File Edit Selection ... < > Tasks
EXPLORER ... Welcome J Calculator.java J TemperatureConverter.java J ToDoListApp.java J NumberGuessingGame.java
J Calculator.class J Calculator.java 1 J NumberGuessingGam... J NumberGuessingGam...
J Taskclass J TemperatureConverter... J TemperatureConverter... J ToDoListApp.class J ToDoListApp.java
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
SUGGESTED ACTIONS
Build Workspace Show Config
J NumberGuessing +

```

```

public class NumberGuessingGame {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Random random = new Random();

        int randomNumber = random.nextInt(bound: 100) + 1; // 1 to 100
        int guess;
        int attempts = 0;

        System.out.println("Welcome to the Number Guessing Game!");
        System.out.println("Guess a number between 1 and 100.");

        do {
            System.out.print("Enter your guess: ");
            guess = sc.nextInt();
            attempts++;

            if (guess > randomNumber) {
                System.out.println(" Too high! Try again.");
            } else if (guess < randomNumber) {
                System.out.println(" Too low! Try again.");
            } else {
                System.out.println(" Congratulations! You guessed the correct number.");
                System.out.println(" Total attempts: " + attempts);
            }
        } while (guess != randomNumber);

        sc.close();
    }
}

```



```

File Edit Selection ... < > Tasks
EXPLORER ... Welcome J Calculator.java J TemperatureConverter.java J ToDoListApp.java J NumberGuessingGame.java
J Calculator.class J Calculator.java 1 J NumberGuessingGam... J NumberGuessingGam...
J Taskclass J TemperatureConverter... J TemperatureConverter... J ToDoListApp.class J ToDoListApp.java
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
SUGGESTED ACTIONS
Build Workspace Show Config
J NumberGuessing +

```

