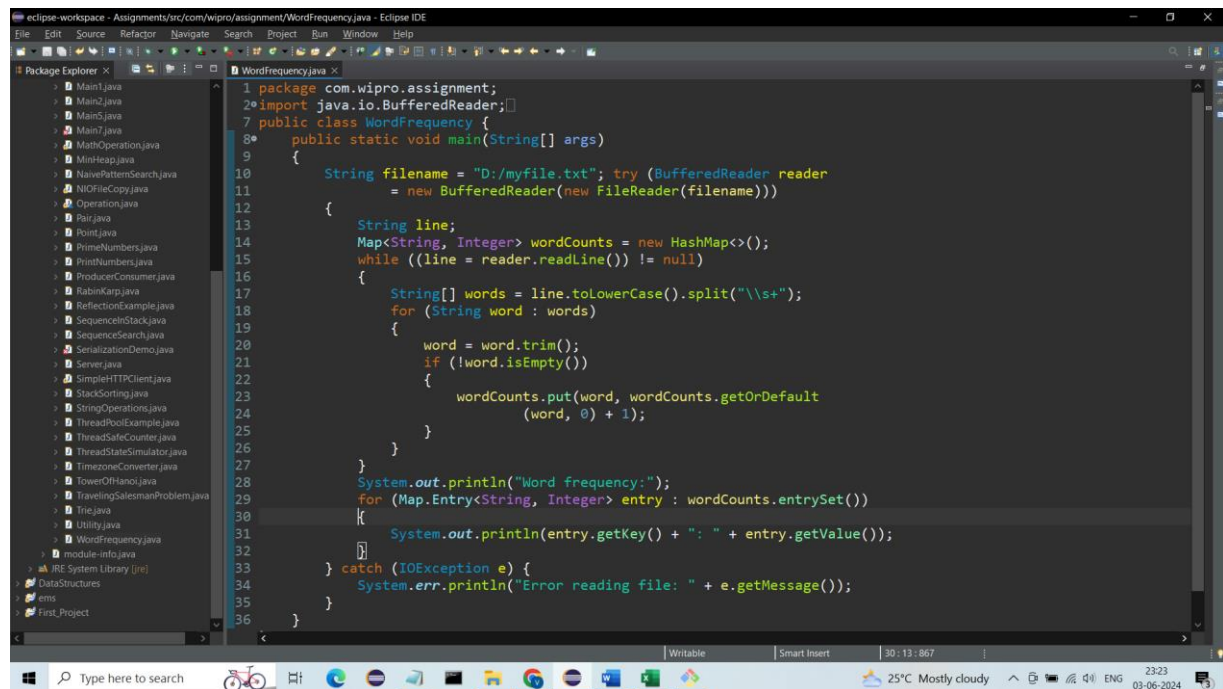


Task 1: Java IO Basics

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

Program: -

The screenshot shows the Eclipse IDE with a project named 'wipro' and a package 'com.wipro.assignment'. The file 'WordFrequency.java' is open in the editor. The code is as follows:

```
1 package com.wipro.assignment;
2 import java.io.BufferedReader;
3 import java.io.FileReader;
4 import java.io.IOException;
5 import java.util.HashMap;
6 import java.util.Map;
7 public class WordFrequency {
8     public static void main(String[] args)
9     {
10         String filename = "D:/myfile.txt"; try (BufferedReader reader
11             = new BufferedReader(new FileReader(filename)))
12         {
13             String line;
14             Map<String, Integer> wordCounts = new HashMap<>();
15             while ((line = reader.readLine()) != null)
16             {
17                 String[] words = line.toLowerCase().split("\\s+");
18                 for (String word : words)
19                 {
20                     word = word.trim();
21                     if (!word.isEmpty())
22                     {
23                         wordCounts.put(word, wordCounts.getDefault
24                             (word, 0) + 1);
25                     }
26                 }
27             }
28             System.out.println("Word frequency:");
29             for (Map.Entry<String, Integer> entry : wordCounts.entrySet())
30             {
31                 System.out.println(entry.getKey() + " : " + entry.getValue());
32             }
33         } catch (IOException e) {
34             System.err.println("Error reading file: " + e.getMessage());
35         }
36     }
37 }
```

The IDE interface includes a Package Explorer on the left, a Run and Debug console on the right, and a status bar at the bottom showing system information like temperature and date.

