DAY 7 AND 8:

TASK 1: JAVA IO BASICS

Write a program that reads a text file and counts the frequency of each word using FileReader and FileWriter.

```
import java.io.*;
import java.util.HashMap;
import java.util.Map;

public class WordFrequencyCounter {
    public static void main(String[] args) {
        String inputFilePath = "input.txt";
        String outputFilePath = "output.txt";

        Map<String, Integer> wordFrequencyMap = new HashMap<>();

        try {
            FileReader fileReader = new FileReader(inputFilePath);
            BufferedReader bufferedReader = new
BufferedReader(fileReader);

            String line;
            while ((line = bufferedReader.readLine()) != null) {
                String[] words = line.split("\\s+");
                for (String word : words) {
                    word = word.replaceAll("[^a-zA-Z]", "").toLowerCase();
                    if (!word.isEmpty()) {
                        wordFrequencyMap.put(word,
wordFrequencyMap.getOrDefault(word, 0) + 1);
                    }
                }
            }
```

```
            bufferedReader.close();

            FileWriter fileWriter = new FileWriter(outputFilePath);
            BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);

            for (Map.Entry<String, Integer> entry :
wordFrequencyMap.entrySet()) {
                bufferedWriter.write(entry.getKey() + ": " + entry.getValue());
                bufferedWriter.newLine();
            }
            bufferedWriter.close();

            System.out.println("Word frequencies written to " +
outputFilePath);
        } catch (IOException e) {
            System.err.println("Error: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```

Problems @ Javadoc Declaration Console ×

```
<terminated> WordFrequencyCounter [Java Application] /snap/eclipse/87/pl
Word frequencies written to /home/rps/wipro2024fsd/output.txt
```

Open Save

output.txt
~/wipro2024fsd

```
counting: 1
a: 1
some: 1
containing: 1
this: 2
words: 1
for: 1
is: 2
used: 1
sample: 1
each: 1
frequency: 1
the: 1
file: 2
of: 1
text: 1
word: 1
```

TASK 2: SERIALIZATION AND DESERIALIZATION

Serialize a custom object to a file and then deserialize it back to recover the object state.

```
import java.io.*;

class CustomObject implements Serializable {
    private static final long serialVersionUID = 1L;
    private String name;
    private int age;

    public CustomObject(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }
}

public class SerializationDemo {
    public static void main(String[] args) {
        // Create a custom object
        CustomObject obj = new CustomObject("John", 30);

        // Serialize the object to a file
        String filename = "custom_object.ser";
        try (FileOutputStream fileOut = new FileOutputStream(filename);
```

```
            ObjectOutputStream objectOut = new
ObjectOutputStream(fileOut)) {
            objectOut.writeObject(obj);
            System.out.println("Custom object serialized and saved to " +
filename);
        } catch (IOException e) {
            System.err.println("Error: " + e.getMessage());
            e.printStackTrace();
        }

        // Deserialize the object from the file
        try (FileInputStream fileIn = new FileInputStream(filename);
             ObjectInputStream objectIn = new ObjectInputStream(fileIn)) {
            CustomObject restoredObj = (CustomObject)
objectIn.readObject();
            System.out.println("Custom object deserialized from file:");
            System.out.println("Name: " + restoredObj.getName());
            System.out.println("Age: " + restoredObj.getAge());
        } catch (IOException | ClassNotFoundException e) {
            System.err.println("Error: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```

Problems @ Javadoc Declaration Console ×

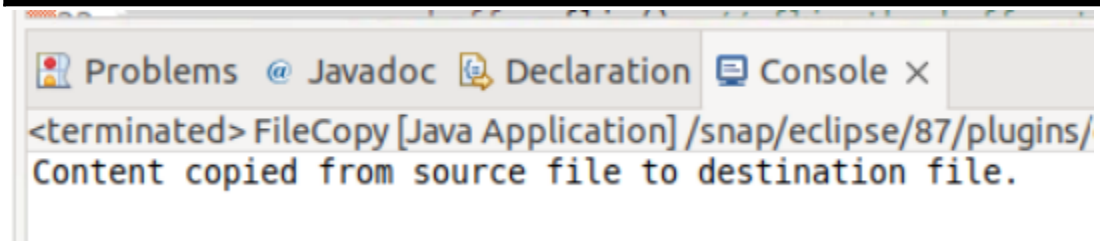
```
<terminated> SerializationDemo [Java Application] /snap/eclipse/87/pl
Custom object serialized and saved to custom_object.ser
Custom object deserialized from file:
Name: John
Age: 30
```

