Manjula Nannuri
Day_21_Assignment

Task 1: Java IO Basics

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

```
import
java.io.BufferedReader;
import
java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
public class WordFreqCounter {
   public static void main(String[]
      args) { String inputFilePath =
      "input.txt"; String outputFilePath
      = "output.txt";
      Map<String, Integer>wordCounts =
     countWordFreq(inputFilePath);
     writeWordCountsToFile(wordCounts, outputFilePath);
   }

   public static Map<String, Integer>countWordFreq(String
```

```
filePath) {Map<String, Integer>wordCounts = new

HashMap<>();

try (BufferedReaderbr = new BufferedReader(new FileReader(filePath))) {
```

```
        String line;
        while ((line = br.readLine()) !=
         null) {String[] words =
         line.split("\\W+");
           for (String word :
              words) {if
              (word.isEmpty()) {
                 continue;
              }
              word = word.toLowerCase();
wordCounts.put(word, wordCounts.getOrDefault(word, 0) + 1);
           }
        }
     } catch (IOException e) {
System.err.println("Error reading the file: " + e.getMessage());
     }

     return wordCounts;
  }

  public static void writeWordCountsToFile(Map<String, Integer>wordCounts, String
     filePath) {try (BufferedWriterbw = new BufferedWriter(new FileWriter(filePath))) {
        for (Map.Entry<String, Integer>entry
          :wordCounts.entrySet()) {bw.write(entry.getKey() + ": "
```

```
      + entry.getValue()); bw.newLine();
    }
} catch (IOException e) {
```

```
System.err.println("Error writing to the file: " + e.getMessage());
      }
   }
}
```

Output:

Error reading the file: input.txt (No such file or

directory)Error writing to the file: output.txt

(Permission denied)

Task 2: Serialization and Deserialization

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

Custom Object: Person

```
import java.io.Serializable;
public class Person implements Serializable {
   private static final long serialVersionUID = 1L;
   private String name;
   private int age;
   public Person(String name, int age) {
      this.name = name;
     this.age = age;
   }
   public String getName() {
       return name;
}
   public intgetAge() {
```

```
        return age;
    }

    public String toString() {
        return "Person{name='" + name + "', age=" + age + "}";
    }
}
```

Serialization and Deserialization

```
import java.io.*;
public class SerializationExample {
  public static void main(String[] args) {
        Person person = new Person("Alexa", 35);
        String filename = "person.ser";
        serializePerson(person, filename);
     Person deserializedPerson = deserializePerson(filename);
        if (deserializedPerson != null) {
System.out.println("Deserialized Person: " + deserializedPerson);
        }
    }
  public static void serializePerson(Person person, String filename) {
try (FileOutputStreamfileOut = new FileOutputStream(filename);
ObjectOutputStream out = new ObjectOutputStream(fileOut)) {
out.writeObject(person);
System.out.println("Serialized data is saved in " + filename);
```

```
        } catch (IOExceptioni) {
i.printStackTrace();
        }
    }

    public static Person deserializePerson(String filename) {
        Person person = null;
        try (FileInputStreamfileIn = new FileInputStream(filename);
ObjectInputStream in = new ObjectInputStream(fileIn)) {
            person = (Person) in.readObject();
        } catch (IOExceptioni) {
i.printStackTrace();
        } catch (ClassNotFoundException c) {
System.out.println("Person class not found");
c.printStackTrace();
        }
        return person;
    }
}
```

Output:

```
Serialized data is saved in person.ser
Deserialized Person: Person{name='Alexa', age=35}
```

Task 3: New IO (NIO)

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

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.nio.ByteBuffer;
import java.nio.channels.FileChannel;
public class FileCopy {
   public static void main(String[] args) {
      try (FileChannel inChannel = new FileInputStream("input.txt").getChannel();
         FileChannel outChannel = new FileOutputStream("output.txt").getChannel()) {
         ByteBuffer buffer = ByteBuffer.allocate(1024);
         while (inChannel.read(buffer) != -1) {
            buffer.flip();  // flip the buffer from writing mode to reading mode
            outChannel.write(buffer);
            buffer.clear(); // clear the buffer for the next read
         }

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

```
}
```

Output :

```
Error during file copy: source.txt
```

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 {
  URL url = new URL(urlString);
HttpURLConnection connection = (HttpURLConnection) url.openConnection();
connection.setRequestMethod("GET");
intresponseCode = connection.getResponseCode();
System.out.println("Response Code: " + responseCode);
System.out.println("Response Headers:");
connection.getHeaderFields().forEach((key, value) -> {
System.out.println(key + ": " + value);
            });
```

```
System.out.println("Response Body:");

        try (BufferedReader reader = new BufferedReader(new
InputStreamReader(connection.getInputStream()))) {

            String line;

            while ((line = reader.readLine()) != null) {

System.out.println(line);

            }

        }

connection.disconnect();

    } catch (IOException e) {

e.printStackTrace();

    }

  }

}
```

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.