Output: -

The screenshot shows the Eclipse IDE with a project named 'com.wipro.assignment'. The 'WordFrequency.java' file is open, displaying the following code:

```
1 package com.wipro.assignment;
2 import java.io.BufferedReader;
3
4 public class WordFrequency {
5     public static void main(String[] args)
6     {
7         String filename = "D:/myfile.txt"; try (BufferedReader reader
8             = new BufferedReader(new FileReader(filename)))
9         {
10             String line;
11             Map<String, Integer> wordCounts = new HashMap<>();
12             while ((line = reader.readLine()) != null)
13             {
14                 String[] words = line.toLowerCase().split("\\s+");
15                 for (String word : words)
16                 {
17                     word = word.trim();
18                     if (!word.isEmpty())
19                     {
20                         wordCounts.put(word, wordCounts.getOrDefault
21                             (word, 0) + 1);
22                     }
23                 }
24             }
25             System.out.println("Word frequency:");
26             for (Map.Entry<String, Integer> entry : wordCounts.entrySet())
27             {
28                 System.out.println(entry.getKey() + ": " + entry.getValue());
29             }
30         }
31         catch (IOException e) {
32             System.err.println("Error reading file: " + e.getMessage());
33         }
34     }
35 }
```

The console output shows the word frequency results:

```
Word frequency:
industry's: 1
been: 1
software: 1
release: 1
evolved: 1
infancy.: 1
type: 2
when: 2
text,: 1
dummy: 2
lorem: 7
english.: 1
accident,: 1
model: 1
text: 2
1960s: 1
publishing: 2
years,: 1
using: 2
still: 1
in: 2
containing: 1
printer: 1
is: 4
it: 6
survived: 1
uncovered: 1
packages: 1
an: 1
typesetting,: 1
```

Task 2: Serialization and Deserialization

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

Custom Class (Employee):

The screenshot shows the Eclipse IDE with a project named 'com.wipro.assignment'. The 'Employee.java' file is open, displaying the following code:

```
1 package com.wipro.assignment;
2 import java.io.Serializable;
3
4 public class Employee implements Serializable
5 {
6     private String name;
7     private int id;
8
9     public Employee(String name, int id)
10    {
11        this.name = name;
12        this.id = id;
13    }
14 }
```

Serialization: -

```

public static void serializeObject(Employee employee,
    String fileName)
    throws IOException {
    FileOutputStream fileOut = new FileOutputStream(fileName);
    ObjectOutputStream out = new ObjectOutputStream(fileOut);
    out.writeObject(employee);
    out.close();
    fileOut.close();
    System.out.println("Serialized employee object to file: "
        + fileName);
}

```

Deserialization: -

```

public static Employee deserializeObject(String fileName) throws IOException,
    ClassNotFoundException {
    FileInputStream fileIn = new FileInputStream(fileName);
    ObjectInputStream in = new ObjectInputStream(fileIn);
    Employee employee = (Employee) in.readObject();
    in.close();
    fileIn.close();
    return employee;
}

```

Output: -

The screenshot shows the Eclipse IDE with the following components:

- Package Explorer:** Shows a project named 'com.wipro.assignment' with various Java files, including 'SerializationDemo.java'.
- Editor:** Displays the code for 'SerializationDemo.java', which includes the `serializeObject` and `deserializeObject` methods, and a `main` method that demonstrates the process. The `main` method creates an `Employee` object, serializes it to 'employee.ser', and then deserializes it back.
- Console:** Shows the output of the program:

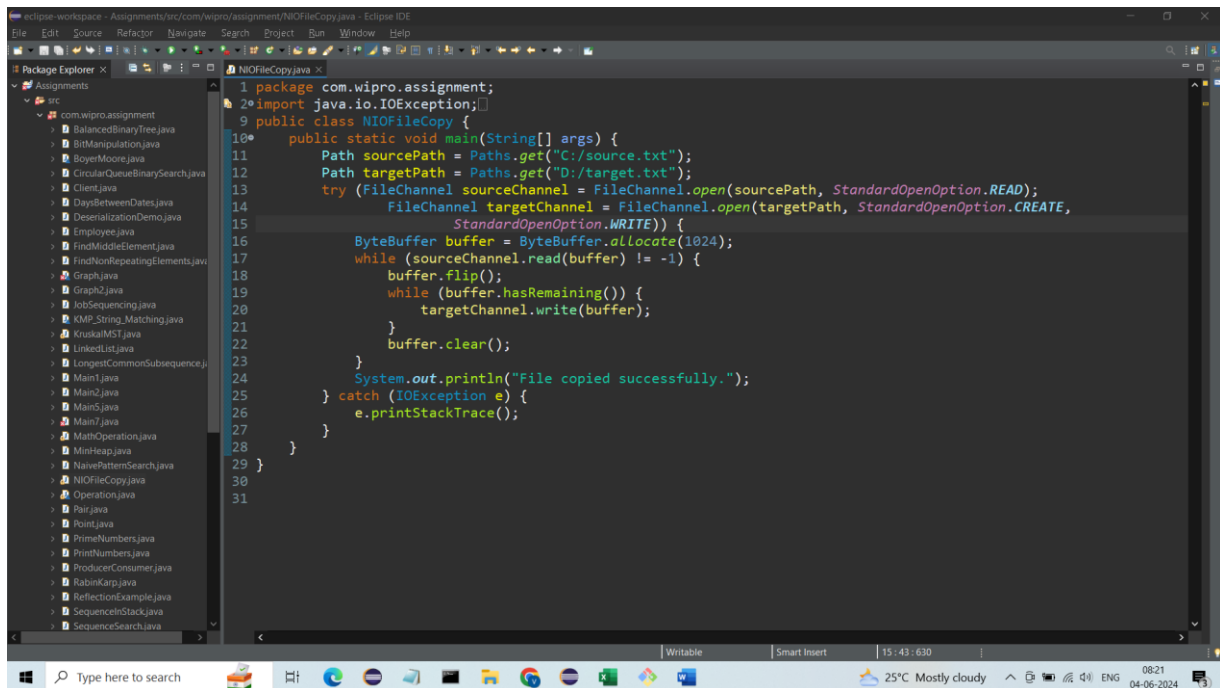

```

Serialized employee object to file: employee.ser
Deserialized employee object: null, null
      
```

