Name:Vishakha Avinash kale

Day20 assignment

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.*;
public class WordFrequencyCounter {
 public static void main(String[] args) {
 String inputFilePath = "input.txt";
 String outputFilePath = "output.txt";
 // Read the text file and count word frequencies
 Map<String, Integer> wordCountMap =
readFileAndCountWords(inputFilePath);
 // Write the word frequencies to the output file
 writeWordFrequenciesToFile(wordCountMap, outputFilePath);
 }
private static Map<String, Integer> readFileAndCountWords(String
filePath) {
 Map<String, Integer> wordCountMap = new HashMap<>();
 try (FileReader fr = new FileReader(filePath);
 BufferedReader br = new BufferedReader(fr)) {
 String line;
 while ((line = br.readLine()) != null) {
 String[] words = line.split("\\W+");
 for (String word : words) {
 if (!word.isEmpty()) {
 word = word.toLowerCase();
 wordCountMap.put(word,
```

```
wordCountMap.getOrDefault(word, 0) + 1);
 }
 }
 }
 } catch (IOException e) {
 System.err.println("Error reading file: " + e.getMessage());
 }
 return wordCountMap;
 }
private static void writeWordFrequenciesToFile(Map<String,
Integer> wordCountMap, String filePath) {
 try (FileWriter fw = new FileWriter(filePath);
 BufferedWriter bw = new BufferedWriter(fw)) {
 for (Map.Entry<String, Integer> entry :
wordCountMap.entrySet()) {
 bw.write(entry.getKey() + ": " + entry.getValue());
 bw.newLine();
 }
 } catch (IOException e) {
 System.err.println("Error writing to file: " + e.getMessage());
 }
 }
}
```

Task 2: Serialization and Deserialization

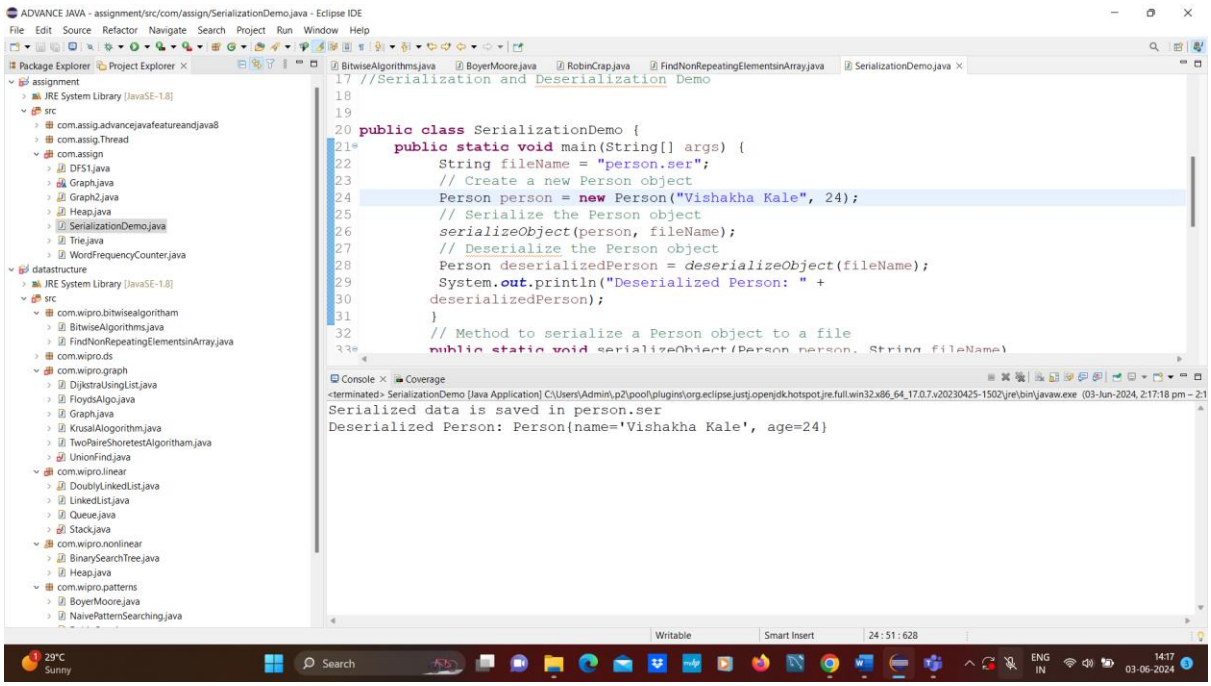
Serialize a custom object to a file and then deserialize it back to recover the object state.