```

PS C:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks> cd "c:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks"
PS C:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks> if ($?) { java NumberGuessingGame }
? Too low! Try again.
Enter your guess: 98
? Too low! Try again.
Enter your guess: 100
? Too high! Try again.
Enter your guess: 98
? Too high! Try again.
Enter your guess: 95
? Congratulations! You guessed the correct number.
? Total attempts: 5
PS C:\Users\Aaryan\Desktop\SaiKit Systems Internship\Tasks>

```

Activate Windows
Go to Settings to activate Windows.

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Code + | □ | ×

ln 36, Col 1 Spaces: 4 UFT-8 CRLF {} Java Go Live

Task5

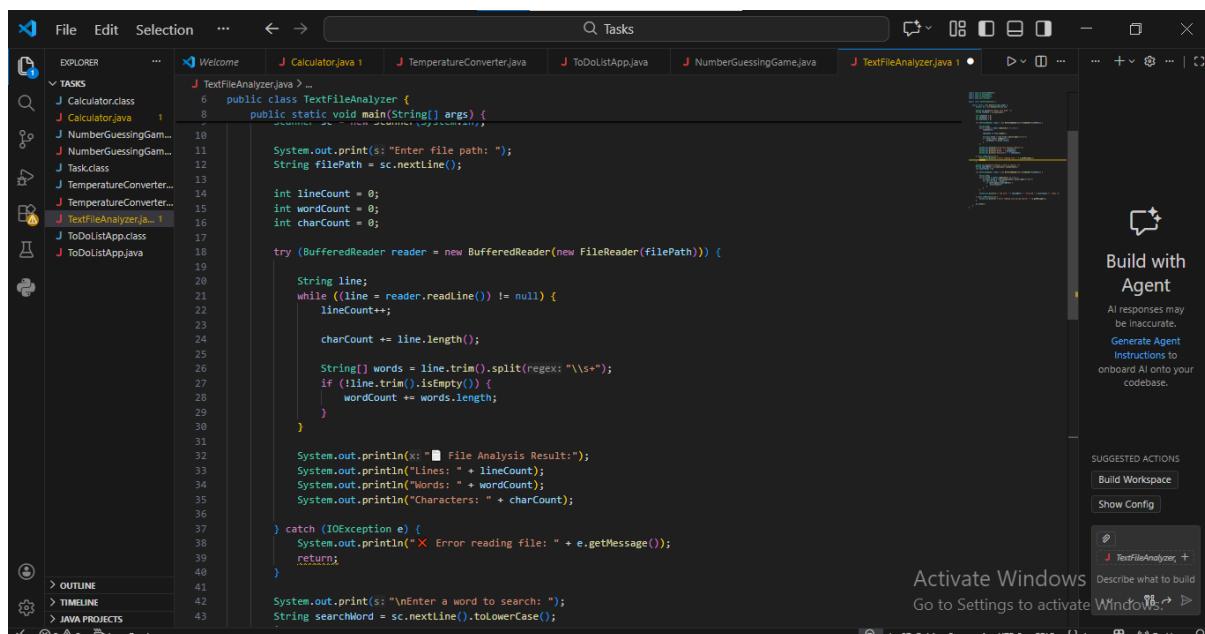
Task 5:

Text File Analyzer Write a program to analyze a text file, counting words, lines, and characters. Implement

functionality to search for specific words. Utilize file I/O, string manipulation, and exception handling. This task strengthens your file-handling skills and text-processing abilities in Java.

Skills :

File Input/Output in Java String Manipulation Exception Handling

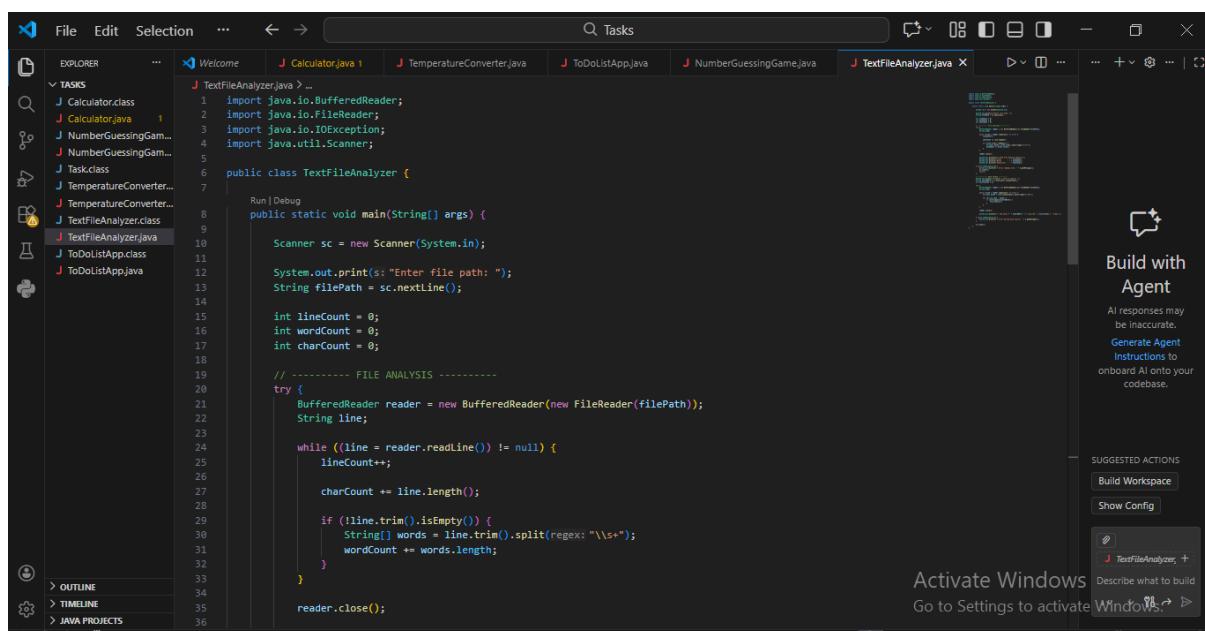


```

File   Edit  Selection  ...  ← →  Tasks
EXPLORER ... Welcome J Calculator.java  J TemperatureConverter.java  J ToDoListApp.java  J NumberGuessingGame.java  J TextFileAnalyzer.java  ...
J Calculator.class
J Calculator.java  1
J NumberGuessingGam...
J NumberGuessingGam...
J Task.class
J TemperatureConverter...
J TemperatureConverter...
J TextFileAnalyzer.java  1
J ToDoListApp.class
J ToDoListApp.java

