Project 2

Louis Chang (hungyic), hungyic@andrew.cmu.edu

Task 0

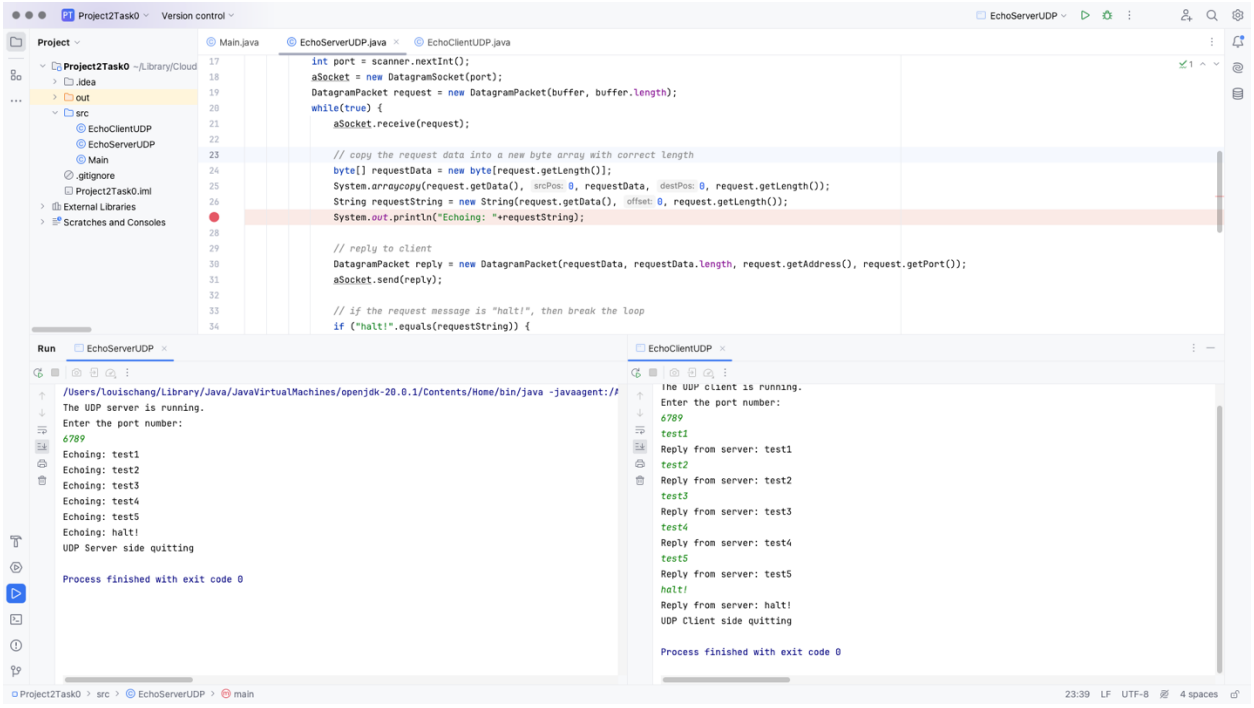
EchoServerUDP.java

```
import java.net.*;
import java.io.*;
public class EchoServerUDP{
    public static void main(String args[]){
        DatagramSocket aSocket = null;
        byte[] buffer = new byte[1000];
        try {
            System.out.println("The UDP server is running.");
            aSocket = new DatagramSocket(6789);
            DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
            while(true) {
                aSocket.receive(request);
                // copy the request data into a new byte array with
correct length
                byte[] requestData = new byte[request.getLength()];
                System.arraycopy(request.getData(), 0, requestData, 0,
request.getLength());
                String requestString = new String(request.getData(), 0,
request.getLength());
                System.out.println("Echoing: "+requestString);
                // reply to client
                DatagramPacket reply = new DatagramPacket(requestData,
requestData.length, request.getAddress(), request.getPort());
                aSocket.send(reply);
                if ("halt!".equals(requestString)) {
                    break;
                }
            }
        } catch (SocketException e) {
            System.out.println("Socket: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("IO: " + e.getMessage());
        } finally {
            if(aSocket != null) {
                System.out.println("UDP Server side quitting");
                aSocket.close();
            }
        }
    }
}
```

EchoClientUDP.java

```
import java.net.*;
import java.io.*;
public class EchoClientUDP{
    public static void main(String args[]){
        // args give message contents and server hostname
        DatagramSocket aSocket = null;
        try {
            System.out.println("The UDP client is running.");
            InetAddress aHost = InetAddress.getByName("localhost");
            int serverPort = 6789;
            aSocket = new DatagramSocket();
            String nextLine;
            BufferedReader typed = new BufferedReader(new
InputStreamReader(System.in));
            while ((nextLine = typed.readLine()) != null) {
                byte[] m = nextLine.getBytes();
                DatagramPacket request = new DatagramPacket(m, m.length,
aHost, serverPort);
                aSocket.send(request);
                byte[] buffer = new byte[1000];
                DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
                aSocket.receive(reply);
                // copy the reply data into a new byte array with correct
length
                byte[] replyData = new byte[reply.getLength()];
                System.arraycopy(reply.getData(), 0, replyData, 0,
reply.getLength());
                String replyMsg = new String(replyData);
                System.out.println("Reply from server: " + replyMsg);
                if ("halt!".equals(replyMsg)) {
                    break;
                }
            }

        } catch (SocketException e) {
            System.out.println("Socket Exception: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("IO Exception: " + e.getMessage());
        } finally {
            if(aSocket != null) {
                System.out.println("UDP Client side quitting");
                aSocket.close();
            }
        }
    }
}
```

Project2Task0ClientConsole


Project2Task0
Version control
EchoServerUDP
Project
Main.java
EchoServerUDP.java
EchoClientUDP.java
Project2Task0 ~/Library/Cloud
.idea
out
src
EchoClientUDP
EchoServerUDP
Main
.gitignore
Project2Task0.iml
External Libraries
Scratches and Consoles
int port = scanner.nextInt();
aSocket = new DatagramSocket(port);
DatagramPacket request = new DatagramPacket(buffer, buffer.length);
while(true) {
aSocket.receive(request);
// copy the request data into a new byte array with correct length
byte[] requestData = new byte[request.getLength()];
System.arraycopy(request.getData(), srcPos: 0, requestData, destPos: 0, request.getLength());
String requestString = new String(request.getData(), offset: 0, request.getLength());
System.out.println("Echoing: "+requestString);
// reply to client
DatagramPacket reply = new DatagramPacket(requestData, requestData.length, request.getAddress(), request.getPort());
aSocket.send(reply);
// if the request message is "halt!", then break the loop
if ("halt!".equals(requestString)) {
Run
EchoServerUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
The UDP server is running.
Enter the port number:
6789
Echoing: test1
Echoing: test2
Echoing: test3
Echoing: test4
Echoing: test5
Echoing: halt!
UDP Server side quitting
Process finished with exit code 0
EchoClientUDP
The UDP client is running.
Enter the port number:
6789
test1
Reply from server: test1
test2
Reply from server: test2
test3
Reply from server: test3
test4
Reply from server: test4
test5
Reply from server: test5
halt!
Reply from server: halt!
UDP Client side quitting
Process finished with exit code 0
Project2Task0 > src > EchoServerUDP > main
23:39
LF
UTF-8
4 spaces

Task 1

EchoServerUDP.java

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;
import java.util.Scanner;

/**
 * Author: Louis Chang (hungyic)
 * Last Modified: 02/22/2024
 */
public class EchoServerUDP {
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        DatagramSocket aSocket = null;
        byte[] buffer = new byte[1000];
        try {
            System.out.println("The UDP server is running.");
            System.out.println("Please enter the port number:");
            int port = scanner.nextInt();
            aSocket = new DatagramSocket(port);
            DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
            while(true) {
                aSocket.receive(request);
                // copy the request data into a new byte array with
correct length
                byte[] requestData = new byte[request.getLength()];
                System.arraycopy(request.getData(), 0, requestData, 0,
request.getLength());
                String requestString = new String(request.getData(), 0,
request.getLength());
                System.out.println("Echoing: "+requestString);
                // reply to client
                DatagramPacket reply = new DatagramPacket(requestData,
requestData.length, request.getAddress(), request.getPort());
                aSocket.send(reply);
                // if the request message is "halt!", then break the loop
                if ("halt!".equals(requestString)) {
                    break;
                }
            }
        } catch (SocketException e) {
            System.out.println("Socket: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("IO: " + e.getMessage());
        } finally {
            if(aSocket != null) {
                System.out.println("UDP Server side quitting");
                aSocket.close();
            }
```

```
        }
    }
}
```

EchoClientUDP.java

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
import java.util.Scanner;

/**
 * Author: Louis Chang (hungyic)
 * Last Modified: 02/22/2024
 */
public class EchoClientUDP{
    public static void main(String[] args){
        // args give message contents and server hostname
        DatagramSocket aSocket = null;
        Scanner scanner = new Scanner(System.in);
        try {
            System.out.println("The UDP client is running.");
            System.out.println("Please enter the port number:");
            int port = scanner.nextInt();
            InetAddress aHost = InetAddress.getByName("localhost");
            aSocket = new DatagramSocket();
            String nextLine;
            BufferedReader typed = new BufferedReader(new
InputStreamReader(System.in));
            while ((nextLine = typed.readLine()) != null) {
                byte[] m = nextLine.getBytes();
                DatagramPacket request = new DatagramPacket(m, m.length,
aHost, port);
                // send the message to the server
                aSocket.send(request);
                byte[] buffer = new byte[1000];
                DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
                aSocket.receive(reply);
                // copy the reply data into a new byte array with correct
length
                byte[] replyData = new byte[reply.getLength()];
                System.arraycopy(reply.getData(), 0, replyData, 0,
reply.getLength());
                String replyMsg = new String(replyData);
                System.out.println("Reply from server: " + replyMsg);
                // if the reply message is "halt!", then break the loop
                if ("halt!".equals(replyMsg)) {
                    break;
```

```
                }
            }

        } catch (SocketException e) {
            System.out.println("Socket Exception: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("IO Exception: " + e.getMessage());
        } finally {
            if(aSocket != null) {
                System.out.println("UDP Client side quitting");
                aSocket.close();
            }
        }
    }
}
```