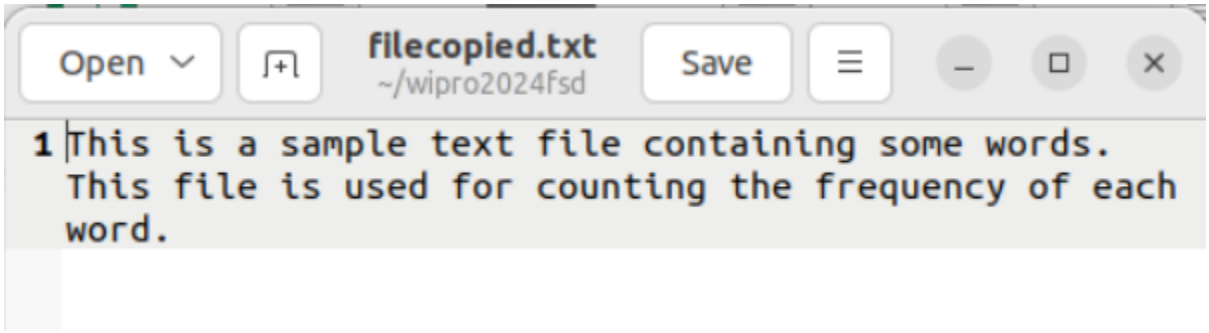
TASK 3: NEW IO (NIO)

Use NIO Channels and Buffers to read content from a file and write to another file.

```
import java.io.IOException;
import java.nio.ByteBuffer;
import java.nio.channels.FileChannel;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;

public class FileCopy {
    public static void main(String[] args) {
        Path sourcePath = Paths.get("source.txt");
        Path destinationPath = Paths.get("destination.txt");

        try (FileChannel sourceChannel = FileChannel.open(sourcePath,
StandardOpenOption.READ);
             FileChannel destinationChannel =
FileChannel.open(destinationPath, StandardOpenOption.CREATE,
                     StandardOpenOption.WRITE,
StandardOpenOption.TRUNCATE_EXISTING)) {

            ByteBuffer buffer = ByteBuffer.allocate(1024);

            while (sourceChannel.read(buffer) != -1) {
                buffer.flip(); // flip the buffer to prepare for reading
                destinationChannel.write(buffer); // write from buffer to
destination channel
                buffer.clear(); // clear the buffer for next read
            }
```

```
            System.out.println("Content copied from source file to destination file.");
        } catch (IOException e) {
            System.err.println("Error: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```





TASK 4: JAVA NETWORKING

Write a simple HTTP client that connects to a URL, sends a request, and displays the response headers and body.

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

public class SimpleHTTPClient {
    public static void main(String[] args) {
        String urlString = "https://www.example.com";

        try {
            // Create URL object
            URL url = new URL(urlString);

            // Open connection
            HttpURLConnection connection = (HttpURLConnection)
url.openConnection();

            // Set request method
            connection.setRequestMethod("GET");

            // Get response code
            int responseCode = connection.getResponseCode();
            System.out.println("Response Code: " + responseCode);

            // Read response headers
            System.out.println("Response Headers:");
            connection.getHeaderFields().forEach((key, value) ->
System.out.println(key + ": " + value));
```

```
            // Read response body
            System.out.println("\nResponse Body:");
            BufferedReader reader = new BufferedReader(new
InputStreamReader(connection.getInputStream()));
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
            reader.close();

            // Disconnect
            connection.disconnect();
        } catch (IOException e) {
            System.err.println("Error: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```

Problems @ Javadoc Declaration Console ×

