

Assignment Day-20

Core Java with DS and Algorithms

Task 1: Java IO Basics

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

```
package day20;

import java.io.BufferedReader;

import java.io.FileReader;

import java.util.HashMap;

import java.util.Map;

public class FileReaderWriter {

    public static void main(String[] args) throws Exception {

        String fileName = "C:\\\\Joshnitha_wipro\\\\ProductFile.txt";

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

        try (BufferedReader reader = new BufferedReader(new FileReader(fileName)))
        {

            String line;

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

                String[] words = line.toLowerCase().split("\\s+");

                for (String word : words) {

                    if (wordCounts.containsKey(word)) {

                        wordCounts.put(word, wordCounts.get(word) + 1);

                    } else {

                        wordCounts.put(word, 1);

                    }

                }

            }

        }

    }

}
```

```

}

}

}

for (Map.Entry<String, Integer> entry : wordCounts.entrySet()) {

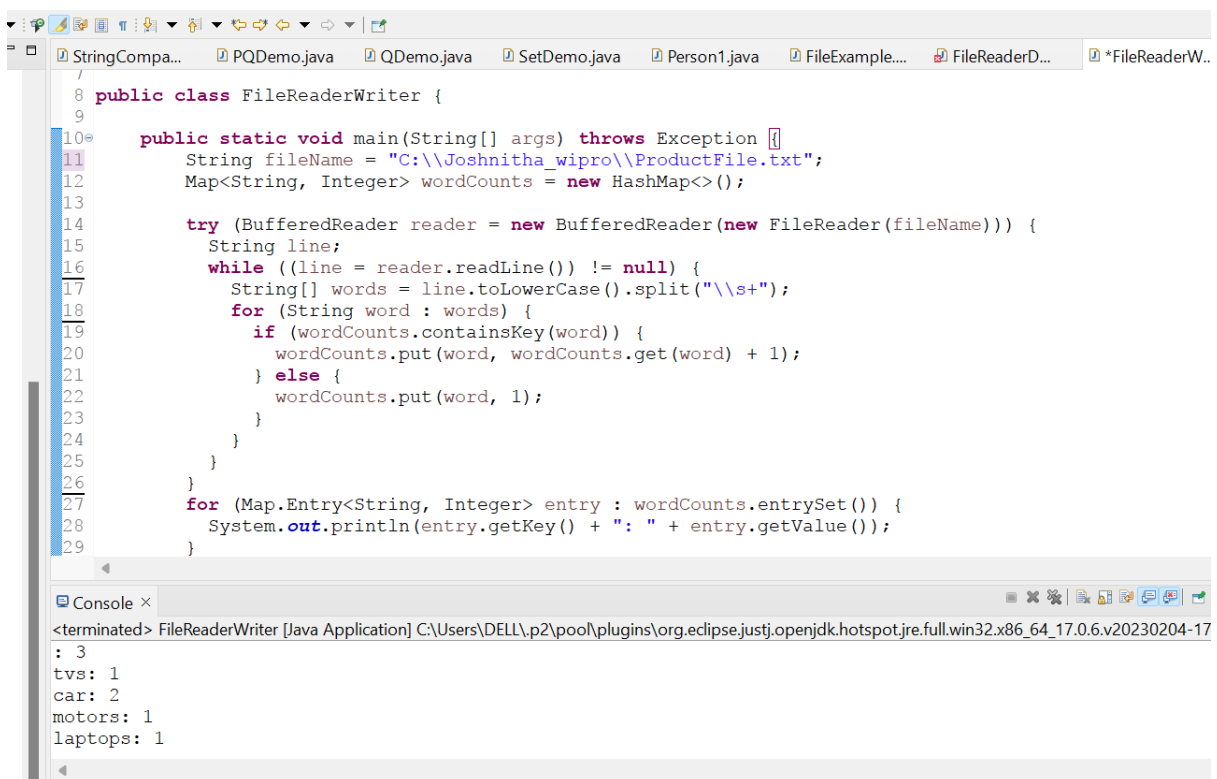
System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}