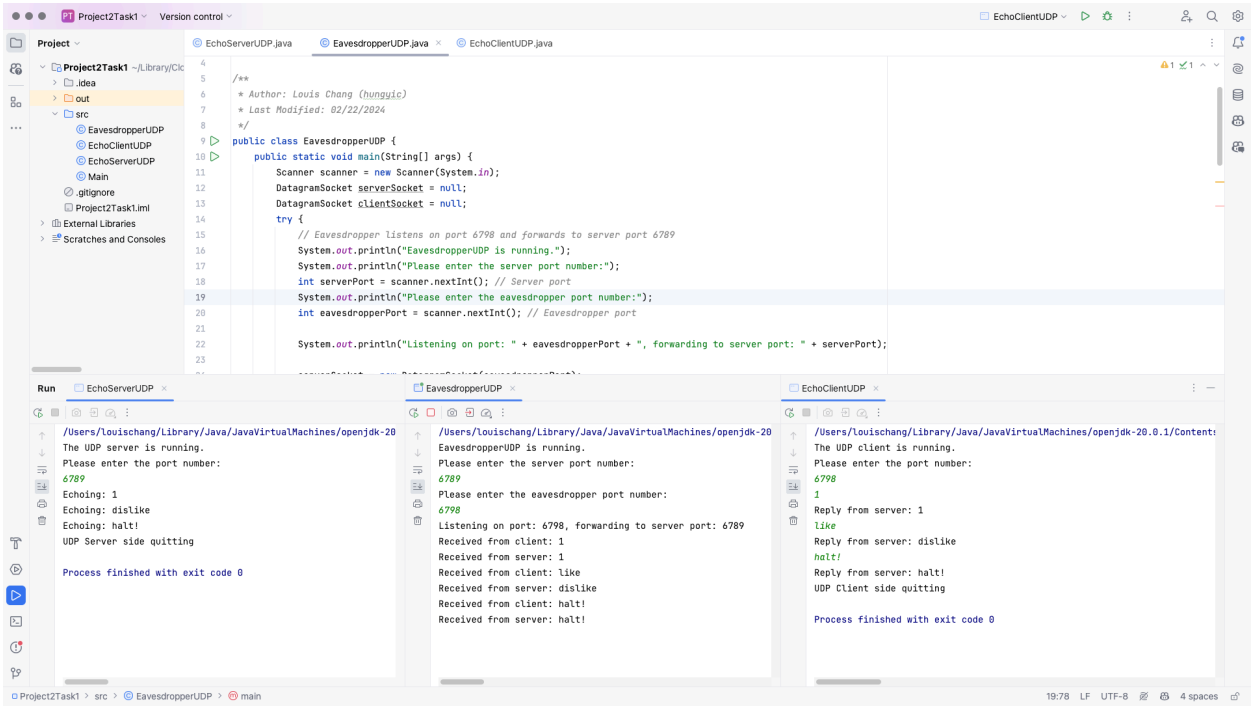
EavesdropperUDP.java

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

/**
 * Author: Louis Chang (hungyic)
 * Last Modified: 02/22/2024
 */
public class EavesdropperUDP {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        DatagramSocket serverSocket = null;
        DatagramSocket clientSocket = null;
        try {
            // Eavesdropper listens on port 6798 and forwards to server
port 6789
            System.out.println("EavesdropperUDP is running.");
            System.out.println("Please enter the server port number:");
            int serverPort = scanner.nextInt(); // Server port
            System.out.println("Please enter the eavesdropper port
number:");
            int eavesdropperPort = scanner.nextInt(); // Eavesdropper port

            System.out.println("Listening on port: " + eavesdropperPort +
", forwarding to server port: " + serverPort);

            serverSocket = new DatagramSocket(eavesdropperPort);
            clientSocket = new DatagramSocket();

            byte[] buffer = new byte[1000];
            while (true) {
                DatagramPacket clientPacket = new DatagramPacket(buffer,
buffer.length);
                serverSocket.receive(clientPacket);
```

```
                // message from client
                String message = new String(clientPacket.getData(), 0,
clientPacket.getLength());
                System.out.println("Received from client: " + message);

                // modify message if it contains "like" but not "dislike"
                if (message.contains("like")
&& !message.contains("dislike")) {
                    message = message.replaceFirst("like", "dislike");
                }

                // send modified message to server
                byte[] modifiedMessage = message.getBytes();
                InetAddress serverAddress =
InetAddress.getByName("localhost");
                DatagramPacket serverPacket = new
DatagramPacket(modifiedMessage, modifiedMessage.length, serverAddress,
serverPort);
                clientSocket.send(serverPacket);

                // receive response from server
                byte[] responseBuffer = new byte[1000];
                DatagramPacket responsePacket = new
DatagramPacket(responseBuffer, responseBuffer.length);
                clientSocket.receive(responsePacket);
                String response = new String(responsePacket.getData(), 0,
responsePacket.getLength());
                System.out.println("Received from server: " + response);

                // send response to client
                InetAddress clientAddress = clientPacket.getAddress();
                int clientPort = clientPacket.getPort();
                DatagramPacket replyPacket = new
DatagramPacket(responseBuffer, responsePacket.getLength(), clientAddress,
clientPort);
                serverSocket.send(replyPacket);
            }
        } catch (SocketException e) {
            System.out.println("Socket: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("IO: " + e.getMessage());
        } finally {
            if (serverSocket != null) serverSocket.close();
            if (clientSocket != null) clientSocket.close();
        }
    }
}
```

The screenshots below are about my client, server, and eavesdropper consoles. They show the results of general commands, replacing “like” as “dislike”, and “halt!” to terminate only client and server.

Project2Task1ThreeConsoles



Task 2

Project2Task2Client

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;

public class AddingClientUDP {
    private static int serverPort;

    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        System.out.println("The client is running.");
        System.out.println("Please enter server port: ");
        serverPort = scanner.nextInt();
        System.out.println();

        while (true) {
            String input = scanner.next();
            if ("halt!".equalsIgnoreCase(input)) {
                System.out.println("Client side quitting.");
                break;
            }
            try {
                int valueToAdd = Integer.parseInt(input);
                int sum = add(valueToAdd);
                System.out.println("The server returned " + sum + ".");
            } catch (IOException e) {
                System.out.println("IO Exception: " + e.getMessage());
            } catch (NumberFormatException e) {
                System.out.println("Please enter a valid integer or
'halt!' to exit.");
            }
        }
        scanner.close();
    }

    public static int add(int i) throws IOException {
        DatagramSocket aSocket = null;
        try {
            aSocket = new DatagramSocket();
            InetAddress aHost = InetAddress.getByName("localhost");
            byte[] sendData =
java.nio.ByteBuffer.allocate(4).putInt(i).array();
            DatagramPacket request = new DatagramPacket(sendData,
sendData.length, aHost, serverPort);
            aSocket.send(request);

            byte[] buffer = new byte[4];
            DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
```

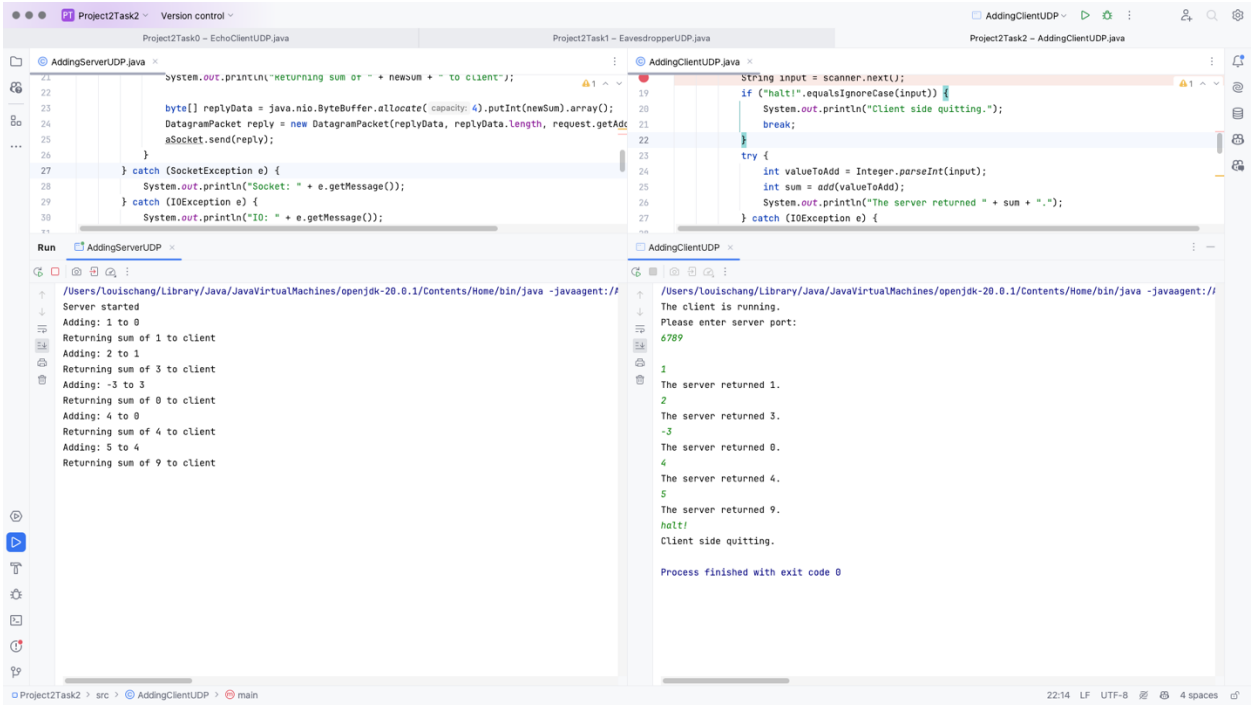
```
            aSocket.receive(reply);
            return java.nio.ByteBuffer.wrap(reply.getData()).getInt();
        } finally {
            if (aSocket != null) aSocket.close();
        }
    }
}
```