Task 3: New IO (NIO)

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

Program: -



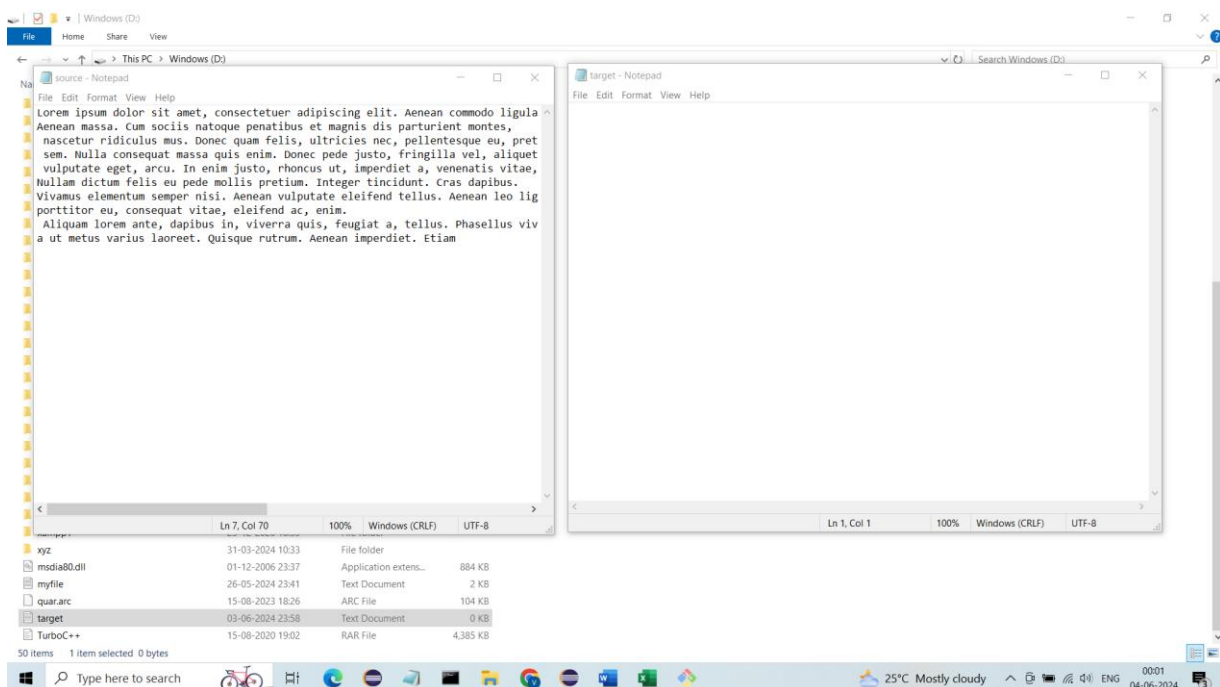
The screenshot shows the Eclipse IDE with a project named 'com.wipro.assignment'. The 'Package Explorer' on the left lists various Java files. The main editor displays the 'NIOFileCopy.java' file, which contains the following code:

```
1 package com.wipro.assignment;
2 import java.io.IOException;
3
4 public class NIOFileCopy {
5     public static void main(String[] args) {
6         Path sourcePath = Paths.get("C:/source.txt");
7         Path targetPath = Paths.get("D:/target.txt");
8         try (FileChannel sourceChannel = FileChannel.open(sourcePath, StandardOpenOption.READ);
9             FileChannel targetChannel = FileChannel.open(targetPath, StandardOpenOption.CREATE,
10                 StandardOpenOption.WRITE)) {
11             ByteBuffer buffer = ByteBuffer.allocate(1024);
12             while (sourceChannel.read(buffer) != -1) {
13                 buffer.flip();
14                 while (buffer.hasRemaining()) {
15                     targetChannel.write(buffer);
16                 }
17                 buffer.clear();
18             }
19             System.out.println("File copied successfully.");
20         } catch (IOException e) {
21             e.printStackTrace();
22         }
23     }
24 }
```