```
<terminated> SimpleHTTPClient [Java Application] /snap/eclipse/87/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.10.v20
Response Code: 200
Response Headers:
null: [HTTP/1.1 200 OK]
X-Cache: [HIT]
Server: [ECAcc (lac/55A7)]
Last-Modified: [Thu, 17 Oct 2019 07:18:26 GMT]
Date: [Sat, 25 May 2024 08:05:02 GMT]
Accept-Ranges: [bytes]
Etag: ["3147526947"]
Cache-Control: [max-age=604800]
Vary: [Accept-Encoding]
Expires: [Sat, 01 Jun 2024 08:05:02 GMT]
Content-Length: [1256]
Age: [303469]
Content-Type: [text/html; charset=UTF-8]

Response Body:
<!doctype html>
<html>
<head>
    <title>Example Domain</title>

    <meta charset="utf-8" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvetica,

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
    }
    @media (max-width: 700px) {
        div {
            margin: 0 auto;
            width: auto;
        }
```

```
    }
    </style>
</head>

<body>
<div>
    <h1>Example Domain</h1>
    <p>This domain is for use in illustrative examples in documents. You may use this
    domain in literature without prior coordination or asking for permission.</p>
    <p><a href="https://www.iana.org/domains/example">More information...</a></p>
</div>
</body>
</html>
```

TASK 5: JAVA NETWORKING AND SERIALIZATION

Develop a basic TCP client and server application where the client sends a serialized object with 2 numbers and operation to be performed on them to the server, and the server computes the result and sends it back to the client. for eg, we could send 2, 2, "+" which would mean 2 + 2

Server

```
import java.io.*;
import java.net.*;

public class TCPServer {
    public static void main(String[] args) {
        int port = 12345;

        try (ServerSocket serverSocket = new ServerSocket(port)) {
            System.out.println("Server waiting for client on port " + port);

            while (true) {
                Socket socket = serverSocket.accept();
                System.out.println("Client connected: " + socket);

                ObjectInputStream objectInputStream = new
ObjectInputStream(socket.getInputStream());
                ObjectOutputStream objectOutputStream = new
ObjectOutputStream(socket.getOutputStream());

                // Read serialized object from client
                Calculation calculation = (Calculation)
objectInputStream.readObject();
                System.out.println("Received: " + calculation);

                // Perform calculation
```

```
            double result;
            switch (calculation.getOperation()) {
                case "+":
                    result = calculation.getNumber1() +
calculation.getNumber2();
                    break;
                case "-":
                    result = calculation.getNumber1() -
calculation.getNumber2();
                    break;
                case "*":
                    result = calculation.getNumber1() *
calculation.getNumber2();
                    break;
                case "/":
                    result = calculation.getNumber1() /
calculation.getNumber2();
                    break;
                default:
                    result = 0;
            }

            // Send result back to client
            objectOutputStream.writeDouble(result);
            objectOutputStream.flush();

            // Close streams and socket
            objectOutputStream.close();
            objectInputStream.close();
            socket.close();
        }
    } catch (IOException | ClassNotFoundException e) {
        e.printStackTrace();
    }
```

```
    }
}
```

Client

```
import java.io.*;
import java.net.*;

public class TCPClient {
    public static void main(String[] args) {
        String host = "localhost";
        int port = 12345;

        try (Socket socket = new Socket(host, port);
             ObjectOutputStream objectOutputStream = new
ObjectOutputStream(socket.getOutputStream());
             ObjectInputStream objectInputStream = new
ObjectInputStream(socket.getInputStream())) {

            // Prepare and send serialized object to server
            Calculation calculation = new Calculation(2, 2, "+");
            objectOutputStream.writeObject(calculation);
            objectOutputStream.flush();
            System.out.println("Sent: " + calculation);

            // Receive result from server
            double result = objectInputStream.readDouble();
            System.out.println("Result: " + result);