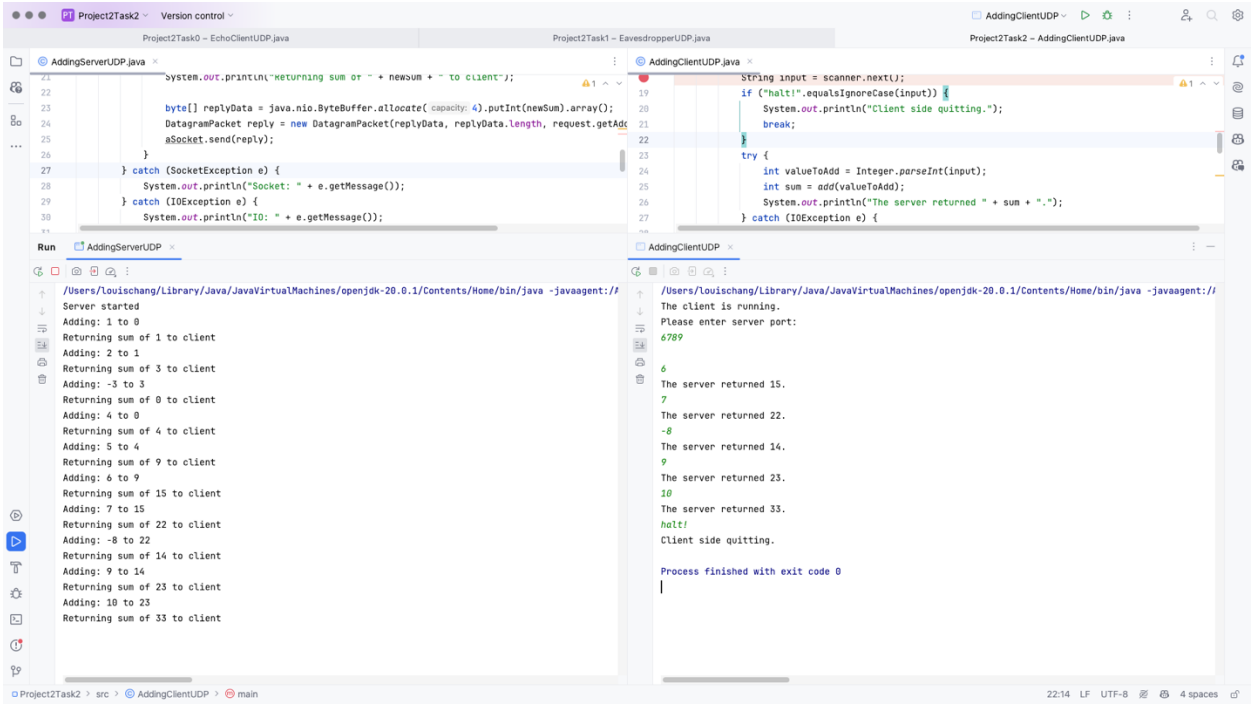
Project2Task2Server

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;

public class AddingServerUDP {
    private static int sum = 0;
    public static void main(String[] args){
        DatagramSocket aSocket = null;
        byte[] buffer = new byte[1000];
        try {
            System.out.println("Server started");
            aSocket = new DatagramSocket(6789);
            while (true) {
                DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
                aSocket.receive(request);
                // 直接從接收到的字節數組中解析整數值
                int receivedValue =
java.nio.ByteBuffer.wrap(request.getData()).getInt();
                int newSum = add(receivedValue);
                System.out.println("Adding: " + receivedValue + " to " +
(newSum - receivedValue));
                System.out.println("Returning sum of " + newSum + " to
client");

                byte[] replyData =
java.nio.ByteBuffer.allocate(4).putInt(newSum).array();
                DatagramPacket reply = new DatagramPacket(replyData,
replyData.length, request.getAddress(), request.getPort());
                aSocket.send(reply);
            }
        } catch (SocketException e) {
            System.out.println("Socket: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("IO: " + e.getMessage());
        } finally {
            if(aSocket != null) {
                System.out.println("UDP Server side quitting");
                aSocket.close();
            }
        }
```

```
    }
    private static int add(int value) {
        sum += value;
        return sum;
    }
}
```

Project2Task2ClientConsole




Project2Task2
Version control
AddingClientUDP
Project2Task0 – EchoClientUDP.java
Project2Task1 – EavesdropperUDP.java
Project2Task2 – AddingClientUDP.java
AddingServerUDP.java
System.out.println("Returning sum of " + newSum + " to client");
byte[] replyData = java.nio.ByteBuffer.allocate(capacity: 4).putInt(newSum).array();
DatagramPacket reply = new DatagramPacket(replyData, replyData.length, request.getAdd
aSocket.send(reply);
} catch (SocketException e) {
System.out.println("Socket: " + e.getMessage());
} catch (IOException e) {
System.out.println("IO: " + e.getMessage());
AddingClientUDP.java
String input = scanner.next();
if ("halt!".equalsIgnoreCase(input)) {
System.out.println("Client side quitting.");
break;
}
try {
int valueToAdd = Integer.parseInt(input);
int sum = add(valueToAdd);
System.out.println("The server returned " + sum + ".");
} catch (IOException e) {
Run
AddingServerUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
Server started
Adding: 1 to 0
Returning sum of 1 to client
Adding: 2 to 1
Returning sum of 3 to client
Adding: -3 to 3
Returning sum of 0 to client
Adding: 4 to 0
Returning sum of 4 to client
Adding: 5 to 4
Returning sum of 9 to client
Adding: 6 to 9
Returning sum of 15 to client
Adding: 7 to 15
Returning sum of 22 to client
Adding: -8 to 22
Returning sum of 14 to client
Adding: 9 to 14
Returning sum of 23 to client
Adding: 10 to 23
Returning sum of 33 to client
AddingClientUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
The client is running.
Please enter server port:
6789
6
The server returned 15.
7
The server returned 22.
-8
The server returned 14.
9
The server returned 23.
10
The server returned 33.
halt!
Client side quitting.
Process finished with exit code 0
Project2Task2 > src > AddingClientUDP > main
22:14 LF UTF-8 4 spaces

Task 3

Project2Task3Client

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.nio.ByteBuffer;
import java.util.Scanner;

public class RemoteVariableClientUDP {
    private static int serverPort;

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("The client is running.");
        System.out.println("Please enter server port: ");
        serverPort = scanner.nextInt();
        System.out.println();

        while (true) {
            System.out.println("1. Add a value to your sum.");
            System.out.println("2. Subtract a value from your sum.");
            System.out.println("3. Get your sum.");
            System.out.println("4. Exit client");
            int operation = scanner.nextInt();
            int value = 0;

            if (operation == 1 || operation == 2) {
                System.out.println("Enter the value to " + (operation ==
1 ? "add" : "subtract") + ": ");
                value = scanner.nextInt();
            } else if (operation == 3) {
                value = 0;
            } else if (operation == 4) {
                System.out.println("Client side quitting.");
                break;
            } else {
                System.out.println("Invalid option.");
                continue;
            }

            System.out.println("Enter your ID: ");
            int id = scanner.nextInt();

            try {
                int result = calculate(id, operation, value);
                System.out.println("The result is " + result + ".\n");
            } catch (IOException e) {
                throw new RuntimeException(e);
            }

        }
```

```
        scanner.close();
    }

    public static int calculate(int id, int operation, int value) throws
IOException {
        DatagramSocket aSocket = null;
        try {
            aSocket = new DatagramSocket();
            InetAddress aHost = InetAddress.getByName("localhost");
            ByteBuffer byteBuffer = ByteBuffer.allocate(1024);
            byteBuffer.putInt(id).putInt(operation).putInt(value);
            byte[] sendData = byteBuffer.array();
            DatagramPacket request = new DatagramPacket(sendData,
sendData.length, aHost, serverPort);
            aSocket.send(request);

            byte[] buffer = new byte[4];
            DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
            aSocket.receive(reply);
            return java.nio.ByteBuffer.wrap(reply.getData()).getInt();
        } finally {
            if (aSocket != null) aSocket.close();
        }
    }
}
```