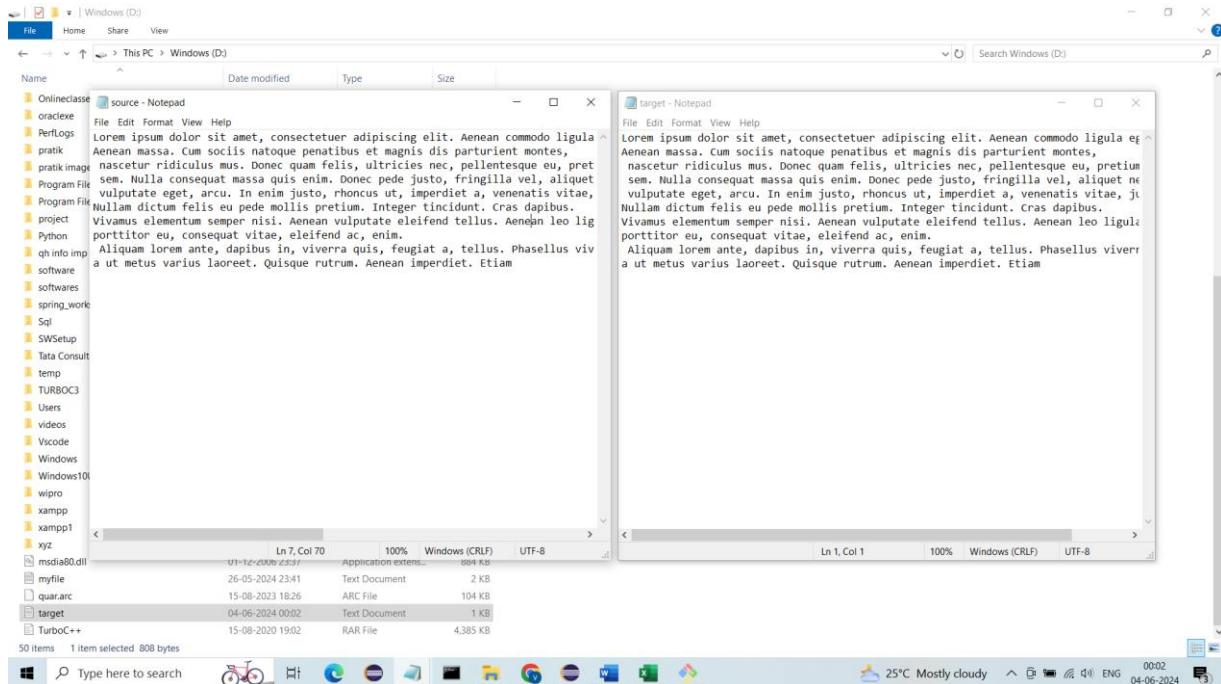
The IDE's status bar at the bottom shows 'Writable', 'Smart Insert', '15:43:630', and system information: '25°C Mostly cloudy', '08:21', and '04-06-2024'.

