

Assignment -11 (24-09-2025)

Roll No - 107

Problem 1: Create Customer class with the relevant information like name, address, id, phone

Java Assignment 8

EXPLORER

JAVA ASSIGNMENT 8

- Animal.class
- AnimalQ2.java
- AnonymousQ9.class
- AnonymousQ9.java
- ApplianceQ3.java
- BankingSystem.java
- Cat.class
- Dog.class
- DogCatQ4.class
- DogCatQ4.java
- EngineQ5.java
- GreetingQ8.java
- ImplementationQ7.java
- InterfaceQ6.java
- Main.java
- PrinterQ10.java
- Shape.class
- ShapeCreator.class
- ShapeCreator\$1.class

BankingSystem.java

```
182 public class BankingSystem {  
183     public static void main(String[] args) {  
184         Customer customer4 = new Customer(name:"Bob Johnson", address:"789 Pine Rd, Uptown", customerId:"CUST004", phone:  
185             System.out.println(x:"Test Case 4: Customer without Account");  
186             customer4.disp();  
187             System.out.println();  
188         System.out.println(x:"Test Case 5: Testing Getters and Setters");  
189         Account account5 = new Account();  
190         account5.setAccountType(accountType:"Current");  
191     }  
192 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Java Assignment 8_978aled6\bin' 'BankingSystem'
Banking System Demo

Test Case 1: Normal Customer with Positive Balance
--- Customer Information ---
Customer Name: John Doe
Customer ID: CUST001
Address: 123 Main St, Anytown
Phone Number: +1-555-0123

Account Details:
Account Type: Savings
Account Number: SAV001
Current Balance: \$5000.0
Minimum Balance: \$1000.0
Annual Interest: \$250.00

=====
Test Case 2: Customer with Negative Balance
--- Customer Information ---
Customer Name: Jane Smith
Customer ID: CUST002
Address: 456 Oak Ave, Downtown
Phone Number: +1-555-0456

Account Details:
Account Type: Checking

Ln 202, Col 9 Spaces: 4 UTF-8 CRLF ↴ Java ⚡ Signed out ⚡ Go Live ⚡ Prettier ⚡ 3:25 PM 9/25/2025

Java Assignment 8

EXPLORER

JAVA ASSIGNMENT 8

- Animal.class
- AnimalQ2.java
- AnonymousQ9.class
- AnonymousQ9.java
- ApplianceQ3.java
- BankingSystem.java
- Cat.class
- Dog.class
- DogCatQ4.class
- DogCatQ4.java
- EngineQ5.java
- GreetingQ8.java
- ImplementationQ7.java
- InterfaceQ6.java
- Main.java
- PrinterQ10.java
- Shape.class
- ShapeCreator.class
- ShapeCreator\$1.class

BankingSystem.java

```
182 public class BankingSystem {  
183     public static void main(String[] args) {  
184         Customer customer4 = new Customer(name:"Bob Johnson", address:"789 Pine Rd, Uptown", customerId:"CUST004", phone:  
185             System.out.println(x:"Test Case 4: Customer without Account");  
186             customer4.disp();  
187             System.out.println();  
188         System.out.println(x:"Test Case 5: Testing Getters and Setters");  
189         Account account5 = new Account();  
190         account5.setAccountType(accountType:"Current");  
191     }  
192 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Account Details:
Account Type: Checking
Account Number: CHK002
Current Balance: \$-500.0
Minimum Balance: \$0.0
Interest Calculation Error: Cannot calculate interest on negative balance: -500.0

=====
Test Case 3: Direct Exception Handling
Exception caught: Cannot calculate interest on negative balance: -1000.0

Test Case 4: Customer without Account
--- Customer Information ---
Customer Name: Bob Johnson
Customer ID: CUST004
Address: 789 Pine Rd, Uptown
Phone Number: +1-555-0789

No account associated with this customer.

=====
Test Case 5: Testing Getters and Setters
--- Customer Information ---
Customer Name: Alice Wilson
Customer ID: CUST005
Address: 321 Elm St, Suburb
Phone Number: +1-555-0321

Ln 202, Col 9 Spaces: 4 UTF-8 CRLF ⚡ Signed out ⚡ Go Live ⚡ Prettier ⚡ 3:25 PM 9/25/2025

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

- File Explorer:** Shows a project named "JAVA ASSIGNMENT 8" containing files like Animal.class, AnimalQ2.java, AnonymousQ9.class, AnonymousQ9.java, ApplianceQ3.java, BankingSystem.java, Cat.class, Dog.class, DogCatQ4.class, DogCatQ4.java, EngineQ5.java, GreetingQ8.java, ImplementationQ7.java, InterfaceQ6.java, Main.java, PrinterQ10.java, Shape.class, ShapeCreator.class, and ShapeCreator\$1.class.
- Code Editor:** Displays the content of `BankingSystem.java`. The code includes sections for test cases 4 and 5, involving Customer and Account objects.
- Terminal:** Shows the output of the program execution, detailing customer and account information for two test cases.
- Bottom Status Bar:** Includes system status like weather (Rainy days ahead), temperature (26°C), and system icons.