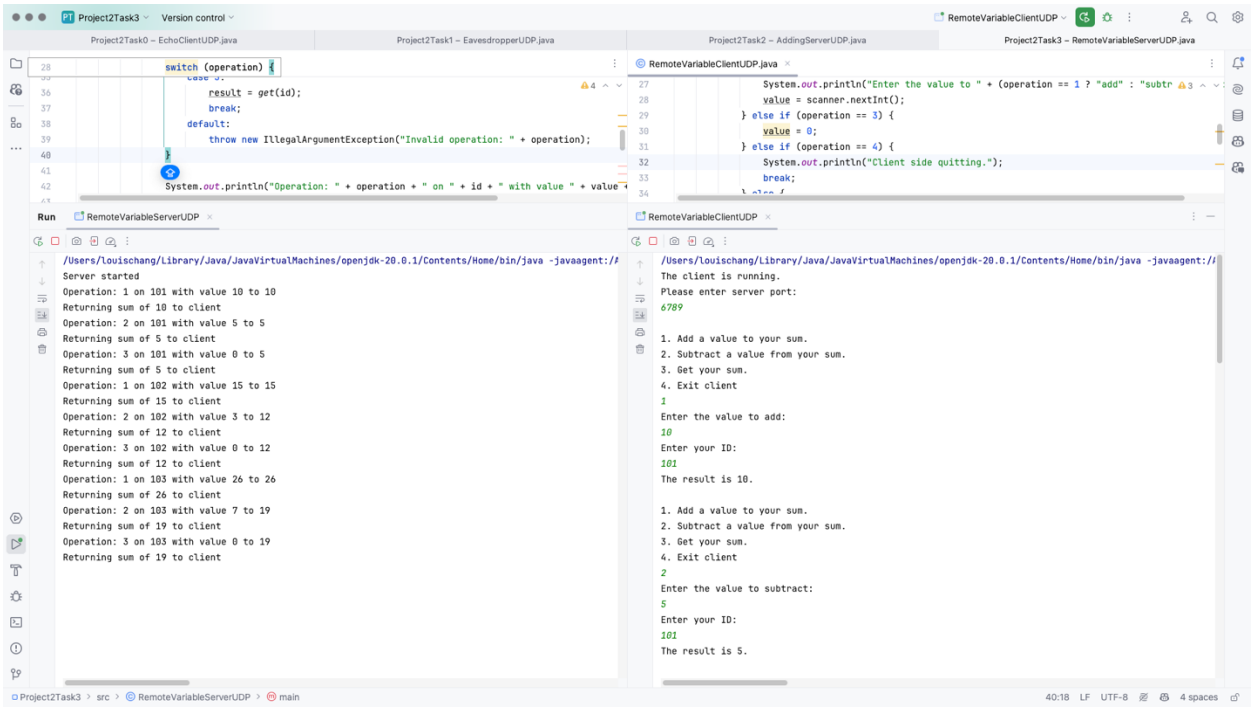
Project2Task3Server

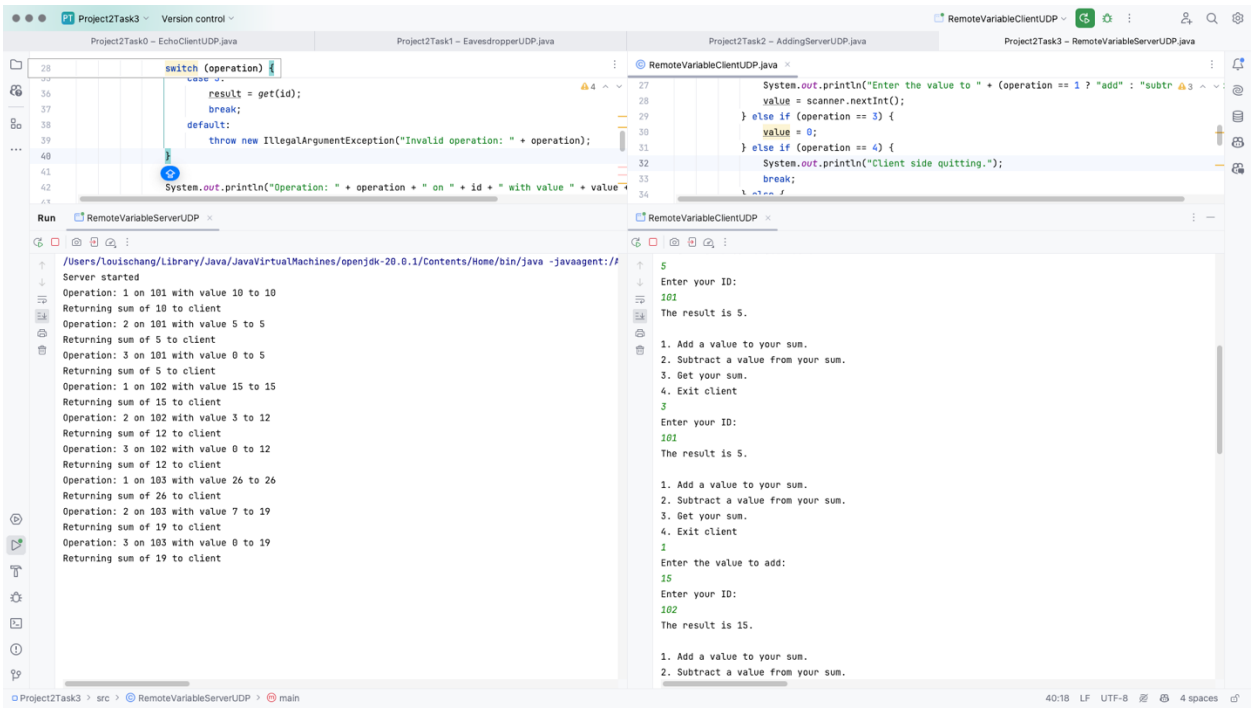
```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;
import java.nio.ByteBuffer;
import java.util.Map;
import java.util.TreeMap;

public class RemoteVariableServerUDP {
    private static final Map<Integer, Integer> map = new TreeMap<>();

    public static void main(String[] args){
        DatagramSocket aSocket = null;
        byte[] buffer = new byte[1000];
        try {
            System.out.println("Server started");
            aSocket = new DatagramSocket(6789);
            while (true) {
                DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
                aSocket.receive(request);
                ByteBuffer byteBuffer = ByteBuffer.wrap(request.getData(),
0, request.getLength());
```

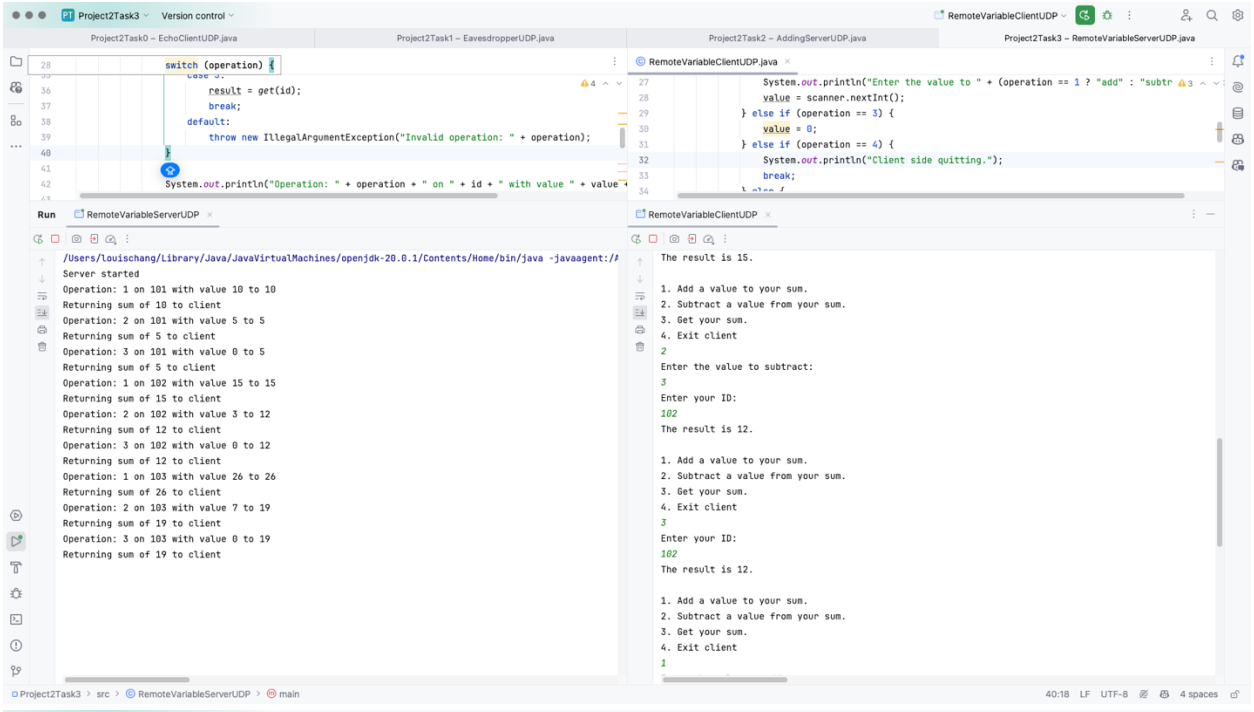
```
                int id = byteBuffer.getInt();
                int operation = byteBuffer.getInt();
                int value = byteBuffer.getInt();

                int result = 0;
                switch (operation) {
                    case 1:
                        result = add(id, value);
                        break;
                    case 2:
                        result = subtract(id, value);
                        break;
                    case 3:
                        result = get(id);
                        break;
                    default:
                        throw new IllegalArgumentException("Invalid
operation: " + operation);
                }

                System.out.println("Operation: " + operation + " on " + id
+ " with value " + value + " to " + result);
                System.out.println("Returning sum of " + result + " to
client");

                byte[] replyData =
java.nio.ByteBuffer.allocate(4).putInt(result).array();
                DatagramPacket reply = new DatagramPacket(replyData,
replyData.length, request.getAddress(), request.getPort());
                aSocket.send(reply);
            }
        } catch (SocketException e) {
            System.out.println("Socket: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("IO: " + e.getMessage());
        } finally {
            if(aSocket != null) {
                System.out.println("UDP Server side quitting");
                aSocket.close();
            }
        }
    }
    private static int add(int id, int value) {
        int sum = map.getOrDefault(id, 0);
        sum += value;
        map.put(id, sum);
        return sum;
    }
    private static int subtract(int id, int value) {
        int sum = map.getOrDefault(id, 0);
        sum -= value;
        map.put(id, sum);
        return sum;
    }
```

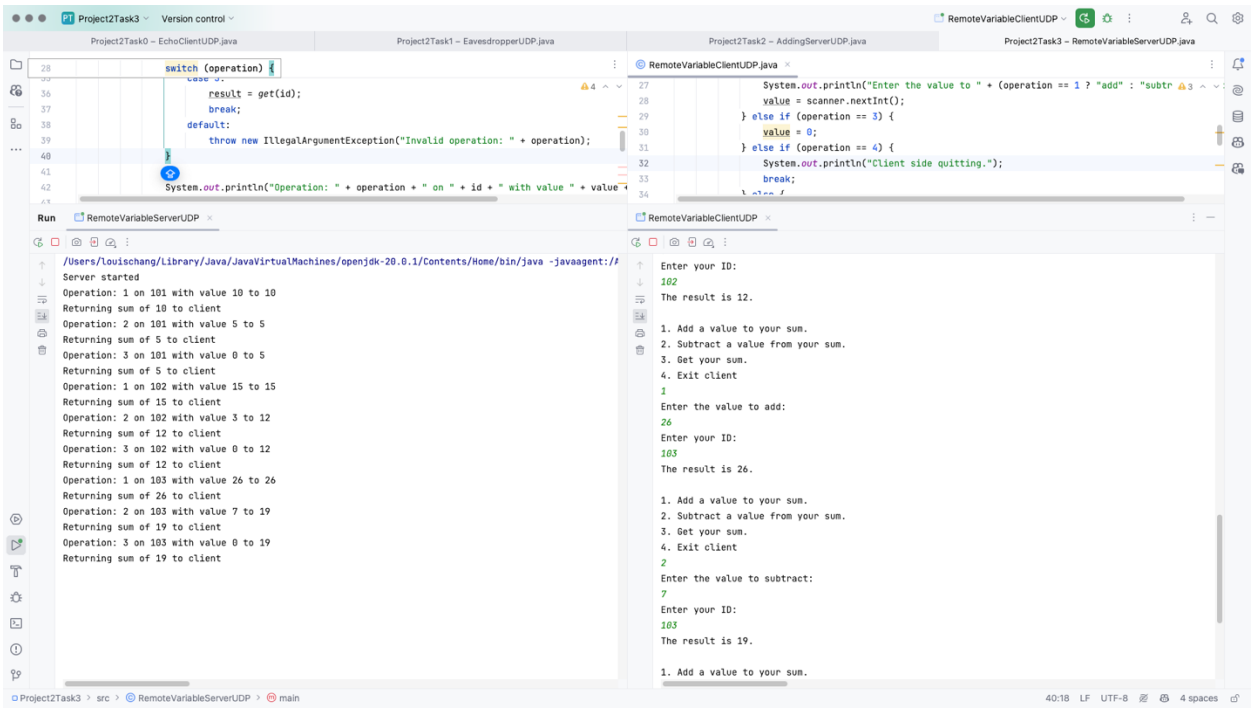
```
    private static int get(int id) {
        int sum = map.getOrDefault(id, 0);
        return sum;
    }
}
```