Output: -

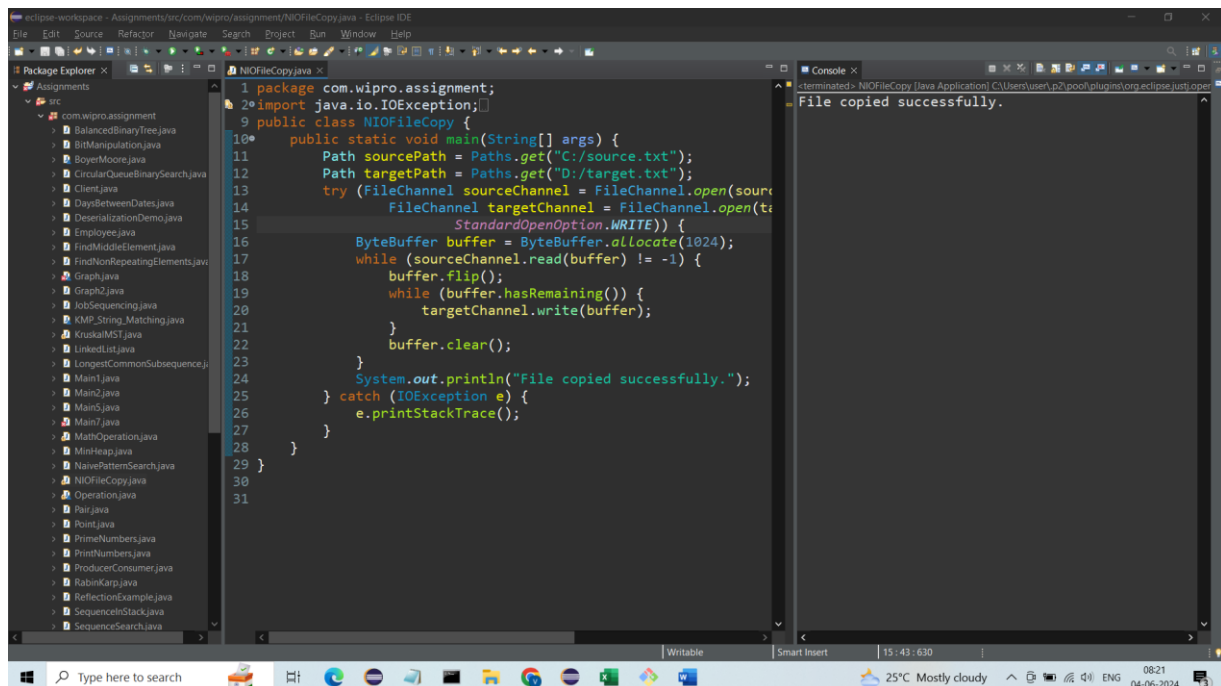
Before Copying: -



After Copying: -



Program Output: -

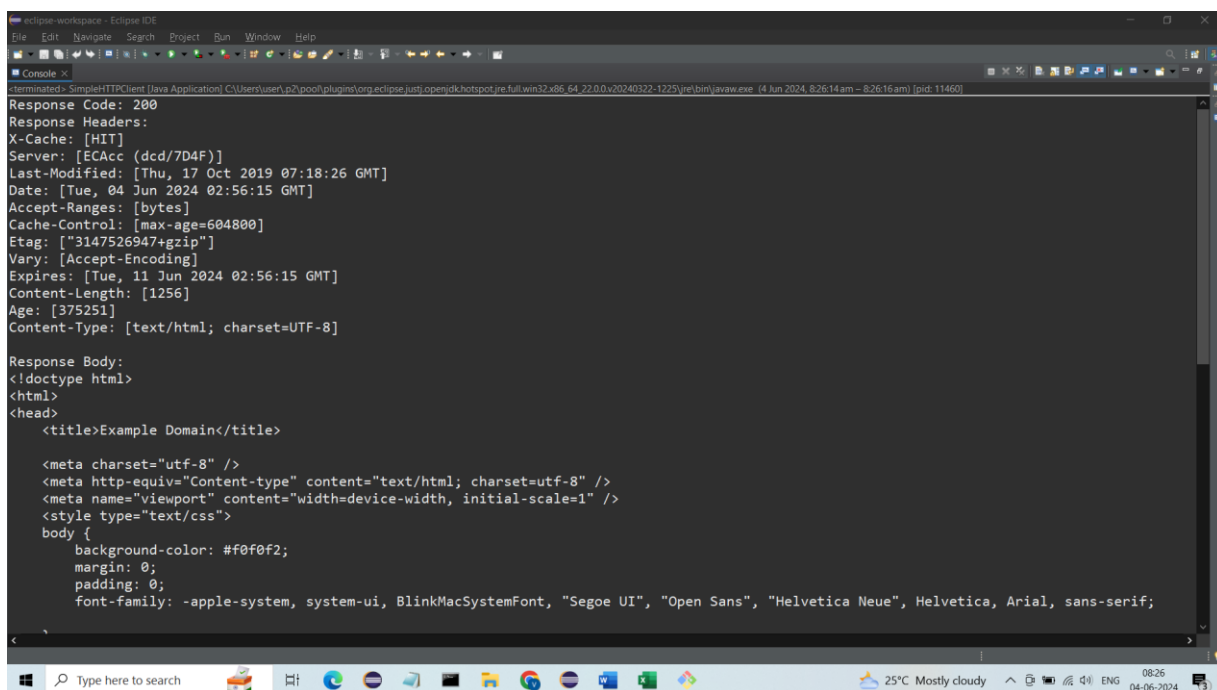


Task 4: Java Networking

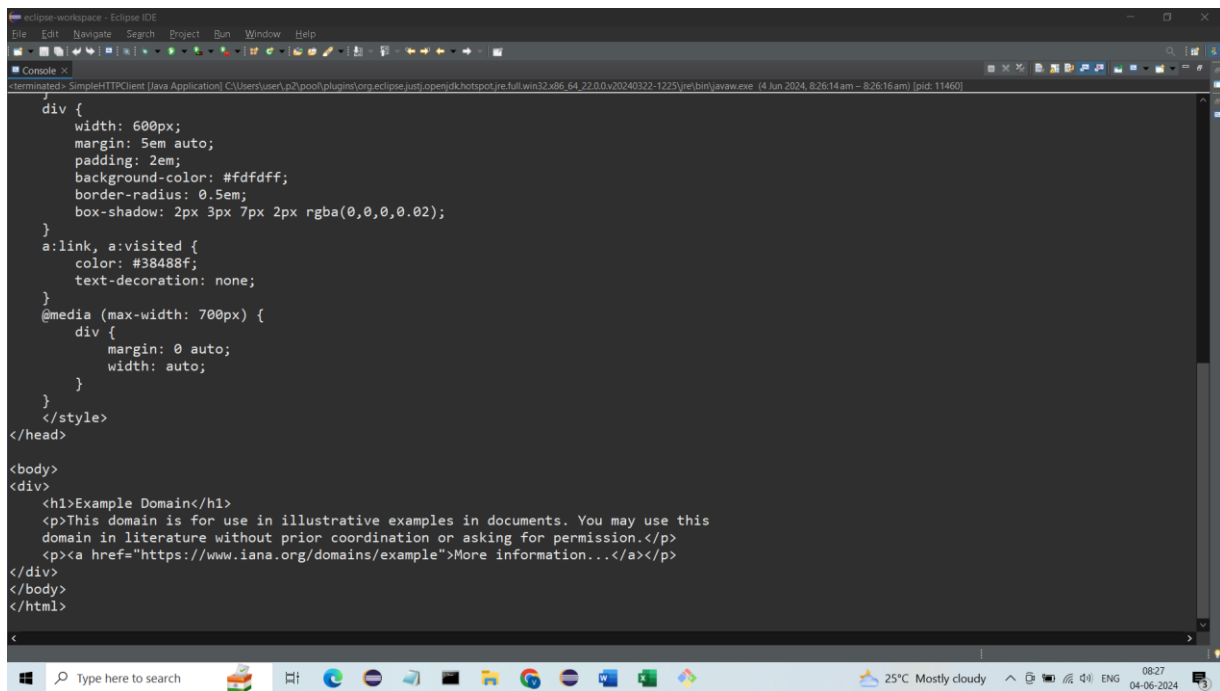
Write a simple HTTP client that connects to a URL, sends a request, and displays the response headers and body.

```
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) -> {  
                if (key != null) {  
                    System.out.println(key + ": " + value);  
                }  
            });  
            System.out.println("\nResponse 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();  
        }  
    }  
}
```