Java IO:

1. Write a Program to read the same program file and find the no. of lines, words and characters. Write the result in in to a text file (result.txt)

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure for "java assignment 9".
- Code Editor:** Displays the `FileAnalyzer.java` file containing Java code for analyzing files.
- Outline View:** Shows the class `FileAnalyzer` with its methods: `lineCount`, `wordCount`, `characterCount`, `sourceFileName`, `resultFileName`, `mainString()`, `readAndAnalyzeFile()`, `countWordsInLine(String)`, `writeResultsToFile()`, and `displayResults()`.
- Console Output:**

```

File Analyzer Program
=====
Results successfully written to: result.txt

ANALYSIS RESULTS:
-----
File analyzed: FileAnalyzer.java
Lines: 161
Words: 575
Characters: 6108

Average words per line: 3.57
Average characters per line: 37.94
Average characters per word: 10.62

Check 'result.txt' for detailed results.
    
```
- System Tray:** Shows weather information: "Heavier rain In 41 minutes".
- Taskbar:** Shows various application icons and the system clock indicating 3:34 PM on 9/25/2025.

2. Write a program to read the same program file and write it to other file with the lines number added before each line, starting from 1.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure for "java assignment 9".
- Code Editor:** Displays the `LineNumberAdder.java` file containing Java code for adding line numbers to a file.
- Outline View:** Shows the class `LineNumberAdder` with its methods: `SOURCE_FILE`, `OUTPUT_FILE`, `mainString()`, `addLineNumbers()`, `writeHeader(PrintWriter)`, `writeFooter(PrintWriter, int)`, `closeResources(BufferedReader, PrintWriter)`, and `addLineNumbersModem()`.
- Console Output:**

```

Processed 9 lines...
Processed 19 lines...
Processed 29 lines...
Processed 39 lines...
Processed 49 lines...
Processed 59 lines...
Processed 69 lines...
Processed 79 lines...
Processed 89 lines...
Processed 99 lines...
Processed 109 lines...
Processed 119 lines...
Processed 129 lines...
Processed 139 lines...
Processed 149 lines...
Processed 159 lines...
Processed 169 lines...
Total lines processed: 169
SUCCESS: Line numbers added successfully!
Check the file 'LineNumberProgram.txt' for the result.
    
```
- System Tray:** Shows weather information: "26°C Light rain".
- Taskbar:** Shows various application icons and the system clock indicating 3:44 PM on 9/25/2025.

3. Write a Java program to read first 3 lines from a file.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project named "JAVA ASSIGNMENT 8" containing various Java files like Animal.class, AnimalQ2.java, AnonymousQ9.class, AnonymousQ9.java, ApplianceQ3.java, BankingSystem.java, Catclass, Dog.class, DogCatQ4.class, DogCatQ4.java, EngineQ5.java, GreetingQ8.java, ImplementationQ7.java, InterfaceQ6.java, Main.java, PrinterQ10.java, ReadfirstThreeLines.java, sample.txt, Shape.class, ShapeCreator.class, and ShapeCreator\$1.class.
- Code Editor:** Displays the content of the "ReadfirstThreeLines.java" file. The code uses a Scanner to read the first three lines of a file named "sample.txt". It includes error handling for FileNotFoundException.
- Terminal:** Shows the output of the program, which reads the first three lines of "sample.txt":

```
First 3 lines using BufferedReader:  
Line 1: This is the first line of the sample file.  
Line 2: Here comes the second line with some text.  
Line 3: The third line contains more sample content.
```

Method 2: Using Files.lines() - Stream API
- Bottom Status Bar:** Shows the current file is "Java Ready", system notifications (Heavy rain Today), and system status (ENG IN, 4:59 PM, 9/25/2025).

4. Write a Java program to find the longest word in a text file

Java Assignment 8

File Edit Selection View Go Run ...