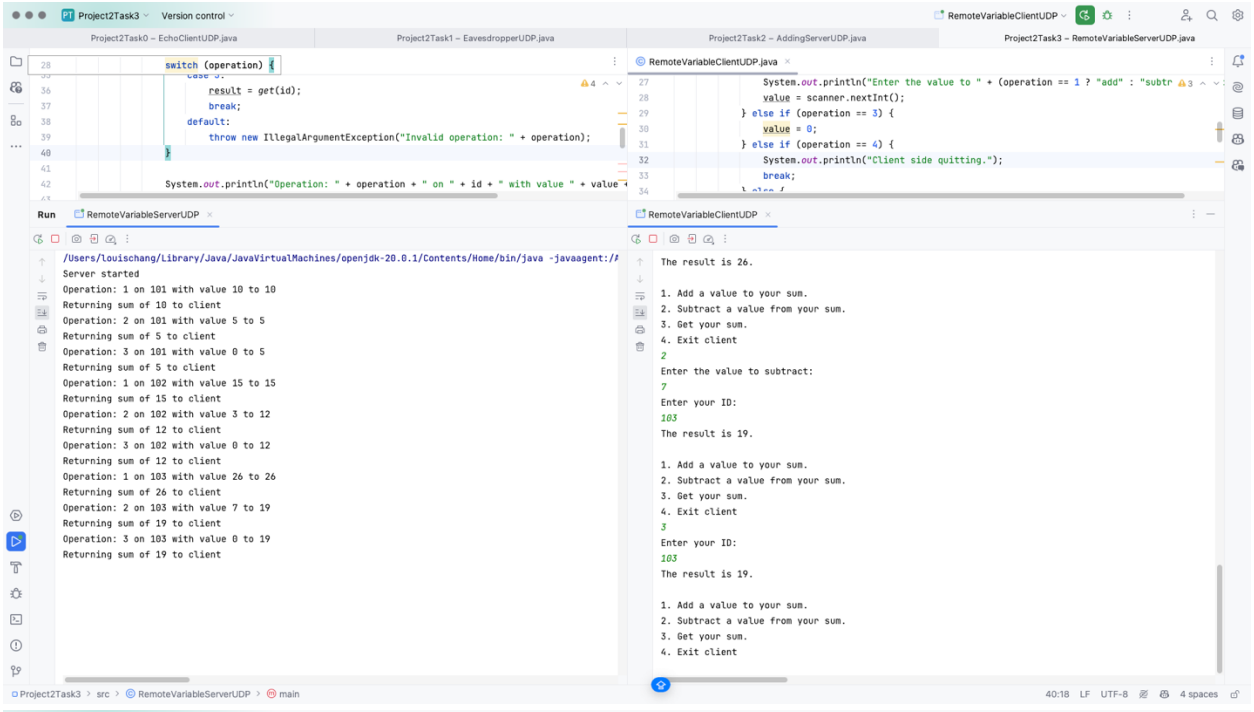
Project2Task3ServerConsole

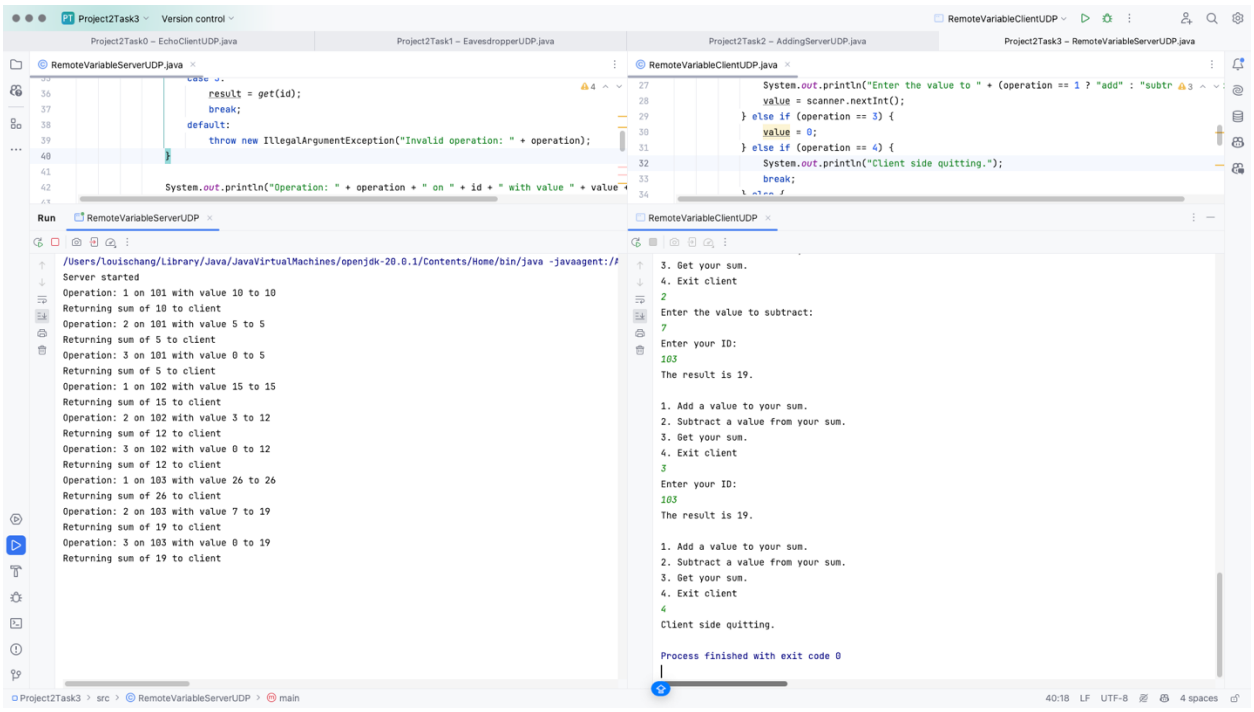


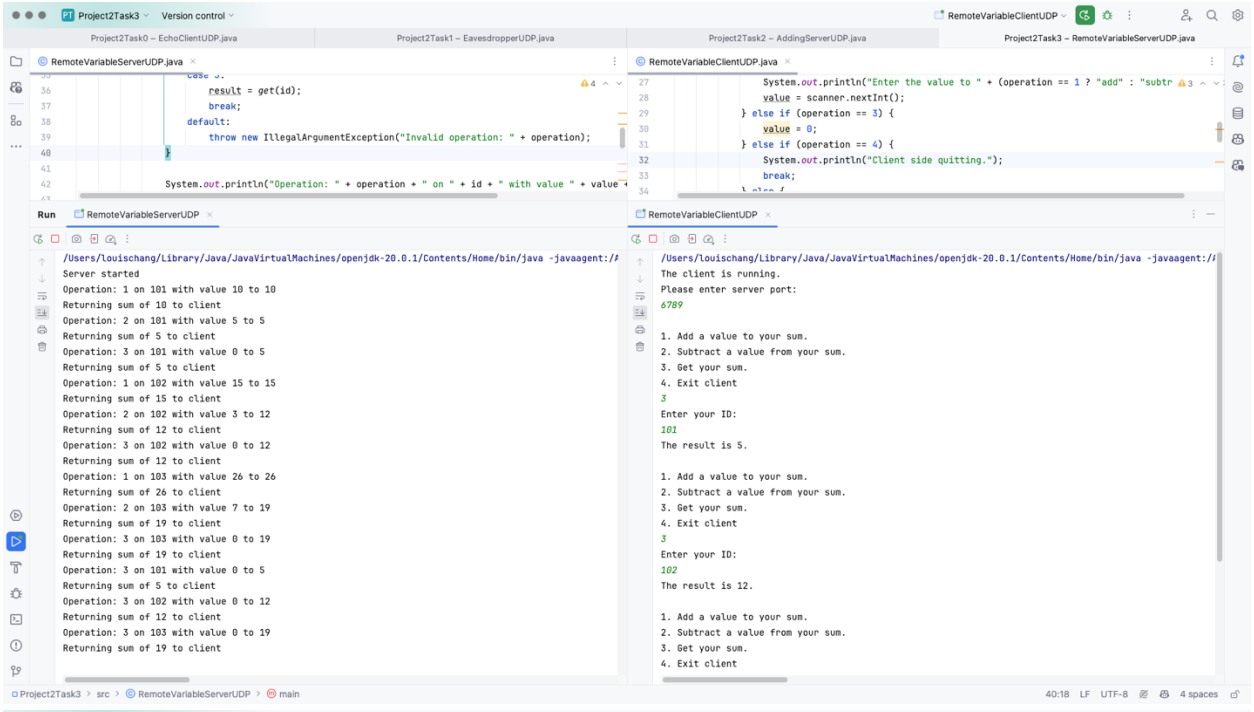


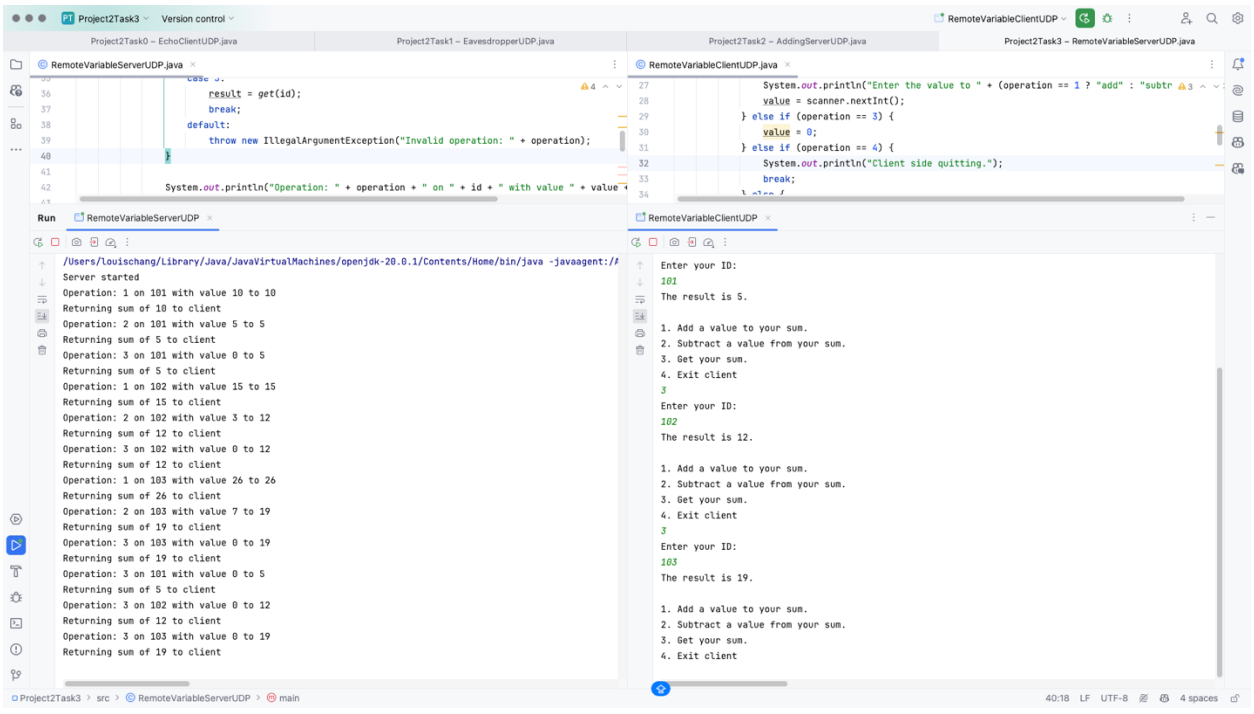

Project2Task3
Version control
RemoteVariableClientUDP
Project2Task0 – EchoClientUDP.java
Project2Task1 – EavesdropperUDP.java
Project2Task2 – AddingServerUDP.java
Project2Task3 – RemoteVariableServerUDP.java
switch (operation) {
result = get(id);
break;
default:
throw new IllegalArgumentException("Invalid operation: " + operation);
}
System.out.println("Operation: " + operation + " on " + id + " with value " + value
RemoteVariableClientUDP.java
System.out.println("Enter the value to " + (operation == 1 ? "add" : "subtr
value = scanner.nextInt();
} else if (operation == 3) {
value = 0;
} else if (operation == 4) {
System.out.println("Client side quitting.");
break;
Run
RemoteVariableServerUDP
RemoteVariableClientUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
Server started
Operation: 1 on 101 with value 10 to 10
Returning sum of 10 to client
Operation: 2 on 101 with value 5 to 5
Returning sum of 5 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 1 on 102 with value 15 to 15
Returning sum of 15 to client
Operation: 2 on 102 with value 3 to 12
Returning sum of 12 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 1 on 103 with value 26 to 26
Returning sum of 26 to client
Operation: 2 on 103 with value 7 to 19
Returning sum of 19 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
The result is 15.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
2
Enter the value to subtract:
3
Enter your ID:
102
The result is 12.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
102
The result is 12.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
1
Project2Task3 > src > RemoteVariableServerUDP > main
40:18 LF UTF-8 4 spaces

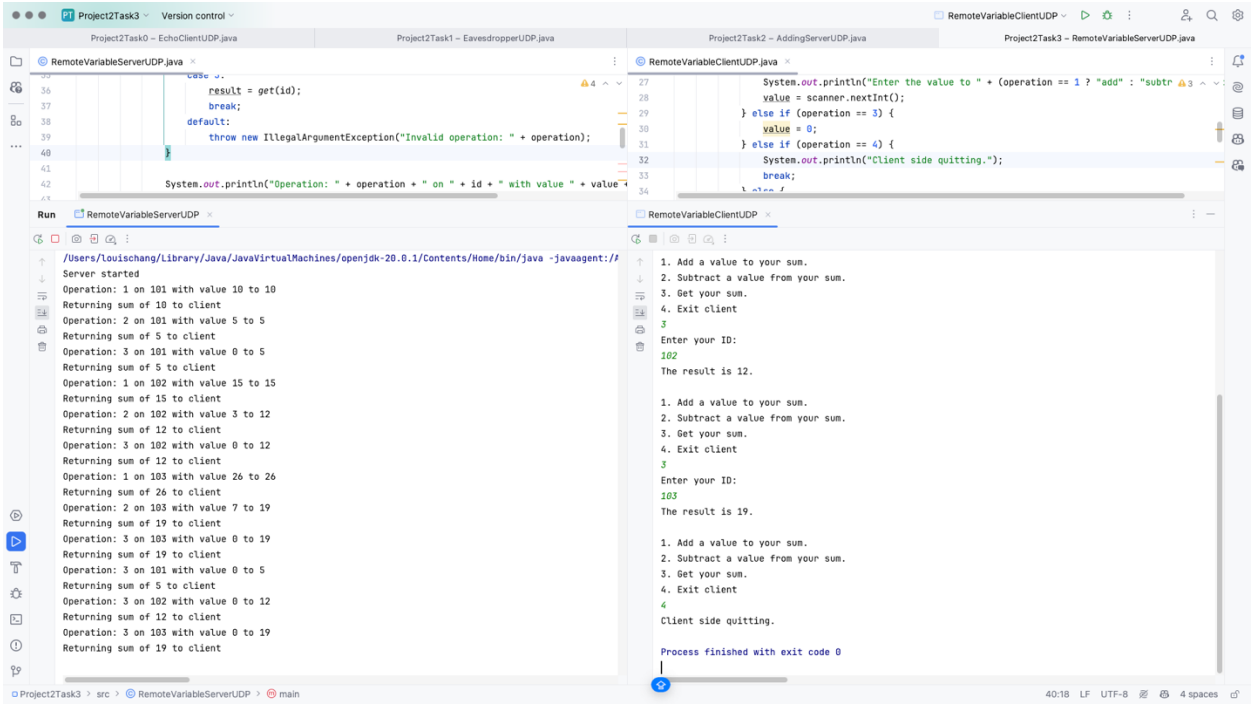

Project2Task3
Version control
RemoteVariableClientUDP
Project2Task0 – EchoClientUDP.java
Project2Task1 – EavesdropperUDP.java
Project2Task2 – AddingServerUDP.java
Project2Task3 – RemoteVariableServerUDP.java
switch (operation) {
result = get(id);
break;
default:
throw new IllegalArgumentException("Invalid operation: " + operation);
}
System.out.println("Operation: " + operation + " on " + id + " with value " + value
RemoteVariableClientUDP.java
System.out.println("Enter the value to " + (operation == 1 ? "add" : "subtr
value = scanner.nextInt();
} else if (operation == 3) {
value = 0;
} else if (operation == 4) {
System.out.println("Client side quitting.");
break;
Run
RemoteVariableServerUDP
RemoteVariableClientUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
Server started
Operation: 1 on 101 with value 10 to 10
Returning sum of 10 to client
Operation: 2 on 101 with value 5 to 5
Returning sum of 5 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 1 on 102 with value 15 to 15
Returning sum of 15 to client
Operation: 2 on 102 with value 3 to 12
Returning sum of 12 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 1 on 103 with value 26 to 26
Returning sum of 26 to client
Operation: 2 on 103 with value 7 to 19
Returning sum of 19 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
Enter your ID:
102
The result is 12.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
1
Enter the value to add:
26
Enter your ID:
103
The result is 26.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
2
Enter the value to subtract:
7
Enter your ID:
103
The result is 19.
1. Add a value to your sum.
Project2Task3 > src > RemoteVariableServerUDP > main
40:18 LF UTF-8 4 spaces


Project2Task3
Version control
RemoteVariableClientUDP
Project2Task0 – EchoClientUDP.java
Project2Task1 – EavesdropperUDP.java
Project2Task2 – AddingServerUDP.java
Project2Task3 – RemoteVariableServerUDP.java
switch (operation) {
result = get(id);
break;
default:
throw new IllegalArgumentException("Invalid operation: " + operation);
}
System.out.println("Operation: " + operation + " on " + id + " with value " + value
RemoteVariableClientUDP.java
System.out.println("Enter the value to " + (operation == 1 ? "add" : "subtr
value = scanner.nextInt();
} else if (operation == 3) {
value = 0;
} else if (operation == 4) {
System.out.println("Client side quitting.");
break;
Run
RemoteVariableServerUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
Server started
Operation: 1 on 101 with value 10 to 10
Returning sum of 10 to client
Operation: 2 on 101 with value 5 to 5
Returning sum of 5 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 1 on 102 with value 15 to 15
Returning sum of 15 to client
Operation: 2 on 102 with value 3 to 12
Returning sum of 12 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 1 on 103 with value 26 to 26
Returning sum of 26 to client
Operation: 2 on 103 with value 7 to 19
Returning sum of 19 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
RemoteVariableClientUDP
The result is 26.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
2
Enter the value to subtract:
7
Enter your ID:
103
The result is 19.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
103
The result is 19.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
Project2Task3 > src > RemoteVariableServerUDP > main
40:18 LF UTF-8 4 spaces


Project2Task3
Version control
RemoteVariableClientUDP
Project2Task0 – EchoClientUDP.java
Project2Task1 – EavesdropperUDP.java
Project2Task2 – AddingServerUDP.java
Project2Task3 – RemoteVariableServerUDP.java
RemoteVariableServerUDP.java
result = get(id);
break;
default:
throw new IllegalArgumentException("Invalid operation: " + operation);
}
System.out.println("Operation: " + operation + " on " + id + " with value " + value
RemoteVariableClientUDP.java
System.out.println("Enter the value to " + (operation == 1 ? "add" : "subtr
value = scanner.nextInt();
} else if (operation == 3) {
value = 0;
} else if (operation == 4) {
System.out.println("Client side quitting.");
break;
Run
RemoteVariableServerUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
Server started
Operation: 1 on 101 with value 10 to 10
Returning sum of 10 to client
Operation: 2 on 101 with value 5 to 5
Returning sum of 5 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 1 on 102 with value 15 to 15
Returning sum of 15 to client
Operation: 2 on 102 with value 3 to 12
Returning sum of 12 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 1 on 103 with value 26 to 26
Returning sum of 26 to client
Operation: 2 on 103 with value 7 to 19
Returning sum of 19 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
RemoteVariableClientUDP
3. Get your sum.
4. Exit client
2
Enter the value to subtract:
7
Enter your ID:
103
The result is 19.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
103
The result is 19.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
4
Client side quitting.
Process finished with exit code 0
Project2Task3 > src > RemoteVariableServerUDP > main
40:18 LF UTF-8 4 spaces


Project2Task3
Version control
RemoteVariableClientUDP
Project2Task0 – EchoClientUDP.java
Project2Task1 – EavesdropperUDP.java
Project2Task2 – AddingServerUDP.java
Project2Task3 – RemoteVariableServerUDP.java
RemoteVariableServerUDP.java
RemoteVariableClientUDP.java
result = get(id);
break;
default:
throw new IllegalArgumentException("Invalid operation: " + operation);
System.out.println("Operation: " + operation + " on " + id + " with value " + value
System.out.println("Enter the value to " + (operation == 1 ? "add" : "subtr
value = scanner.nextInt();
} else if (operation == 3) {
value = 0;
} else if (operation == 4) {
System.out.println("Client side quitting.");
break;
Run
RemoteVariableServerUDP
RemoteVariableClientUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
Server started
Operation: 1 on 101 with value 10 to 10
Returning sum of 10 to client
Operation: 2 on 101 with value 5 to 5
Returning sum of 5 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 1 on 102 with value 15 to 15
Returning sum of 15 to client
Operation: 2 on 102 with value 3 to 12
Returning sum of 12 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 1 on 103 with value 26 to 26
Returning sum of 26 to client
Operation: 2 on 103 with value 7 to 19
Returning sum of 19 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
The client is running.
Please enter server port:
6789
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
101
The result is 5.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
102
The result is 12.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
Project2Task3 > src > RemoteVariableServerUDP > main
40:18 LF UTF-8 4 spaces