```



```

8 public class FileReaderWriter {
9
10     public static void main(String[] args) throws Exception {
11         String fileName = "C:\\Joshnitha_wipro\\ProductFile.txt";
12         Map<String, Integer> wordCounts = new HashMap<>();
13
14         try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {
15             String line;
16             while ((line = reader.readLine()) != null) {
17                 String[] words = line.toLowerCase().split("\\s+");
18                 for (String word : words) {
19                     if (wordCounts.containsKey(word)) {
20                         wordCounts.put(word, wordCounts.get(word) + 1);
21                     } else {
22                         wordCounts.put(word, 1);
23                     }
24                 }
25             }
26         }
27         for (Map.Entry<String, Integer> entry : wordCounts.entrySet()) {
28             System.out.println(entry.getKey() + ": " + entry.getValue());
29         }
30     }
31 }

```

Console ×

```

<terminated> FileReaderWriter [Java Application] C:\Users\DELL\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-17
: 3
tvs: 1
car: 2
motors: 1
laptops: 1

```

Task 2: Serialization and Deserialization

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

```

package serialization;

import java.io.Serializable;

public class Employee implements Serializable{

private transient int eid;

private String ename; // declare it static and try

```

```

public Employee(int eid, String ename) {

    super();

    this.eid = eid;

    this.ename = ename;

}

@Override

public String toString() {

    return "Employee [eid=" + eid + ", ename=" + ename + "]";

}

}

import java.io.ObjectOutputStream;

public class SerializationExample {

    public static void main(String[] args) throws IOException,
    FileNotFoundException {

        Employee emp = new Employee(101, "javeed");

        FileOutputStream fos = new FileOutputStream("employee.ser");

        ObjectOutputStream oos = new ObjectOutputStream(fos);

        oos.writeObject(emp);

        System.out.println("Employee obj Serialized");

    }

}

package serialization;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.ObjectInputStream;

public class DeserializationExample {

```

```
public static void main(String[] args) throws FileNotFoundException,
IOException , ClassNotFoundException {
```

```
FileInputStream fis = new FileInputStream("employee.ser");
```

```
ObjectInputStream ois = new ObjectInputStream(fis);
```

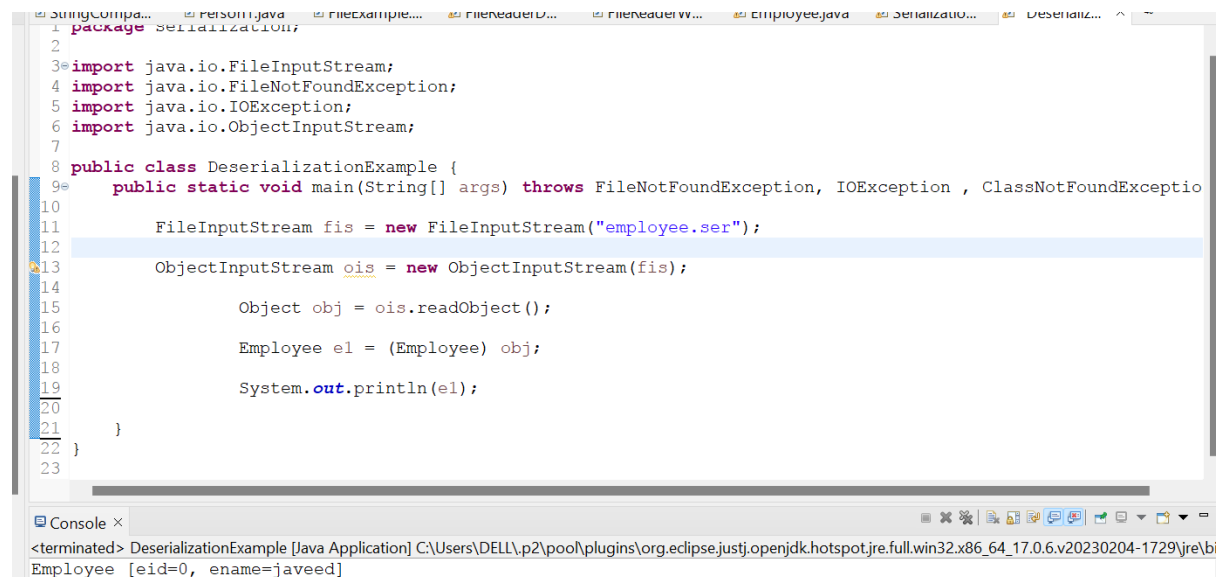
```
Object obj = ois.readObject();
```

```
Employee e1 = (Employee) obj;
```

```
System.out.println(e1);
```

```
}
```

```
}
```