```
import java.io.*;
//Custom class implementing Serializable
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;
}
@Override
public String toString() {
 return "Person{name='" + name + "', age=" + age + "}";
}
}
//Serialization and Deserialization Demo
public class SerializationDemo {
public static void main(String[] args) {
 String fileName = "person.ser";
 // Create a new Person object
 Person person = new Person("Vijay Patil", 24);
 // Serialize the Person object
 serializeObject(person, fileName);
 // Deserialize the Person object
 Person deserializedPerson = deserializeObject(fileName);
 System.out.println("Deserialized Person: " +
deserializedPerson);
}
// Method to serialize a Person object to a file
public static void serializeObject(Person person, String fileName)
{
 try (FileOutputStream fileOut = new FileOutputStream(fileName);
```

```
 ObjectOutputStream out = new ObjectOutputStream(fileOut))
{
 out.writeObject(person);
 System.out.println("Serialized data is saved in " +
fileName);
 } catch (IOException i) {
 i.printStackTrace();
 }
}
// Method to deserialize a Person object from a file
public static Person deserializeObject(String fileName) {
 Person person = null;
 try (FileInputStream fileIn = new FileInputStream(fileName);
 ObjectInputStream in = new ObjectInputStream(fileIn)) {
 person = (Person) in.readObject();
 } catch (IOException | ClassNotFoundException i) {
 i.printStackTrace();
}
return person;
}
}
```

OUTPUT:



Task 3: New IO (NIO)

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

```
package com.wipro;


import java.io.IOException;

import java.nio.ByteBuffer;

import java.nio.channels.FileChannel;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.nio.file.StandardOpenOption;


public class Mnioc {

        String fileName = "mydir/rhymes.txt";


  public void createDirectory() {

    Path p = Paths.get("mydir");
```

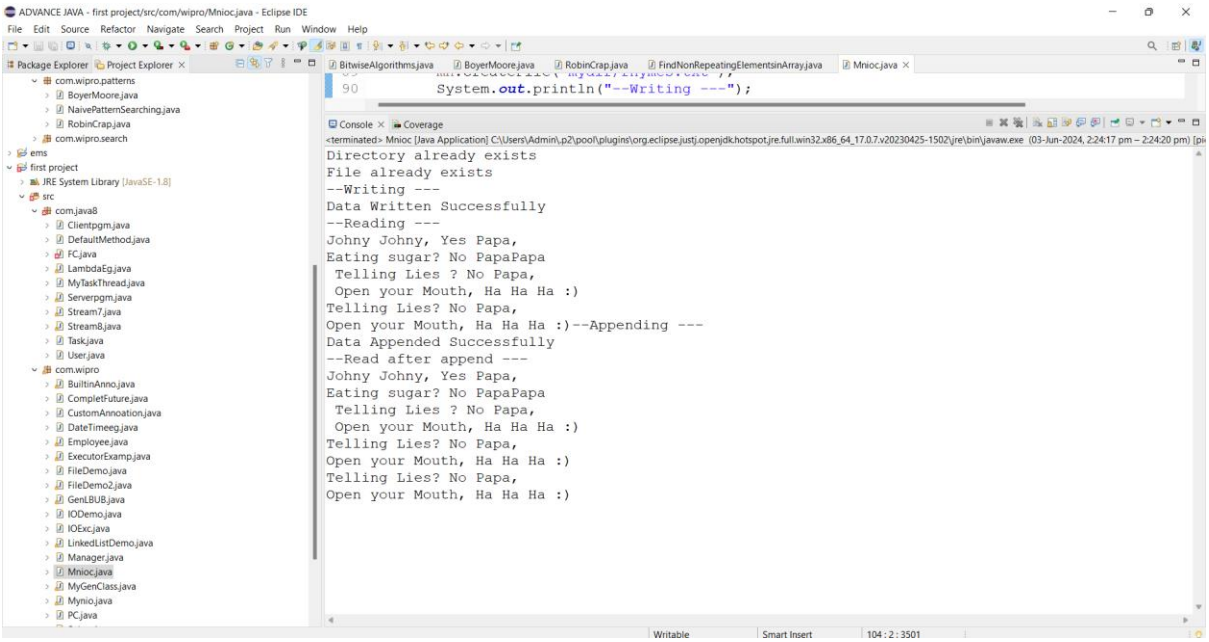
```
    try {
        if (Files.exists(p)) {
            System.out.println("Directory already exists");
        } else {
            Path cPath = Files.createDirectories(p);
            System.out.println("Directory created at " + cPath.toString());
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public void createFile(String fileName) {
    Path f = Paths.get(fileName);
    try {
        if (Files.exists(f)) {
            System.out.println("File already exists");
        } else {
            Path cFile = Files.createFile(f);
            System.out.println("File created at " + cFile.toString());
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public void readFile() {
    Path f = Paths.get(fileName);
    try (FileChannel fileChannel = FileChannel.open(f, StandardOpenOption.READ)) {
```