Project2Task3
Version control
RemoteVariableClientUDP
Project2Task0 – EchoClientUDP.java
Project2Task1 – EavesdropperUDP.java
Project2Task2 – AddingServerUDP.java
Project2Task3 – RemoteVariableServerUDP.java
RemoteVariableServerUDP.java
RemoteVariableClientUDP.java
result = get(id);
break;
default:
throw new IllegalArgumentException("Invalid operation: " + operation);
System.out.println("Operation: " + operation + " on " + id + " with value " + value
System.out.println("Enter the value to " + (operation == 1 ? "add" : "subtr
value = scanner.nextInt();
} else if (operation == 3) {
value = 0;
} else if (operation == 4) {
System.out.println("Client side quitting.");
break;
Run
RemoteVariableServerUDP
RemoteVariableClientUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
Server started
Operation: 1 on 101 with value 10 to 10
Returning sum of 10 to client
Operation: 2 on 101 with value 5 to 5
Returning sum of 5 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 1 on 102 with value 15 to 15
Returning sum of 15 to client
Operation: 2 on 102 with value 3 to 12
Returning sum of 12 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 1 on 103 with value 26 to 26
Returning sum of 26 to client
Operation: 2 on 103 with value 7 to 19
Returning sum of 19 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
Enter your ID:
101
The result is 5.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
102
The result is 12.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
103
The result is 19.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
Project2Task3 > src > RemoteVariableServerUDP > main
40:18 LF UTF-8 4 spaces


Project2Task3
Version control
RemoteVariableClientUDP
Project2Task0 – EchoClientUDP.java
Project2Task1 – EavesdropperUDP.java
Project2Task2 – AddingServerUDP.java
Project2Task3 – RemoteVariableServerUDP.java
RemoteVariableServerUDP.java
result = get(id);
break;
default:
throw new IllegalArgumentException("Invalid operation: " + operation);
}
System.out.println("Operation: " + operation + " on " + id + " with value " + value
RemoteVariableClientUDP.java
System.out.println("Enter the value to " + (operation == 1 ? "add" : "subtr
value = scanner.nextInt();
} else if (operation == 3) {
value = 0;
} else if (operation == 4) {
System.out.println("Client side quitting.");
break;
Run
RemoteVariableServerUDP
/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/
Server started
Operation: 1 on 101 with value 10 to 10
Returning sum of 10 to client
Operation: 2 on 101 with value 5 to 5
Returning sum of 5 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 1 on 102 with value 15 to 15
Returning sum of 15 to client
Operation: 2 on 102 with value 3 to 12
Returning sum of 12 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 1 on 103 with value 26 to 26
Returning sum of 26 to client
Operation: 2 on 103 with value 7 to 19
Returning sum of 19 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
Operation: 3 on 101 with value 0 to 5
Returning sum of 5 to client
Operation: 3 on 102 with value 0 to 12
Returning sum of 12 to client
Operation: 3 on 103 with value 0 to 19
Returning sum of 19 to client
RemoteVariableClientUDP
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
102
The result is 12.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
3
Enter your ID:
103
The result is 19.
1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.
4. Exit client
4
Client side quitting.
Process finished with exit code 0
Project2Task3 > src > RemoteVariableServerUDP > main
40:18 LF UTF-8 4 spaces

Task 4

Project2Task4Client

```
import com.google.gson.Gson;
import jsonObj.req.*;
import jsonObj.res.*;

import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
import java.util.Arrays;
import java.util.Scanner;

/**
 * Author: Louis Chang (hungyic)
 * Last Modified: 02/22/2024
 */
public class NeuralNetworkClient {
    final String host = "localhost";
    static int serverPort = 6789;
    static Scanner scanner = new Scanner(System.in);
    static Gson gson = new Gson();

    public static void main(String[] args) {
        NeuralNetworkClient client = new NeuralNetworkClient();
        Scanner scanner = new Scanner(System.in);
        System.out.println("The client is running.");
        System.out.println();

        while (true) {
            NeuralNetworkRequest req;

            // Display the menu and get the user's selection.
            int userSelection = client.menu();
            if (userSelection == 0) {   // Display the current truth
table.
                req = new GetCurrentRangeRequest();
            } else if (userSelection == 1) {    // Provide four inputs for
the range of the two input truth table and build a new neural network.
                System.out.println("Enter the four results of a 4 by 2
truth table. Each value should be 0 or 1.");
                double range1 = scanner.nextDouble();
                double range2 = scanner.nextDouble();
                double range3 = scanner.nextDouble();
                double range4 = scanner.nextDouble();
                req = new SetCurrentRangeRequest(range1, range2, range3,
range4);
            } else if (userSelection == 2) {    // Perform a single
training step.
                req = new TrainRequest(1);
            } else if (userSelection == 3) {    // Perform n training
```

```
steps.
                System.out.println("Enter the number of training sets.");
                int steps = scanner.nextInt();
                req = new TrainRequest(steps);
            } else if (userSelection == 4) {    // Test with a pair of
inputs.
                System.out.println("Enter a pair of doubles from a row of
the truth table. These are domain values.");
                double input1 = scanner.nextDouble();
                double input2 = scanner.nextDouble();
                req = new TestRequest(input1, input2);
            } else if (userSelection == 5) {    // Exit program.
                System.out.println("Client side quitting.");
                break;
            } else {    // Invalid option.
                System.out.println("Invalid option.");
                continue;
            }

            // Send the request and process the response.
            String resStr = client.sendRequest(req);
            if (resStr.contains(NeuralNetworkResponse.STATUS_OK)) { //
Determine the type of response and process it.
                if
(resStr.contains(NeuralNetworkResponse.GET_CURRENT_RANGE)) { // Get the
current range
                    GetCurrentRangeResponse rangeRes =
gson.fromJson(resStr, GetCurrentRangeResponse.class);
                    double[][] userTrainingSets =
rangeRes.getUserTrainingSets();
                    System.out.println("Working with the following truth
table");
                    for (double[] userTrainingSet : userTrainingSets) {

System.out.println(Arrays.toString(userTrainingSet));
                    }
                } else if
(resStr.contains(NeuralNetworkResponse.SET_CURRENT_RANGE)) {  // Set the
current range
                    SetCurrentRangeResponse setRes = gson.fromJson(resStr,
SetCurrentRangeResponse.class);
                    System.out.println("The range has been set.");
                } else if (resStr.contains(NeuralNetworkResponse.TRAIN))
{  // Train the neural network
                    TrainResponse trainRes = gson.fromJson(resStr,
TrainResponse.class);
                    System.out.println("Training complete. Error: " +
trainRes.getError());
                    System.out.println("After " + trainRes.getSteps() + "
training steps, our error " + trainRes.getError());
                } else if (resStr.contains(NeuralNetworkResponse.TEST))
{   // Test the neural network
                    TestResponse testRes = gson.fromJson(resStr,
TestResponse.class);
```

```
                    System.out.println("The range value is approximately "
+ testRes.getRange());
                }
            } else {
                System.out.println("Error!");
            }
        }
        scanner.close();
    }
    private int menu() {
        System.out.println("Using a neural network to learn a truth
table.\nMain Menu");
        System.out.println("0. Display the current truth table.");
        System.out.println("1. Provide four inputs for the range of the
two input truth table and build a new neural network. To test XOR, enter 0
1  1  0.");
        System.out.println("2. Perform a single training step.");
        System.out.println("3. Perform n training steps. 10000 is a
typical value for n.");
        System.out.println("4. Test with a pair of inputs.");
        System.out.println("5. Exit program.");
        return scanner.nextInt();
    }
    private String sendRequest(NeuralNetworkRequest req) {
        DatagramSocket aSocket = null;
        try {
            aSocket = new DatagramSocket();
            InetAddress aHost = InetAddress.getByName(host);
            String jsonStr = gson.toJson(req);
            byte[] sendData =
jsonStr.getBytes(java.nio.charset.StandardCharsets.UTF_8);
            DatagramPacket request = new DatagramPacket(sendData,
sendData.length, aHost, serverPort);
            aSocket.send(request);

            byte[] buffer = new byte[1000];
            DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
            aSocket.receive(reply);
            return new String(reply.getData(), 0, reply.getLength());
        } catch (IOException e) {
            throw new RuntimeException(e);
        } finally {
            if (aSocket != null) aSocket.close();
        }
    }

}
```