The screenshot shows an IDE with a project named 'Serialization'. The 'DeserializationExample.java' file is open, displaying the following code:

```
1 package Serialization;
2
3 import java.io.FileInputStream;
4 import java.io.FileNotFoundException;
5 import java.io.IOException;
6 import java.io.ObjectInputStream;
7
8 public class DeserializationExample {
9     public static void main(String[] args) throws FileNotFoundException, IOException , ClassNotFoundException {
10
11         FileInputStream fis = new FileInputStream("employee.ser");
12
13         ObjectInputStream ois = new ObjectInputStream(fis);
14
15         Object obj = ois.readObject();
16
17         Employee e1 = (Employee) obj;
18
19         System.out.println(e1);
20     }
21 }
22
23
```

The console output at the bottom shows the program terminated successfully and printed the employee details:

```
<terminated> DeserializationExample [Java Application] C:\Users\DELL\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-1729\jre\bin\java.exe
Employee [eid=0, ename=javeed]
```

Task 3: New IO (NIO)

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

```
package channel;
```

```
import java.io.FileNotFoundException;
```

```
import java.io.IOException;
```

```
import java.io.RandomAccessFile;
```

```
import java.nio.ByteBuffer;
```

```
import java.nio.channels.FileChannel;
```

```

public class FileChannelDemo {

    public static void main(String[] args) throws
    FileNotFoundException, IOException {

        RandomAccessFile file = new RandomAccessFile("c:/Test/Input.txt", "r");

        FileChannel fileChannel = file.getChannel();

        ByteBuffer byteBuffer = ByteBuffer.allocate(512); // creates new buffer of
        size 512 bytes

        while(fileChannel.read(byteBuffer) > 0) {

            byteBuffer.flip();

            while(byteBuffer.hasRemaining()) {

                System.out.print((char) byteBuffer.get());

            }

        }

        fileChannel.close();

    }

}

```

The screenshot shows an IDE window with the following tabs: StringCompa..., Deserializa..., TestOperatio..., EmployeeSor..., FileChannelD..., ServerSocke..., and SocketChann... The active tab is FileChannelD... The code in the editor is as follows:

```

1 package channel;
2
3 import java.io.FileNotFoundException;
4 import java.io.IOException;
5 import java.io.RandomAccessFile;
6 import java.nio.ByteBuffer;
7 import java.nio.channels.FileChannel;
8
9 public class FileChannelDemo {
10
11     public static void main(String[] args) throws FileNotFoundException, IOException {
12
13         RandomAccessFile file = new RandomAccessFile("c:/Test/Input.txt", "r");
14
15         FileChannel fileChannel = file.getChannel();
16
17         ByteBuffer byteBuffer = ByteBuffer.allocate(512); // creates new buffer of size 512 bytes
18
19         while(fileChannel.read(byteBuffer) > 0) {
20
21             byteBuffer.flip();
22
23             while(byteBuffer.hasRemaining()) {
24
25                 System.out.print((char) byteBuffer.get());
26
27             }
28
29         }
30
31         fileChannel.close();
32
33     }
34
35 }

```

The console output at the bottom shows the following text:

```

<terminated> FileChannelDemo [Java Application] C:\Users\DELL\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-1729\jre\bin\java
Hello world
abcd
ABCD
Gd mrng

```

Task 4: Java Networking

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

```
package day20;

import java.io.BufferedReader;

import java.io.IOException;

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");

            int responseCode = 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 (ServerSocket serverSocket = 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 (ObjectInputStream ois = new ObjectInputStream(socket.getInputStream());
```

```
ObjectOutputStream oos = 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 int performOperation(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);
ObjectOutputStream oos = new ObjectOutputStream(socket.getOutputStream());
ObjectInputStream ois = 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();

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

        ZonedDateTime sourceZonedDateTime = localDateTime.atZone(ZoneId.of(sourceTimezone));

        ZonedDateTime targetZonedDateTime =
            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

```