EXPLORER

JAVA ASSIGNMENT 8

- Animal.class
- AnimalQ2.java
- AnonymousQ9.class
- AnonymousQ9.java
- ApplianceQ3.java
- BankingSystem.java
- Cat.class
- Dog.class
- DogCatQ4.class
- DogCatQ4.java
- EngineQ5.java
- GreetingQ8.java
- ImplementationQ7.java
- InterfaceQ6.java
- LongestWordFinder.java
- Main.java
- PrinterQ10.java
- ReadFirstThreeLines.java
- sample_text.txt
- sample.txt
- Shape.class
- ShapeCreator.class
- ShapeCreator\$1.class

LongestWordFinder.java > LongestWordFinder

```
public class LongestWordFinder {
    private static void createSampleFileIfNotExists() {
        try (PrintWriter writer = new PrintWriter(new FileWriter(DEFAULT_FILE))) {
            writer.println("The quick brown fox jumps over the lazy dog.");
            writer.println("Java programming is extraordinarily powerful and versatile.");
            writer.println("Supercalifragilisticexpialidocious is a very long word!");
            writer.println("Short words: a, an, the, is, in, on, at, to, of, for.");
            writer.println("Programming languages include: Java, Python, JavaScript, C++.");
            writer.println("This file contains various words of different lengths.");
            writer.println("Some technical terms: algorithm, implementation, optimization.");
            writer.println("Antidisestablishmentarianism is another extremely long word.");
        }
        System.out.println("Sample file '" + DEFAULT_FILE + "' created for demonstration.");
        System.out.println();
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Sample file 'sample_text.txt' created for demonstration.

Analyzing default file: sample_text.txt

Method 1: Using BufferedReader

Longest word: "supercalifragilisticexpialidocious"
Length: 34 characters
Total words analyzed: 61

Method 2: Using Stream API

Longest word: "supercalifragilisticexpialidocious"
Length: 34 characters

Method 3: Using Scanner

Longest word: "supercalifragilisticexpialidocious"
Length: 34 characters
Total words analyzed: 61

Method 4: Find All Longest Words

Maximum word length: 34 characters

powershell

Run: LongestWordFinder

Ln 288, Col 5 Spaces: 4 UTF-8 CRLF Java Signed out Go Live Prettier

7 2 8 Java: Ready

Top Stories Ba***ds of Boly...

Search

3:00 PM 9/26/2025

Java Assignment 8

File Edit Selection View Go Run ...

EXPLORER

JAVA ASSIGNMENT 8

- Animal.class
- AnimalQ2.java
- AnonymousQ9.class
- AnonymousQ9.java
- ApplianceQ3.java
- BankingSystem.java
- Cat.class
- Dog.class
- DogCatQ4.class
- DogCatQ4.java
- EngineQ5.java
- GreetingQ8.java
- ImplementationQ7.java
- InterfaceQ6.java
- LongestWordFinder.java
- Main.java
- PrinterQ10.java
- ReadFirstThreeLines.java
- sample_text.txt
- sample.txt
- Shape.class
- ShapeCreator.class
- ShapeCreator\$1.class

LongestWordFinder.java > LongestWordFinder

```
public class LongestWordFinder {
    private static void createSampleFileIfNotExists() {
        try (PrintWriter writer = new PrintWriter(new FileWriter(DEFAULT_FILE))) {
            writer.println("The quick brown fox jumps over the lazy dog.");
            writer.println("Java programming is extraordinarily powerful and versatile.");
            writer.println("Supercalifragilisticexpialidocious is a very long word!");
            writer.println("Short words: a, an, the, is, in, on, at, to, of, for.");
            writer.println("Programming languages include: Java, Python, JavaScript, C++.");
            writer.println("This file contains various words of different lengths.");
            writer.println("Some technical terms: algorithm, implementation, optimization.");
            writer.println("Antidisestablishmentarianism is another extremely long word.");
        }
        System.out.println("Sample file '" + DEFAULT_FILE + "' created for demonstration.");
        System.out.println();
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Total words analyzed: 61

Method 4: Find All Longest Words

Maximum word length: 34 characters
Longest word: "supercalifragilisticexpialidocious"

Method 5: Detailed Word Analysis

== DETAILED WORD ANALYSIS ==
Total words: 61
Unique words: 49
Shortest word: "a" (1 chars)
Longest word: "supercalifragilisticexpialidocious" (34 chars)
Average word length: 6.03 characters

== WORD LENGTH DISTRIBUTION ==

| Word Length | Count | Percentage |
|-------------|----------|------------|
| 1 chars: | 3 words | (4.9%) |
| 2 chars: | 11 words | (18.0%) |
| 3 chars: | 7 words | (11.5%) |
| 4 chars: | 12 words | (19.7%) |
| 5 chars: | 7 words | (11.5%) |
| 6 chars: | 1 words | (1.6%) |
| 7 chars: | 4 words | (6.6%) |