Project2Task4Server

```
import com.google.gson.Gson;
import jsonObj.req.NeuralNetworkRequest;
```

```java
import jsonObj.req.SetCurrentRangeRequest;
import jsonObj.req.TestRequest;
import jsonObj.req.TrainRequest;
import jsonObj.res.*;
import neuronObj.NeuronLayer;

import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;
import java.util.*;

/**
 * Author: Louis Chang (hungyic)
 * Last Modified: 02/22/2024
 */
public class NeuralNetworkServer {

    public static void main(String args[]) {

        // Create an initial truth table with all 0's in the range.
        List<Double[][]> userTrainingSets = new ArrayList<>(Arrays.asList(
                new Double[][]{{0.0, 0.0}, {0.0}},
                new Double[][]{{0.0, 1.0}, {0.0}},
                new Double[][]{{1.0, 0.0}, {0.0}},
                new Double[][]{{1.0, 1.0}, {0.0}}
        ));
        //     Create a neural network suitable for working with truth
tables.
        //     There will be two inputs, 5 hidden neurons, and 1 output.
All weights and biases will be random.
        //     This is the initial neural network on start up.
        NeuralNetwork neuralNetwork = new NeuralNetwork(2, 5, 1, null,
null, null, null);

        Random rand = new Random();
        int random_choice;

        // Hold a list of doubles for input for the neural network to
train on.
        // In this example, if we want to train the neural network to
learn the XOR,
        // the list would have two doubles, say 0 1 or 1 0 or 1 1.
        List<Double> userTrainingInputs;

        // Hold a list of double for the output of training. For example,
XOR would produce 1 double as output.
        List<Double> userTrainingOutputs;

        Gson gson = new Gson();
        DatagramSocket aSocket = null;
        byte[] buffer = new byte[1000];
        try {
            System.out.println("The Neural Network server is running.");
```

```
            aSocket = new DatagramSocket(6789);
            DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
            while(true) {
                aSocket.receive(request);
                // copy the request data into a new byte array with
correct length
                byte[] requestData = new byte[request.getLength()];
                System.arraycopy(request.getData(), 0, requestData, 0,
request.getLength());
                String requestString = new String(request.getData(), 0,
request.getLength());
                System.out.println(requestString);

                // Process the request
                NeuralNetworkResponse response;
                if
(requestString.contains(NeuralNetworkRequest.GET_CURRENT_RANGE)) {   //
Get the current range
                    double[][] responseData = new double[4][3];
                    for (int r = 0; r < 4; r++) {
                        responseData[r][0] =
userTrainingSets.get(r)[0][0];
                        responseData[r][1] =
userTrainingSets.get(r)[0][1];
                        responseData[r][2] =
userTrainingSets.get(r)[1][0];
                    }
                    response = new GetCurrentRangeResponse(responseData);
                } else if
(requestString.contains(NeuralNetworkRequest.SET_CURRENT_RANGE)) {    //
Set the current range
                    SetCurrentRangeRequest req =
gson.fromJson(requestString, SetCurrentRangeRequest.class);
                    userTrainingSets = new ArrayList<>(Arrays.asList(
                            new Double[][]{{0.0, 0.0}, {req.getVal1()}},
                            new Double[][]{{0.0, 1.0}, {req.getVal2()}},
                            new Double[][]{{1.0, 0.0}, {req.getVal3()}},
                            new Double[][]{{1.0, 1.0}, {req.getVal4()}}
                    ));
                    // Build a new neural network with new random weights.
                    neuralNetwork = new NeuralNetwork(2, 5, 1, null, null,
null, null);
                    response = new SetCurrentRangeResponse();
                } else if
(requestString.contains(NeuralNetworkRequest.TRAIN)) {    // Train the
neural network
                    TrainRequest req = gson.fromJson(requestString,
TrainRequest.class);
                    int n = req.getIterations();
                    for (int i = 0; i < n; i++) {
                        random_choice = rand.nextInt(4);
                        // Get the two inputs
                        userTrainingInputs =
```

```
Arrays.asList(userTrainingSets.get(random_choice)[0]);
                        // Get the one output
                        userTrainingOutputs =
Arrays.asList(userTrainingSets.get(random_choice)[1]);
                        // Show that row to the neural network
                        neuralNetwork.train(userTrainingInputs,
userTrainingOutputs);
                    }
                    // Show error as we train
                    double error =
neuralNetwork.calculateTotalError(userTrainingSets);
                    response = new TrainResponse(n, error);
                } else if
(requestString.contains(NeuralNetworkRequest.TEST)) { // Test the neural
network
                    TestRequest req = gson.fromJson(requestString,
TestRequest.class);
                    double input0 = req.getVal1();
                    double input1 = req.getVal2();
                    List<Double> testUserInputs = new
ArrayList<>(Arrays.asList(input0, input1));
                    List<Double> userOutput =
neuralNetwork.feedForward(testUserInputs);
                    response = new TestResponse(userOutput.get(0));
                } else {    // Error
                    response = new ErrorResponse();
                }

                // reply to client
                String resJsonStr = gson.toJson(response);
                byte[] replyData =
resJsonStr.getBytes(java.nio.charset.StandardCharsets.UTF_8);
                DatagramPacket reply = new DatagramPacket(replyData,
replyData.length, request.getAddress(), request.getPort());
                aSocket.send(reply);
            }
        } catch (SocketException e) {
            System.out.println("Socket: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("IO: " + e.getMessage());
        } finally {
            if(aSocket != null) {
                System.out.println("UDP Server side quitting");
                aSocket.close();
            }
        }
    }


}
```

Project2Task4ClientConsole

AND

Server

/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54744:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath /Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/out/production/Project2Task4:/Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/libs/gson-2.10.1.jar NeuralNetworkServer

The Neural Network server is running.

Request: {"request":"getCurrentRange"}

Response: {"userTrainingSets":[[0.0,0.0,0.0],[0.0,1.0,0.0],[1.0,0.0,0.0],[1.0,1.0,0.0]],"response":"getCurrentRange","status":"OK"}

Request: {"val1":0.0,"val2":0.0,"val3":0.0,"val4":1.0,"request":"setCurrentRange"}

Response: {"response":"setCurrentRange","status":"OK"}

Request: {"iterations":1,"request":"train"}

Response: {"steps":1,"error":1.1084549190399549,"response":"train","status":"OK"}

Request: {"iterations":10000,"request":"train"}

Response: {"steps":10000,"error":0.0017290011127243906,"response":"train","status":"OK"}

Request: {"val1":0.0,"val2":1.0,"request":"test"}

Response: {"range":0.028391682068385715,"response":"test","status":"OK"}

Request: {"val1":1.0,"val2":1.0,"request":"test"}

Response: {"range":0.9563219554569747,"response":"test","status":"OK"}

Client

/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54748:/Applications/IntelliJ IDEA.app/Contents/bin -

Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath /Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/out/production/Project2Task4:/Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/libs/gson-2.10.1.jar NeuralNetworkClient

The client is running.

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

0

Working with the following truth table

[0.0, 0.0, 0.0]

[0.0, 1.0, 0.0]

[1.0, 0.0, 0.0]

[1.0, 1.0, 0.0]

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

1

Enter the four results of a 4 by 2 truth table. Each value should be 0 or 1.

0 0 0 1

The range has been set.

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

2

Training complete. Error: 1.1084549190399549

After 1 training steps, our error 1.1084549190399549

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

3

Enter the number of training sets.

10000

Training complete. Error: 0.0017290011127243906

After 10000 training steps, our error 0.0017290011127243906

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

0 1

The range value is approximately 0.028391682068385715

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

1

1

The range value is approximately 0.9563219554569747

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

5

Client side quitting.

Process finished with exit code 0

OR

Server

/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54820:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath /Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/out/production/Project2Task4:/Users/louisch

ang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/libs/gson-2.10.1.jar NeuralNetworkServer

The Neural Network server is running.

Request: {"request":"getCurrentRange"}

Response: {"userTrainingSets":[[0.0,0.0,0.0],[0.0,1.0,0.0],[1.0,0.0,0.0],[1.0,1.0,0.0]],"response":"getCurrentRange","status":"OK"}

Request: {"val1":0.0,"val2":1.0,"val3":1.0,"val4":1.0,"request":"setCurrentRange"}

Response: {"response":"setCurrentRange","status":"OK"}

Request: {"iterations":1,"request":"train"}

Response: {"steps":1,"error":0.3972011875805595,"response":"train","status":"OK"}

Request: {"iterations":10000,"request":"train"}

Response: {"steps":10000,"error":0.0011188945284393045,"response":"train","status":"OK"}

Request: {"val1":0.0,"val2":0.0,"request":"test"}

Response: {"range":0.03627045832220423,"response":"test","status":"OK"}

Request: {"val1":1.0,"val2":0.0,"request":"test"}

Response: {"range":0.9787818305142123,"response":"test","status":"OK"}

Request: {"val1":0.0,"val2":1.0,"request":"test"}

Response: {"range":0.9786271552897065,"response":"test","status":"OK"}

Request: {"val1":1.0,"val2":1.0,"request":"test"}

Response: {"range":0.9960969623832302,"response":"test","status":"OK"}

Client

/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54824:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -

classpath /Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/out/production/Project2Task4:/Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/libs/gson-2.10.1.jar NeuralNetworkClient

The client is running.

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

0

Working with the following truth table

[0.0, 0.0, 0.0]

[0.0, 1.0, 0.0]

[1.0, 0.0, 0.0]

[1.0, 1.0, 0.0]

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

1

Enter the four results of a 4 by 2 truth table. Each value should be 0 or 1.

0 1 1 1

The range has been set.

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

2

Training complete. Error: 0.3972011875805595

After 1 training steps, our error 0.3972011875805595

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

3

Enter the number of training sets.

10000

Training complete. Error: 0.0011188945284393045

After 10000 training steps, our error 0.0011188945284393045

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

0 0

The range value is approximately 0.03627045832220423

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

1 0

The range value is approximately 0.9787818305142123

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

0 1

The range value is approximately 0.9786271552897065

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

1 1

The range value is approximately 0.9960969623832302

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

5

Client side quitting.

Process finished with exit code 0

XOR

Server

/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54878:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath /Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/out/production/Project2Task4:/Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/libs/gson-2.10.1.jar NeuralNetworkServer

The Neural Network server is running.

Request: {"request":"getCurrentRange"}

Response:
{"userTrainingSets":[[0.0,0.0,0.0],[0.0,1.0,0.0],[1.0,0.0,0.0],[1.0,1.0,0.0]],"response":"getCurrentRange","status":"OK"}

Request: {"val1":0.0,"val2":1.0,"val3":1.0,"val4":0.0,"request":"setCurrentRange"}

Response: {"response":"setCurrentRange","status":"OK"}

Request: {"iterations":1,"request":"train"}

Response: {"steps":1,"error":0.8333599643312776,"response":"train","status":"OK"}

Request: {"iterations":10000,"request":"train"}

Response: {"steps":10000,"error":0.008496602731396057,"response":"train","status":"OK"}

Request: {"val1":0.0,"val2":0.0,"request":"test"}

Response: {"range":0.07884362892839757,"response":"test","status":"OK"}

Request: {"val1":1.0,"val2":0.0,"request":"test"}

Response: {"range":0.937885306189667,"response":"test","status":"OK"}

Request: {"val1":1.0,"val2":1.0,"request":"test"}

Response: {"range":0.05837590189499034,"response":"test","status":"OK"}

Client

/Users/louischang/Library/Java/JavaVirtualMachines/openjdk-20.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54885:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath /Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/out/production/Project2Task4:/Users/louischang/Library/CloudStorage/OneDrive-andrew.cmu.edu/DS/Project2/Project2Task4/libs/gson-2.10.1.jar NeuralNetworkClient

The client is running.

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

0

Working with the following truth table

[0.0, 0.0, 0.0]

[0.0, 1.0, 0.0]

[1.0, 0.0, 0.0]

[1.0, 1.0, 0.0]

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

1

Enter the four results of a 4 by 2 truth table. Each value should be 0 or 1.

0 1 1 0

The range has been set.

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

2

Training complete. Error: 0.8333599643312776

After 1 training steps, our error 0.8333599643312776

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

3

Enter the number of training sets.

10000

Training complete. Error: 0.008496602731396057

After 10000 training steps, our error 0.008496602731396057

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

0 0

The range value is approximately 0.07884362892839757

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

1 0

The range value is approximately 0.937885306189667

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

4

Enter a pair of doubles from a row of the truth table. These are domain values.

1 1

The range value is approximately 0.05837590189499034

Using a neural network to learn a truth table.

Main Menu

0. Display the current truth table.

1. Provide four inputs for the range of the two input truth table and build a new neural network. To test XOR, enter 0 1 1 0.

2. Perform a single training step.

3. Perform n training steps. 10000 is a typical value for n.

4. Test with a pair of inputs.

5. Exit program.

5

Client side quitting.

Process finished with exit code 0