Output: -



```
<terminated> SimpleHTTPClient [Java Application] C:\Users\user\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.22.0.0\j220\bin\javaw.exe (4 Jun 2024, 8:26:16 am) [pid: 11460]  
Response Code: 200  
Response Headers:  
X-Cache: [HIT]  
Server: [ECCAcc (dcd/7D4F)]  
Last-Modified: [Thu, 17 Oct 2019 07:18:26 GMT]  
Date: [Tue, 04 Jun 2024 02:56:15 GMT]  
Accept-Ranges: [bytes]  
Cache-Control: [max-age=604800]  
Etag: ["3147526947+gzip"]  
Vary: [Accept-Encoding]  
Expires: [Tue, 11 Jun 2024 02:56:15 GMT]  
Content-Length: [1256]  
Age: [375251]  
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, Arial, sans-serif;
```



```
terminated SimpleHTTPClient [Java Application] C:\Users\user\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.22.0.0\20240322-1223\jre\bin\javaw.exe (4 Jun 2024, 8:26:14am) [pid: 11460]



## 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  $2 + 2$



Request class: -


```

```
Request.java × Server.java Response.java
1 package com.wipro.assignment;
2
3 import java.io.Serializable;
4
5 @SuppressWarnings("serial")
6 class Request implements Serializable {
7     private int num1;
8     private int num2;
9     private String operation;
10
11     public Request(int num1, int num2, String operation) {
12         this.num1 = num1;
13         this.num2 = num2;
14         this.operation = operation;
15     }
16
17     public int getNum1() {
18         return num1;
19     }
20
21     public int getNum2() {
22         return num2;
23     }
24
25     public String getOperation() {
26         return operation;
27     }
28
29 }
30
```

Response Class: -


```

package com.wipro.assignment;

import java.io.Serializable;

class Response implements Serializable {
    private int result;

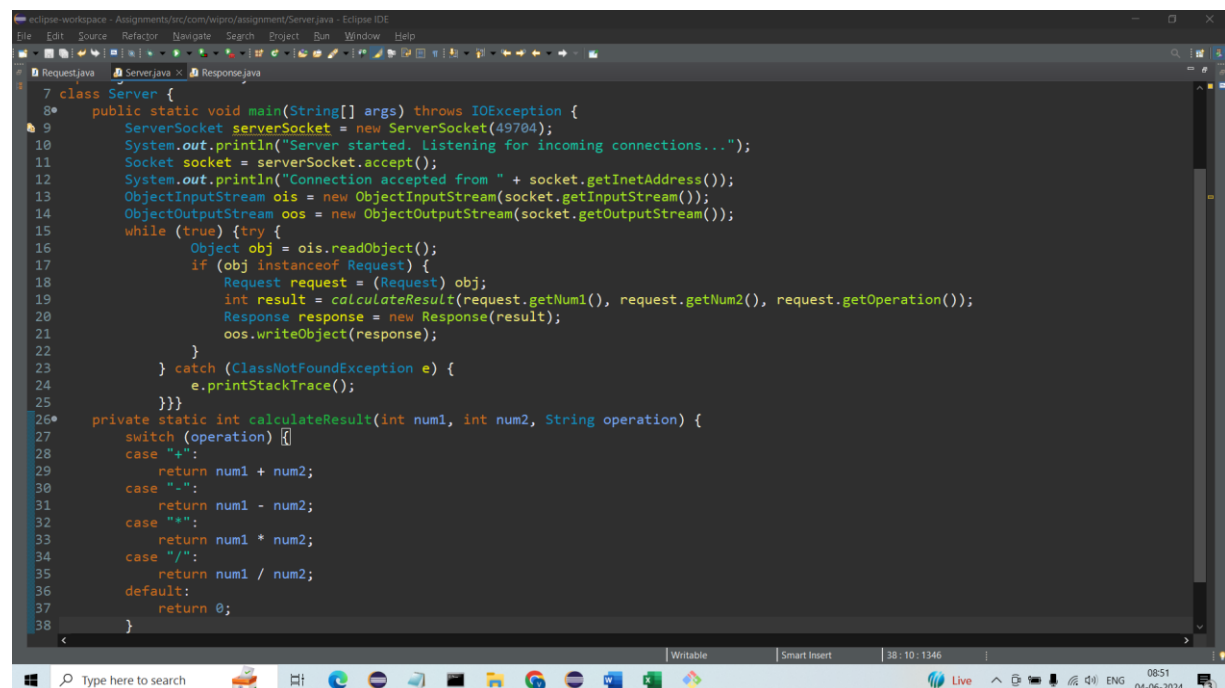
    • public Response(int result) {
        this.result = result;
    }

    • public int getResult() {
        return result;
    }

}