powershell

Run: LongestWordFinder

Ln 288, Col 5 Spaces: 4 UTF-8 CRLF Java Signed out Go Live Prettier

7 2 8 Java: Ready

26°C Cloudy

Search

3:00 PM 9/26/2025

The screenshot shows a Java IDE interface with the title "Java Assignment 8". The left sidebar displays a project structure under "JAVA ASSIGNMENT 8" containing various Java files like Animal.class, AnimalQ2.java, AnonymousQ9.java, etc. The main editor window shows a Java class named "LongestWordFinder". The code prints a sample file to the console and then analyzes it to find the longest word and its distribution. The output window shows the results:

```
Longest word: "supercalifragilisticexpialidocious" (34 chars)
Average word length: 6.03 characters

*** WORD LENGTH DISTRIBUTION ***
1 chars: 3 words (4.9%)
2 chars: 11 words (18.0%)
3 chars: 7 words (11.5%)
4 chars: 12 words (19.7%)
5 chars: 7 words (11.5%)
6 chars: 1 words (1.6%)
7 chars: 4 words (6.6%)
8 chars: 2 words (3.3%)
9 chars: 6 words (9.8%)
10 chars: 1 words (1.6%)
11 chars: 2 words (3.3%)
12 chars: 1 words (1.6%)
14 chars: 1 words (1.6%)
15 chars: 1 words (1.6%)
28 chars: 1 words (1.6%)
34 chars: 1 words (1.6%)
```

5. Write a programs to implemnt Caeser cipher using files.

Write to the file (enc_message.txt) with using caeser cipher with

the displacement value = 3.

Read the file (enc_message.txt) and decode the Cipher text and write it into

a file (dec_message.txt)

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Explorer:** Shows a project named "JAVA ASSIGNMENT 8" containing files like Animal.class, AnimalQ2.java, AnonymousQ9.class, ApplianceQ3.java, BankingSystem.java, CaeserCipher.java, Cat.class, dec_message.txt, Dog.class, DogCatQ4.class, DogCatQ4.java, enc_message.txt, EngineQ5.java, GreetingQ8.java, ImplementationQ7.java, InterfaceQ6.java, LongestWordFinder.java, Main.java, PrinterQ10.java, ReadFirstThreeLines.java, sample_text.txt, sample.txt, Shape.class, ShapeCreator.class, and ShapeCreator\$1.class.
- Code Editor:** The active file is "CaeserCipher.java". The code implements a Caesar cipher with a displacement of 3. It prints the original message and the encrypted message to the console.
- Terminal:** The terminal output shows the execution of the Java program and the resulting encrypted and decrypted messages.
- Status Bar:** Shows the current weather as 26°C Cloudy, the date as 9/26/2025, and the time as 3:43 PM.

6. Write a program to find unique words in file

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Explorer:** Shows a project named "JAVA ASSIGNMENT 8" containing files like Animal.class, AnimalQ2.java, AnonymousQ9.class, AnonymousQ9.java, ApplianceQ3.java, BankingSystem.java, CaeserCipher.java, Cat.class, dec_message.txt, Dog.class, DogCatQ4.class, DogCatQ4.java, enc_message.txt, EngineQ5.java, GreetingQ8.java, ImplementationQ7.java, InterfaceQ6.java, LongestWordFinder.java, Main.java, PrinterQ10.java, ReadFirstThreeLines.java, sample_text.txt, sample.txt, Shape.class, ShapeCreator.class, ShapeCreator\$1.class, unique_words.txt, and UniqueWordsFinder.java.
- Code Editor:** The active file is "UniqueWordsFinder.java". The code creates a sample file "sample.txt" and then reads it to find unique words.
- Terminal:** The terminal output shows the creation of the sample file and the analysis of the file to find unique words.
- Status Bar:** Shows the current weather as Rain warning In effect, the date as 9/26/2025, and the time as 3:47 PM.

7. Write a program to find duplicate words in a file

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

- File Explorer:** Shows various Java files and classes under "JAVA ASSIGNMENT 8".
- Active File:** `DuplicateWordsFinder.java` (Line 133)
- Code:** A snippet of Java code for creating a sample file containing a story about a fox and a dog.
- Output:** Results of running the program on a file named "sample.txt".
 - Total words: 57
 - Unique words: 24
 - Duplicate words: 12
- Detailed Output:** A list of duplicate words and their frequencies.

| Word | Frequency |
|-------------|-------------------|
| the | (appears 9 times) |
| and | (appears 5 times) |
| dog | (appears 4 times) |
| fox | (appears 4 times) |
| is | (appears 4 times) |
| lazy | (appears 4 times) |
| quick | (appears 4 times) |
| java | (appears 3 times) |
| brown | (appears 2 times) |
| fun | (appears 2 times) |
| programming | (appears 2 times) |
| was | (appears 2 times) |
- Terminal:** Shows the command used to run the program: PS D:\CDAC HYD\JAVA\Java Assignment >
- Bottom Bar:** Includes a weather icon (26°C, Cloudy), system icons, and a status bar showing the date and time (9/26/2025, 3:51 PM).