```
            ByteBuffer buffer = ByteBuffer.allocate(1024);
            while (fileChannel.read(buffer) > 0) {
                buffer.flip();
                while (buffer.hasRemaining()) {
                    System.out.print((char) buffer.get());
                }
                buffer.clear();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void writeFile(String fileName) {
        Path f = Paths.get(fileName);
        try (FileChannel fileChannel = FileChannel.open(f, StandardOpenOption.WRITE,
StandardOpenOption.CREATE)) {
            String content = "Johny Johny, Yes Papa,\nEating sugar? No Papa";
            ByteBuffer buffer = ByteBuffer.wrap(content.getBytes());
            fileChannel.write(buffer);
            System.out.println("Data Written Successfully");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void appendFile(String fileName) {
        Path f = Paths.get(fileName);
        try (FileChannel fileChannel = FileChannel.open(f, StandardOpenOption.APPEND)) {
            String content = "\nTelling Lies? No Papa,\nOpen your Mouth, Ha Ha Ha :)";
```

```
        ByteBuffer buffer = ByteBuffer.wrap(content.getBytes());
        fileChannel.write(buffer);
        System.out.println("Data Appended Successfully");
    } catch (IOException e) {
        e.printStackTrace();
    }
}

public static void main(String[] args) {
    Mnioc mn = new Mnioc();

    // Create a directory
    mn.createDirectory();

    // Create a file
    mn.createFile("mydir/rhymes.txt");
    System.out.println("--Writing ---");
    // Write to a file
    mn.writeFile(mn.fileName);
    System.out.println("--Reading ---");
    // Read from file
    mn.readFile();
    System.out.println("--Appending ---");
    // Append to a file
    mn.appendFile(mn.fileName);
    System.out.println("--Read after append ---");
    // Read from file
    mn.readFile();
}
```

}

OUTPUT:



Task 4: Java Networking

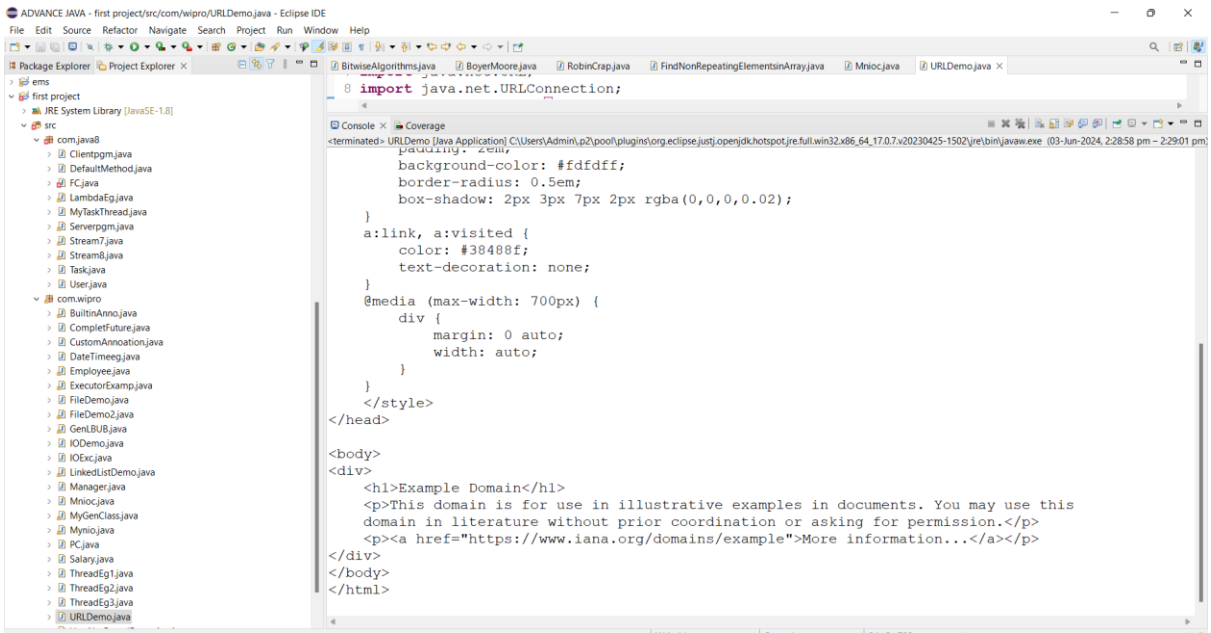
Write a simple HTTP client that connects to a URL, sends a request,

and displays the response headers and body.