```
import java.io.Serializable;

public class Operation implements Serializable {

  private static final long serialVersionUID = 1L;

  private int number1;

  private int number2;
```

```
    private String operation;
    public Operation(int number1, int number2, String operation) {
this.number1 = number1;
this.number2 = number2;
this.operation = operation;
    }

    public int getNumber1() {
        return number1;
    }
    public int getNumber2() {
        return number2;
    }
    public String getOperation() {
        return operation;
    }
}
// server implementation
import java.io.*;
import java.net.ServerSocket;
import  java.net.Socket;
public class Server {
    public static void main(String[] args) {
    int port = 12345;
```

```
        try (ServerSocketserverSocket = new ServerSocket(port)) {
System.out.println("Server is listening on port " + port);

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

                new ServerThread(socket).start();
            }
        } catch (IOException e) {
e.printStackTrace();
        }
    }
}

class ServerThread extends Thread {
    private Socket socket;

    public ServerThread(Socket socket) {
this.socket = socket;
    }

    public void run() {
```

```
        try (ObjectInputStreamois = new ObjectInputStream(socket.getInputStream());
ObjectOutputStreamoos = new ObjectOutputStream(socket.getOutputStream())) {


            Operation operation = (Operation) ois.readObject();
int result = performOperation(operation);
oos.writeInt(result);
oos.flush();
        } catch (IOException | ClassNotFoundException e) {
e.printStackTrace();
        }
    }


    private intperformOperation(Operation operation) {
int num1 = operation.getNumber1();
int num2 = operation.getNumber2();
        String op = operation.getOperation();


        switch (op) {
            case "+":
                return num1 + num2;
            case "-":
                return num1 - num2;
            case "*":
                return num1 * num2;
```

```
            case "/":
                if (num2 != 0) {
                    return num1 / num2;
                } else {
                    throw new IllegalArgumentException("Division by zero");
                }
            default:
                throw new IllegalArgumentException("Invalid operation: " + op);
        }
    }
}
```

Implement the Client:

```
import java.io.*;
import java.net.Socket;
public class Client {
    public static void main(String[] args) {
        String hostname = "localhost";
int port = 12345;
        try (Socket socket = new Socket(hostname, port);
ObjectOutputStreamoos = new ObjectOutputStream(socket.getOutputStream());
ObjectInputStreamois = new ObjectInputStream(socket.getInputStream())) {
Operation operation = new Operation(2, 2, "+");
oos.writeObject(operation);
oos.flush();
```

```
int result = ois.readInt();
System.out.println("Result: " + result);
        } catch (IOException e) {
e.printStackTrace();
        }
    }
}
```

Server Output:

Server is listening on port 12345

Client connected

Client output:

Result: 4

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 DaysBtwDatesCalculator {
    public static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in);
System.out.print("Enter the first date (YYYY-MM-DD): ");
        String date1Str = scanner.next();
System.out.print("Enter the second date (YYYY-MM-DD): ");
        String date2Str = scanner.next();
LocalDate date1 = LocalDate.parse(date1Str);
LocalDate date2 = LocalDate.parse(date2Str);
 long daysBetween = ChronoUnit.DAYS.between(date1, date2);
System.out.println("Number of days between " + date1 + " and " + date2 + " is: " +
Math.abs(daysBetween));
    }
}
```

Output:

Enter the first date (YYYY-MM-DD): 2001-03-19

Enter the second date (YYYY-MM-DD): 2006-04-18

Number of days between 2001-03-09 and 2005-04-08 is: 2511

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:ss): ");

      String dateTimeStr = scanner.nextLine();

System.out.print("Enter the source timezone (e.g., America/New_York): ");

      String sourceTimezone = scanner.nextLine();

System.out.print("Enter the target timezone (e.g., Europe/London): ");

      String targetTimezone = scanner.nextLine();

LocalDateTimelocalDateTime = LocalDateTime.parse(dateTimeStr,
DateTimeFormatter.ofPattern("yyyy-MM-ddHH:mm:ss"));

ZonedDateTimesourceZonedDateTime = localDateTime.atZone(ZoneId.of(sourceTimezone));

ZonedDateTimetargetZonedDateTime
sourceZonedDateTime.withZoneSameInstant(ZoneId.of(targetTimezone));

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

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

   }

}
```

Output:

```
Enter the date and time (YYYY-MM-DD HH:mm:ss): 2024-06-10 12:00:00

Enter the source timezone (e.g., America/New_York): America/New_York

Enter the target timezone (e.g., Europe/London): Europe/London
```

Converted time in Europe/London: 2024-06-10 17:00:00