6  public class TextFileAnalyzer {
7      public static void main(String[] args) {
8          Scanner sc = new Scanner(System.in);
9
10         System.out.print("Enter file path: ");
11         String filePath = sc.nextLine();
12
13         int lineCount = 0;
14         int wordCount = 0;
15         int charCount = 0;
16
17         try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {
18
19             String line;
20             while ((line = reader.readLine()) != null) {
21                 lineCount++;
22
23                 charCount += line.length();
24
25                 String[] words = line.trim().split(regex: "\\s+");
26                 if (!line.trim().isEmpty()) {
27                     wordCount += words.length;
28                 }
29
30             }
31
32             System.out.println("File Analysis Result:");
33             System.out.println("Lines: " + lineCount);
34             System.out.println("Words: " + wordCount);
35             System.out.println("Characters: " + charCount);
36
37         } catch (IOException e) {
38             System.out.println("Error reading file: " + e.getMessage());
39             return;
40         }
41
42         System.out.print("\nEnter a word to search: ");
43         String searchWord = sc.nextLine();
44     }
}

```



```

File   Edit  Selection  ...  ← →  Tasks
EXPLORER ... Welcome J Calculator.java  J TemperatureConverter.java  J ToDoListApp.java  J NumberGuessingGame.java  J TextFileAnalyzer.java  ...
J Calculator.class
J Calculator.java  1
J NumberGuessingGam...
J NumberGuessingGam...
J Task.class
J TemperatureConverter...
J TemperatureConverter...
J TextFileAnalyzer.class
J TextFileAnalyzer.java  1
J ToDoListApp.class
J ToDoListApp.java