```
package com.socket;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.net.MalformedURLException;

import java.net.URL;

import java.net.URLConnection;

public class URLDemo {

public static void main(String[] args) {

 try {

 URL url = new URL("http://www.example.com");


 URLConnection urlcon = url.openConnection();
```

```
BufferedReader br = new BufferedReader(new
InputStreamReader(urlcon.getInputStream()));

String line;
while((line = br.readLine()) != null) {
System.out.println(line);
}
br.close();


} catch (MalformedURLException e) {

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

e.printStackTrace();
}
```

OUTPUT:



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

Operationrequest:

```
package clinentserverapplication;

import java.io.Serializable;

public class OperationRequest implements Serializable {

 private static final long serialVersionUID = 1L;


private double number1;

private double number2;

 private String operation;


 public OperationRequest(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;

 }

}
```

Opretation server:;

```
package clinentserverapplication;
import java.io.*;
import java.net.*;
public class OperationServer {
 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) {
 try (Socket socket = serverSocket.accept();
 ObjectInputStream ois = new
ObjectInputStream(socket.getInputStream());
 ObjectOutputStream oos = new
ObjectOutputStream(socket.getOutputStream())) {
 OperationRequest request = (OperationRequest)
ois.readObject();
 double result = performOperation(request);
 oos.writeObject(result);
 oos.flush();
 } catch (IOException | ClassNotFoundException ex) {
 ex.printStackTrace();
 }
 }
 } catch (IOException ex) {
 ex.printStackTrace();
 }
```

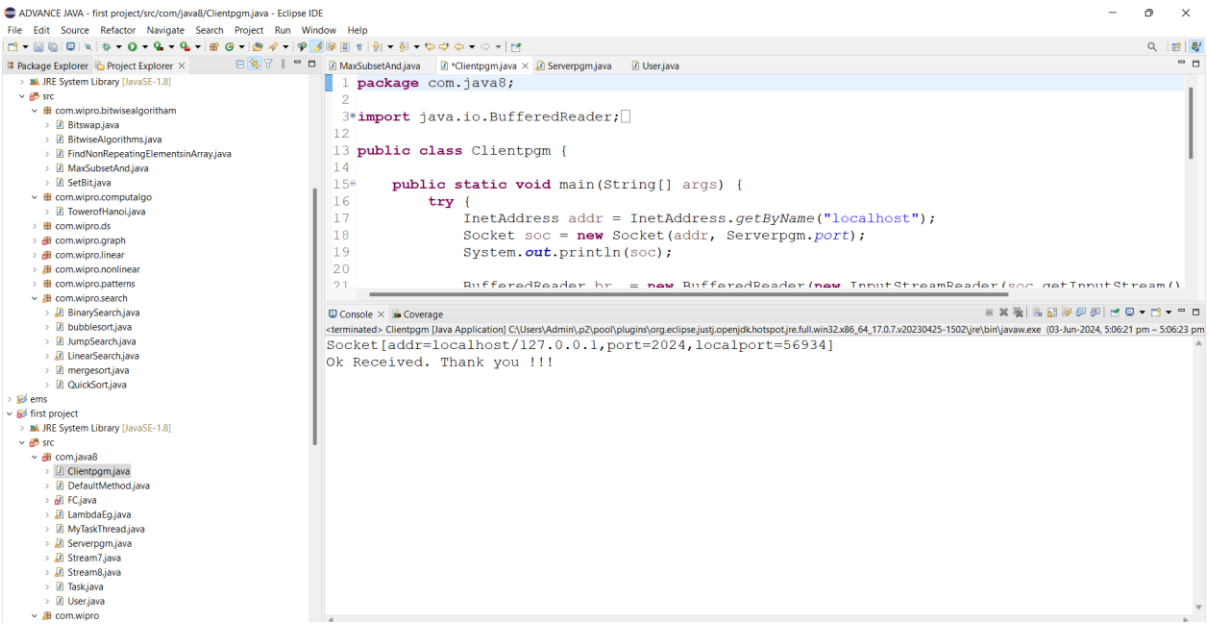
```
 }
 private static double performOperation(OperationRequest
request) {
 double number1 = request.getNumber1();
double number2 = request.getNumber2();
 String operation = request.getOperation();
 switch (operation) {
 case "+":
 return number1 + number2;
 case "-":
 return number1 - number2;
 case "*":
 return number1 * number2;
 case "/":
 if (number2 != 0) {
 return number1 / number2;
 } else {
 throw new IllegalArgumentException("Division by zero");
 }
 default:
 throw new UnsupportedOperationException("Unsupported
operation: " + operation);
 }
 }
}
```

Operation client:

```
package clinentserverapplication;


import java.io.*;
```

```
import java.net.*;
public class OperationClient {
 public static void main(String[] args) {
 String host = "localhost";
 int port = 12345;
 try (Socket socket = new Socket(host, port);
 ObjectOutputStream oos = new
ObjectOutputStream(socket.getOutputStream());
 ObjectInputStream ois = new
ObjectInputStream(socket.getInputStream())) {
 OperationRequest request = new OperationRequest(2, 2, "+");
 oos.writeObject(request);
 oos.flush();
 double result = (double) ois.readObject();
 System.out.println("Result: " + result);
 } catch (IOException | ClassNotFoundException ex) {
 ex.printStackTrace();
}
 }
}
```

Output:

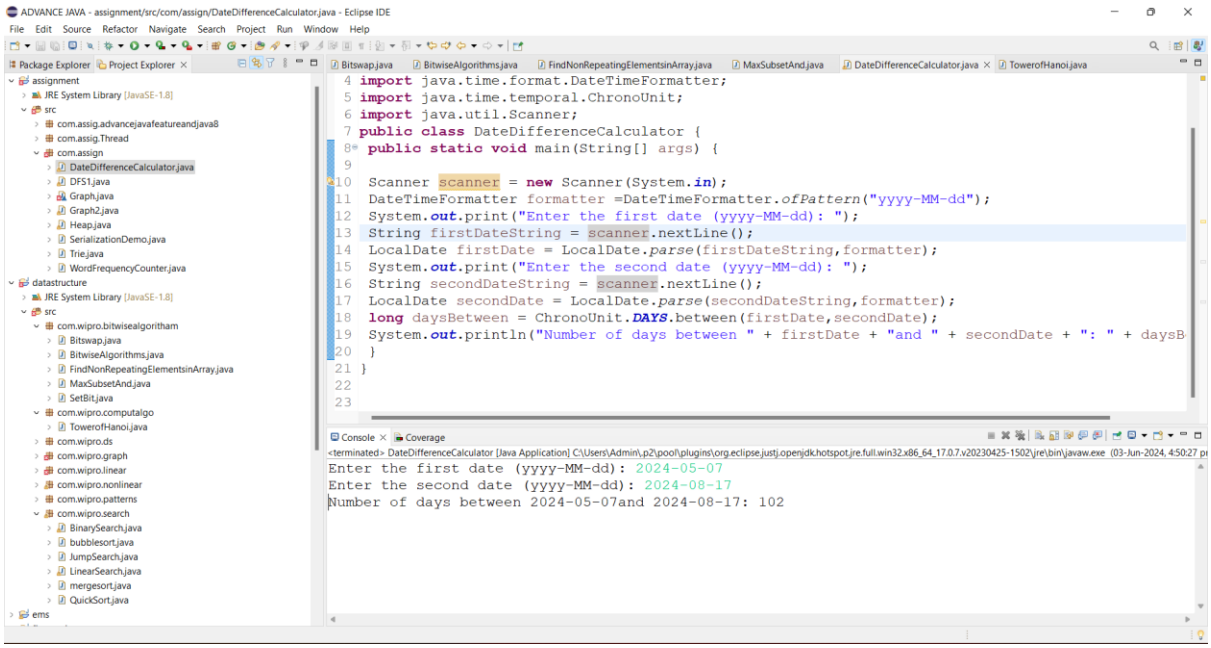


Task 6: Java 8 Date and Time API Write a program that calculates the number of days between two dates input by the user.