```

Server Class: -



```

7 class Server {
8     public static void main(String[] args) throws IOException {
9         ServerSocket serverSocket = new ServerSocket(49704);
10        System.out.println("Server started. Listening for incoming connections...");
11        Socket socket = serverSocket.accept();
12        System.out.println("Connection accepted from " + socket.getInetAddress());
13        ObjectInputStream ois = new ObjectInputStream(socket.getInputStream());
14        ObjectOutputStream oos = new ObjectOutputStream(socket.getOutputStream());
15        while (true) {try {
16            Object obj = ois.readObject();
17            if (obj instanceof Request) {
18                Request request = (Request) obj;
19                int result = calculateResult(request.getNum1(), request.getNum2(), request.getOperation());
20                Response response = new Response(result);
21                oos.writeObject(response);
22            }
23        } catch (ClassNotFoundException e) {
24            e.printStackTrace();
25        }
26    }
27    private static int calculateResult(int num1, int num2, String operation) {
28        switch (operation) {
29            case "+":
30                return num1 + num2;
31            case "-":
32                return num1 - num2;
33            case "*":
34                return num1 * num2;
35            case "/":
36                return num1 / num2;
37            default:
38                return 0;
39        }
40    }
41 }

```

Output: -

```
1 package com.wipro.connection;
2 import java.io.IOException;
3 import java.io.ObjectInputStream;
4 import java.io.ObjectOutputStream;
5 import java.net.ServerSocket;
6 import java.net.Socket;
7 class Server {
8     public static void main(String[] args) throws IOException {
9         ServerSocket serverSocket = new ServerSocket(49708);
10        System.out.println("Server started. Listening for incoming");
11        Socket socket = serverSocket.accept();
12        System.out.println("Connection accepted from " + socket.getInetAddress());
13        ObjectInputStream ois = new ObjectInputStream(socket.getInputStream());
14        ObjectOutputStream oos = new ObjectOutputStream(socket.getOutputStream());
15        while (true) {try {
16            Object obj = ois.readObject();
17            if (obj instanceof Request) {
18                Request request = (Request) obj;
19                int result = calculateResult(request.getNum1(), request.getNum2(), request.getOperation());
20                Response response = new Response(result);
21                oos.writeObject(response);
22            } catch (ClassNotFoundException e) {
23                e.printStackTrace();
24            }
25        }
26    }
27    private static int calculateResult(int num1, int num2, String operation) {
28        switch (operation) {
29            case "+":
30                return num1 + num2;
31            case "-":
32                return num1 - num2;
33        }
34    }
35 }
```

Server started. Listening for incoming connections...

Proto	Local Address	Foreign Address	State
TCP	127.0.0.1:49704	DESKTOP-73BD0TJ:49705	ESTABLISHED

Task 6: Java 8 Date and Time API

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

Program: -

```

DaysBetweenDates.java x
1 package com.wipro.assignment;
2 import java.time.LocalDate;
6 public class DaysBetweenDates
7 {
8     public static void main(String[] args)
9     {
10         Scanner scanner = new Scanner(System.in);
11         System.out.print("Enter the first date (yyyy-MM-dd): ");
12         String firstDateString = scanner.nextLine();
13         LocalDate firstDate = LocalDate.parse(firstDateString,
14             DateTimeFormatter.ISO_LOCAL_DATE);
15         System.out.print("Enter the second date (yyyy-MM-dd): ");
16         String secondDateString = scanner.nextLine();
17         LocalDate secondDate = LocalDate.parse(secondDateString,
18             DateTimeFormatter.ISO_LOCAL_DATE);
19         long daysBetween = Math.abs(ChronoUnit.DAYS.between(firstDate,
20             secondDate));
21         System.out.println("Number of days between " + firstDate + " and "
22             + secondDate + " is: " + daysBetween);
23         scanner.close();
24     }
25 }

```

Output: -

The screenshot shows the Eclipse IDE with the 'DaysBetweenDates.java' file open. The code is identical to the one shown in the previous block. The 'Console' window on the right displays the following output:

```

Enter the first date (yyyy-MM-dd): 2000-10-06
Enter the second date (yyyy-MM-dd): 2024-10-06
Number of days between 2000-10-06 and 2024-10-06 is: 8766

```

The IDE interface includes a menu bar (File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help), a toolbar, and a status bar at the bottom showing '25°C Cloudy' and the date '04-06-2024'.