1  import java.io.BufferedReader;
2  import java.io.FileReader;
3  import java.io.IOException;
4  import java.util.Scanner;
5
6  public class TextFileAnalyzer {
7
8      Run | Debug
9      public static void main(String[] args) {
10
11         Scanner sc = new Scanner(System.in);
12
13         System.out.print("Enter file path: ");
14         String filePath = sc.nextLine();
15
16         int lineCount = 0;
17         int wordCount = 0;
18         int charCount = 0;
19
20         // ----- FILE ANALYSIS -----
21         try {
22             BufferedReader reader = new BufferedReader(new FileReader(filePath));
23             String line;
24
25             while ((line = reader.readLine()) != null) {
26                 lineCount++;
27
28                 charCount += line.length();
29
30                 if (!line.trim().isEmpty()) {
31                     String[] words = line.trim().split(regex: "\\s+");
32                     wordCount += words.length;
33                 }
34
35             }
36
37             reader.close();
38
39         } catch (IOException e) {
40             System.out.println("Error reading file: " + e.getMessage());
41             return;
42         }
43
44         System.out.print("Enter a word to search: ");
45         String searchWord = sc.nextLine();
46     }
}

```

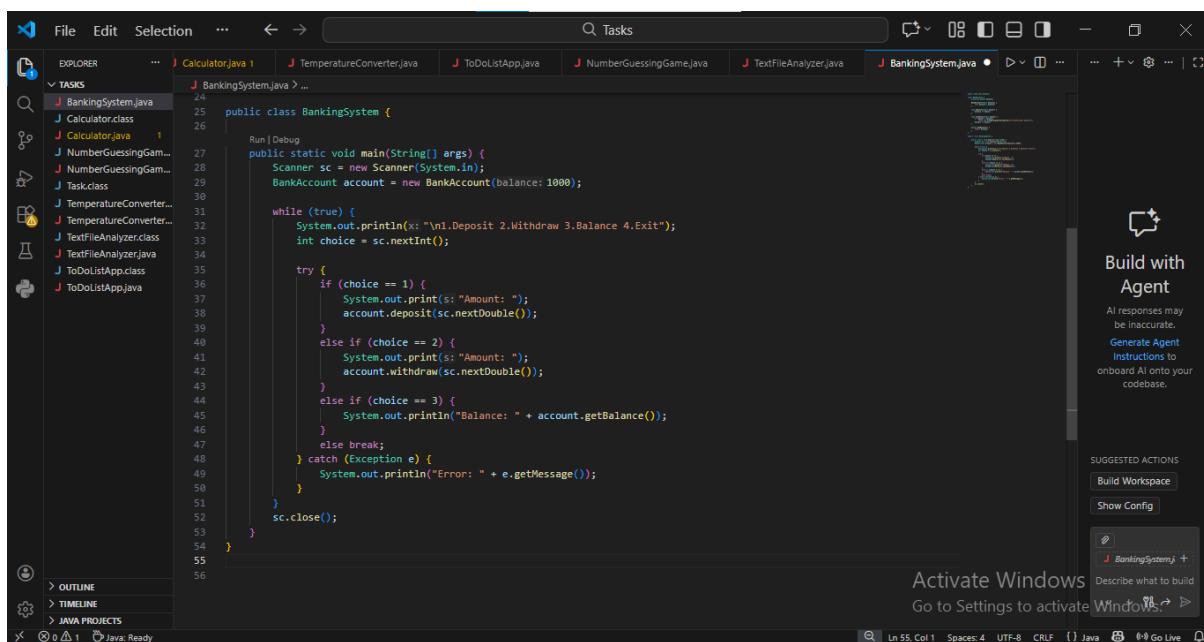
Task6

Task 6:

Basic Banking System Description: Develop a basic banking system where users can manage accounts, perform transactions, and view balances. Utilize classes for representing bank accounts and handle operations like deposit, withdrawal, and transaction history. This task enhances your understanding of classes, inheritance, and exception handling.

Skills: -

Object-Oriented Inheritance) User Input Handling
Exception Handling Programming (Classes, Inheritance)
User Input Handling Exception Handling



The screenshot shows a Java IDE interface with the following details:

- File Bar:** File, Edit, Selection, ...
- Toolbar:** Standard icons for file operations.
- Explorer View:** Shows multiple Java files: Calculator.java, TemperatureConverter.java, ToDoListApp.java, NumberGuessingGame.java, TextFileAnalyzer.java, and BankingSystem.java (which is currently selected).
- Code Editor:** Displays the source code for BankingSystem.java. The code implements a simple banking system with deposit, withdrawal, and balance checking functionality using a BankAccount object and a Scanner for user input.
- Right Sidebar:**
 - Build with Agent:** A section with a warning message: "AI responses may be inaccurate. Generate Agent Instructions to onboard AI to your codebase."
 - Suggested Actions:** Buttons for "Build Workspace" and "Show Config".
 - A tooltip for the BankingSystem.java file: "Activate Windows Go to Settings to activate Windows".
- Bottom Status Bar:** Shows "Ln 55, Col 1 Spaces: 4 UTF-8 CR LF {} Java Go Live".

The screenshot shows a Java development environment with the following details:

- Explorer View:** Shows various Java files and classes, with `BankingSystem.java` currently selected.
- Code Editor:** Displays the `BankingSystem.java` file content. The code defines a `BankingSystem` class with a static inner class `BankAccount`. It includes methods for depositing and withdrawing funds, with validation logic for non-negative amounts and sufficient balance.
- Terminal:** Shows the command-line output of running the `BankingSystem` class. The user enters choice 1 (Deposit) and specifies a deposit amount of 1000000, which is successful.
- Status Bar:** Includes links to "Activate" and "Go to Settings".

```
BankingSystem.java
public class BankingSystem {
    static class BankAccount {
        public BankAccount(double initialBalance) {
        }
        public void deposit(double amount) {
            if (amount <= 0) {
                throw new IllegalArgumentException("Deposit amount must be positive");
            }
            balance += amount;
            System.out.println("Deposit successful.");
        }
        public void withdraw(double amount) {
            if (amount <= 0) {
                throw new IllegalArgumentException("Withdrawal amount must be positive");
            }
            if (amount > balance) {
                throw new IllegalArgumentException("Insufficient balance");
            }
        }
    }
}

PS C:\Users\Aaryan\Desktop\SaiKet Systems Internship\Tasks> cd "c:\Users\Aaryan\Desktop\SaiKet Systems Internship\Tasks\" ; if ($?) { java BankingSystem }
2. Withdraw
3. Check Balance
4. Exit
Enter your choice: 1
Enter deposit amount: 1000000
Deposit successful.

===== BASIC BANKING SYSTEM =====
1. Deposit
2. Withdraw
3. Check Balance
4. Exit
Enter your choice: 2
```