```
import java.time.LocalDate;

import java.time.format.DateTimeFormatter;

import java.time.temporal.ChronoUnit;

import java.util.Scanner;

public class DateDifferenceCalculator {

 public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

 DateTimeFormatter formatter =

DateTimeFormatter.ofPattern("yyyy-MM-dd");

 System.out.print("Enter the first date (yyyy-MM-dd): ");

 String firstDateString = scanner.nextLine();

 LocalDate firstDate = LocalDate.parse(firstDateString,

formatter);

 System.out.print("Enter the second date (yyyy-MM-dd): ");

 String secondDateString = scanner.nextLine();

 LocalDate secondDate = LocalDate.parse(secondDateString,

formatter);
```

```
 long daysBetween = ChronoUnit.DAYS.between(firstDate,
secondDate);
 System.out.println("Number of days between " + firstDate + "
and " + secondDate + ": " + daysBetween);
 }
}
```

OUTPUT:



Task 7: Timezone

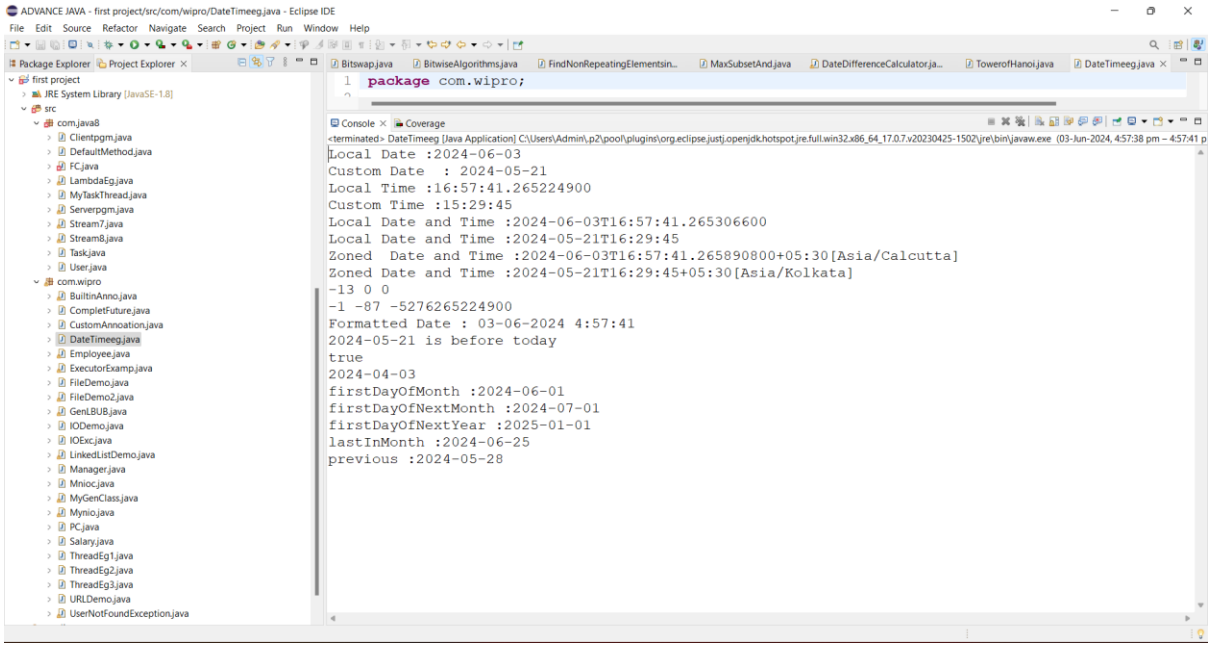
Create a timezone converter that takes a time in one timezone and converts it to another timezone