        } catch (IOException e) {
            e.printStackTrace();
        }
```

```
    }
}
```

Calculation

```
import java.io.Serializable;

public class Calculation implements Serializable {
    private static final long serialVersionUID = 1L;
    private double number1;
    private double number2;
    private String operation;

    public Calculation(double number1, double number2, String operation)
{
        this.number1 = number1;
        this.number2 = number2;
        this.operation = operation;
    }

    public double getNumber1() {
        return number1;
    }

    public double getNumber2() {
        return number2;
    }

    public String getOperation() {
        return operation;
    }
```

```
    @Override
    public String toString() {
        return number1 + " " + operation + " " + number2;
    }
}
```

Server Output

```
Console ×  Problems  @ Javadoc  Declaration
TCPServer [Java Application] /snap/eclipse/87/plugins/org.eclipse.justj.openjdk.hotspot
Server waiting for client on port 12345
Client connected: Socket[addr=/127.0.0.1,port=49508,localport=12345]
Received: 2.0 + 2.0
```

Client Output

```
Console ×  Problems  @ Javadoc  Declaration
<terminated> TCPClient [Java Application] /snap/eclipse/87/plugins/
Sent: 2.0 + 2.0
Result: 4.0
```

TASK 6: JAVA 8 DATE AND TIME API

Write a program that calculates the number of days between two dates input by the user.

```
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.temporal.ChronoUnit;
import java.util.Scanner;

public class DaysBetweenDatesCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Prompt the user to enter the first date
        System.out.print("Enter the first date (YYYY-MM-DD): ");
        String date1Str = scanner.nextLine();

        // Prompt the user to enter the second date
        System.out.print("Enter the second date (YYYY-MM-DD): ");
        String date2Str = scanner.nextLine();

        // Parse the input strings to LocalDate objects
        LocalDate date1 = LocalDate.parse(date1Str,
DateTimeFormatter.ISO_LOCAL_DATE);
        LocalDate date2 = LocalDate.parse(date2Str,
DateTimeFormatter.ISO_LOCAL_DATE);

        // Calculate the number of days between the two dates
        long daysBetween = ChronoUnit.DAYS.between(date1, date2);

        // Output the result
        System.out.println("Number of days between the two dates: " +
daysBetween);
```

```
        scanner.close();
    }
}
```

```
Console ×   Problems   @ Javadoc   Declaration
<terminated> DaysBetweenDatesCalculator [Java Application] /sna
Enter the first date (YYYY-MM-DD): 2000-03-08
Enter the second date (YYYY-MM-DD): 2024-05-24
Number of days between the two dates: 8843
```

TASK 7: TIMEZONE

Create a timezone converter that takes a time in one timezone and converts it to another timezone.

```
import java.time.LocalDateTime;
import java.time.ZoneId;
import java.time.ZonedDateTime;
import java.time.format.DateTimeFormatter;
import java.util.Scanner;

public class TimezoneConverter {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the date and time (YYYY-MM-DD HH:MM): ");
        String inputDateTime = scanner.nextLine();

        System.out.print("Enter the source timezone: ");
        String sourceTimezone = scanner.nextLine();

        System.out.print("Enter the target timezone: ");
        String targetTimezone = scanner.nextLine();

        LocalDateTime localDateTime = LocalDateTime.parse(inputDateTime, DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm"));

        ZonedDateTime sourceZonedDateTime = ZonedDateTime.of(localDateTime, ZoneId.of(sourceTimezone));
```

```
        ZonedDateTime targetZonedDateTime =
sourceZonedDateTime.withZoneSameInstant(ZoneId.of(targetTimezone))
;

        String outputDateTime = DateTimeFormatter.ofPattern("yyyy-MM-dd
HH:mm").format(targetZonedDateTime);

        System.out.println("Converted time in " + targetTimezone + ": " +
outputDateTime);

        scanner.close();
    }
}
```

```
Console ×   Problems   @ Javadoc   Declaration
<terminated> TimezoneConverter [Java Application] /snap/eclipse/87/plugins/
Enter the date and time (YYYY-MM-DD HH:MM): 2024-05-25 14:02
Enter the source timezone: America/New_York
Enter the target timezone: Asia/Kolkata
Converted time in Asia/Kolkata: 2024-05-25 23:32
```