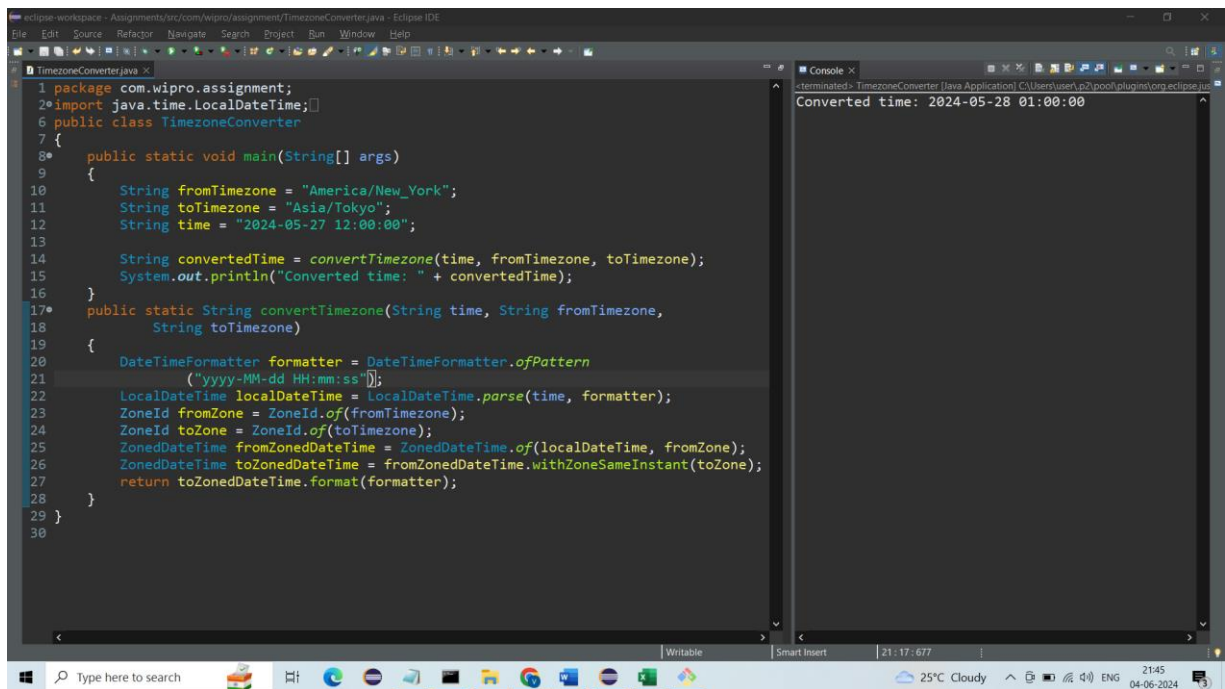
Task 7: Timezone

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

Function: -

```
}  
public static String convertTimezone(String time, String fromTimezone, String toTimezone)  
{  
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");  
    LocalDateTime localDateTime = LocalDateTime.parse(time, formatter);  
    ZoneId fromZone = ZoneId.of(fromTimezone);  
    ZoneId toZone = ZoneId.of(toTimezone);  
    ZonedDateTime fromZonedDateTime = ZonedDateTime.of(localDateTime, fromZone);  
    ZonedDateTime toZonedDateTime = fromZonedDateTime.withZoneSameInstant(toZone);  
    return toZonedDateTime.format(formatter);  
}
```

Output: -



The screenshot shows the Eclipse IDE with the file `TimezoneConverter.java` open. The code defines a `convertTimezone` method and a `main` method. The `main` method sets `fromTimezone` to "America/New_York", `toTimezone` to "Asia/Tokyo", and `time` to "2024-05-27 12:00:00". It then calls `convertTimezone` and prints the result. The console output shows "Converted time: 2024-05-28 01:00:00".

```
1 package com.wipro.assignment;  
2 import java.time.LocalDateTime;  
6 public class TimezoneConverter  
7 {  
8     public static void main(String[] args)  
9     {  
10         String fromTimezone = "America/New_York";  
11         String toTimezone = "Asia/Tokyo";  
12         String time = "2024-05-27 12:00:00";  
13  
14         String convertedTime = convertTimezone(time, fromTimezone, toTimezone);  
15         System.out.println("Converted time: " + convertedTime);  
16     }  
17     public static String convertTimezone(String time, String fromTimezone,  
18         String toTimezone)  
19     {  
20         DateTimeFormatter formatter = DateTimeFormatter.ofPattern  
21             ("yyyy-MM-dd HH:mm:ss");  
22         LocalDateTime localDateTime = LocalDateTime.parse(time, formatter);  
23         ZoneId fromZone = ZoneId.of(fromTimezone);  
24         ZoneId toZone = ZoneId.of(toTimezone);  
25         ZonedDateTime fromZonedDateTime = ZonedDateTime.of(localDateTime, fromZone);  
26         ZonedDateTime toZonedDateTime = fromZonedDateTime.withZoneSameInstant(toZone);  
27         return toZonedDateTime.format(formatter);  
28     }  
29 }  
30
```

Console Output:
Converted time: 2024-05-28 01:00:00