```
package com.wipro;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.LocalTime;
import java.time.Month;
import java.time.Period;
import java.time.Year;
import java.time.format.DateTimeFormatter;
import java.time.temporal.TemporalAdjuster;
```

```
import java.time.temporal.TemporalAdjusters;
import java.util.Calendar;
import java.util.TimeZone;
public class DateTime {
 public static void main(String[] args) {
 // TODO Auto-generated method stub
 LocalDate localdate =LocalDate.now();
 System.out.println(localdate);

LocalDate cdate=LocalDate.of(2024, Month.MAY, 21);
System.out.println(cdate);
LocalTime tt= LocalTime.now();
System.out.println(tt);
LocalDateTime lt= LocalDateTime.now();
System.out.println(lt);
TimeZone zone = TimeZone.getTimeZone("Asia/Kolkata");
System.out.println("The Offset value of TimeZone: " +
zone.getOffset(Calendar.ZONE_OFFSET));
 Period p=Period.between(localdate, cdate);
 System.out.println(p);

DateTimeFormatter formatter =
DateTimeFormatter.ofPattern("HH:mm:ss");
 String formattedTime = tt.format(formatter);
 System.out.println(formattedTime);


 DateTimeFormatter formatter1 =
DateTimeFormatter.ofPattern("dd/MM/yyyy HH:mm:ss");
```

```
 String formattedDateTime = lt.format(formatter1);
 System.out.println(formattedDateTime);

 int year=2024;
 System.out.println(Year.isLeap(year));

 //TemporalAdjuster class in java
 System.out.println("first day of
month"+cdate.with(TemporalAdjusters.firstDayOfMonth()));
 System.out.println("first day of next
month"+cdate.with(TemporalAdjusters.firstDayOfNextMonth()));
 System.out.println("first day of next
year"+cdate.with(TemporalAdjusters.firstDayOfNextYear()));

 }
// public static void checkdade(LocalDate id)
// {
//
// LocalDate today=LocalDate.now();
// if()
// System.out.println(id + "is before today");
// }
 }
```

OUTPUT:


ADVANCE JAVA - first project/src/com/wipro/DateTimeeg.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Project Explorer
first project
JRE System Library [JavaSE-1.8]
src
com.java8
Clientpgm.java
DefaultMethod.java
FC.java
LambdaEg.java
MyTaskThread.java
Serverpgm.java
Stream7.java
Stream8.java
Task.java
User.java
com.wipro
BuiltinAnno.java
CompletFuture.java
CustomAnnoation.java
DateTimeeg.java
Employee.java
ExecutorExamp.java
FileDemo.java
FileDemo2.java
GenLBUB.java
IODemo.java
IOExc.java
LinkedListDemo.java
Manager.java
Mniocj.java
MyGenClass.java
Mynio.java
PC.java
Salary.java
ThreadEg1.java
ThreadEg2.java
ThreadEg3.java
URLDemo.java
UserNotFoundException.java
Bitswap.java
BitwiseAlgorithms.java
FindNonRepeatingElementsin...
MaxSubsetAnd.java
DateDifferenceCalculator.ja...
TowerofHanoi.java
DateTimeeg.java
1 package com.wipro;
Console
Coverage
<terminated> DateTimeeg [Java Application] C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (03-Jun-2024, 4:57:38 pm – 4:57:41 p
Local Date :2024-06-03
Custom Date : 2024-05-21
Local Time :16:57:41.265224900
Custom Time :15:29:45
Local Date and Time :2024-06-03T16:57:41.265306600
Local Date and Time :2024-05-21T16:29:45
Zoned Date and Time :2024-06-03T16:57:41.265890800+05:30[Asia/Calcutta]
Zoned Date and Time :2024-05-21T16:29:45+05:30[Asia/Kolkata]
-13 0 0
-1 -87 -5276265224900
Formatted Date : 03-06-2024 4:57:41
2024-05-21 is before today
true
2024-04-03
firstDayOfMonth :2024-06-01
firstDayOfNextMonth :2024-07-01
firstDayOfNextYear :2025-01-01
lastInMonth :2024-06-25
previous :2024-05-28