Distributed Systems – Project 2

Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu

Project2Task1

Project2Task1Client:

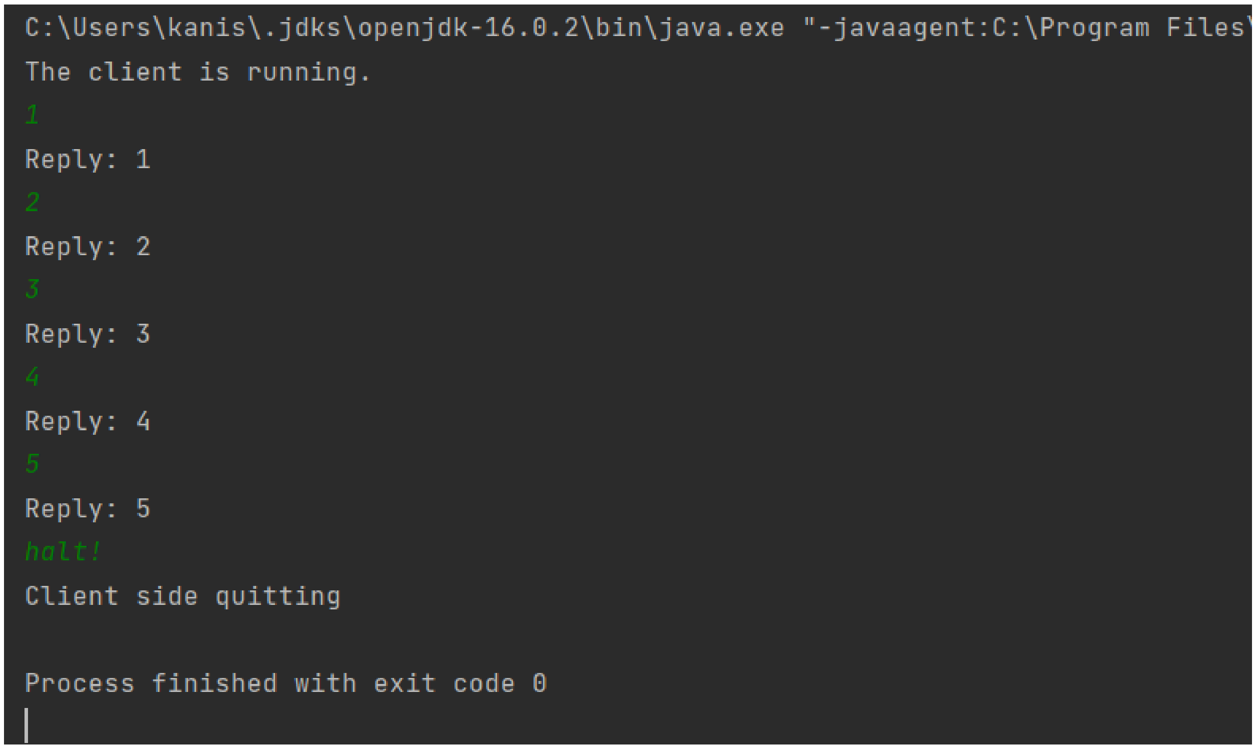
```
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */

import java.net.*;
import java.io.*;
import java.util.Arrays;

/**
 * class EchoClientUDP
 * UDP Client for sending requests to the server
 */
public class EchoClientUDP{
    /**
     * Connecting and parsing requests to the server
     * @param args
     */
    public static void main(String args[]){
        // Initializing DatagramSocket
        DatagramSocket aSocket = null;
        try {
            //Getting the address of the server
            InetAddress aHost = InetAddress.getByName("localhost");
            //Server port
            int serverPort = 6789;
            //Socket initialization
            aSocket = new DatagramSocket();
            String nextLine;
            //BufferedReader to get the input from the user
            BufferedReader typed = new BufferedReader(new
InputStreamReader(System.in));
            System.out.println("The client is running.");
            //While loop till the user keeps sending data
            while ((nextLine = typed.readLine()) != null) {
                //converting the data in byte array
                byte [] m = nextLine.getBytes();
                //DatagramPacket to the server
                DatagramPacket request = new DatagramPacket(m,  m.length,
aHost, serverPort);
                //Sending the datagram packet to the server
                aSocket.send(request);
                //Size of the byte array
                byte[] buffer = new byte[1000];
                //Replying back to the server
                DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
                //Getting the data from the server
                String requestString = new String(request.getData());
                //If the user did not send halt
                if(!(requestString.equalsIgnoreCase("halt!")))
                {
                    //Receive reply from the server
                    aSocket.receive(reply);
                    //Byte array in the length og the data that was recieved
- truncating the extra length
                    byte[] reply_bytes = Arrays.copyOf(reply.getData(),
reply.getLength());
                    //Converting byte array to string
                    String replyString = new String(reply_bytes);
                    //Printing the reply for the user
                    System.out.println("Reply: " + replyString);
                }
                else {
                    //Client quits if user sends halt
                    System.out.println("Client side quitting");
                    break; //breaking out of the code
                }
            }

        }catch (SocketException e) //Socket Exception if connection issues
        {System.out.println("Socket: " + e.getMessage());
        }catch (IOException e) //input output exception for the user
        {System.out.println("IO: " + e.getMessage());
        }finally  //after the request is done, if socket if not null then
close it
        {if(aSocket != null) aSocket.close();}
    }
}
```

Project2Task1Server:

```
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */

import java.net.*;
import java.io.*;
import java.util.Arrays;

/**
 * class EchoServerUDP
 * UDP Server for echoing and replying to the client
 */
public class EchoServerUDP{
    /**
     * Connection and parsing of messages using UDP
     * @param args no arguments
     */
    public static void main(String args[]){
        //Assigning null to UDP socket
        DatagramSocket aSocket = null;
        //Byte array with 1000 length
        byte[] buffer = new byte[1000];
        try{
            //connection to the socket with the port 6789
            aSocket = new DatagramSocket(6789);
            //Creating a datagram packet to sent the data
            DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
            System.out.println("The server is running.");
            //while loop for checking the connection and printing the
recieved data
            while(true){
                //Recieving the request from client
                aSocket.receive(request);
                //Replying back to the client
                DatagramPacket reply = new DatagramPacket(request.getData(),
                        request.getLength(), request.getAddress(),
request.getPort());
                //Converting the response to the length of the given response
instead of 1000
                byte[] response =
Arrays.copyOf(request.getData(),request.getLength());
                //Converting byte array to string
                String requestString = new String(response);
                //If the server receives halt message
                if(requestString.equalsIgnoreCase("halt!"))
                {
                    //Server quits
                    System.out.println("Server side quitting");
                    break;
                }
                else {
                    //Printing the port of the client
                    System.out.println("Client listening on: " +
request.getPort());
                    //Echoing the request from the client
                    System.out.println("Echoing: " + requestString);
                    //Sending the reply to the client
                    aSocket.send(reply);
                }
            }
        }catch (SocketException e) //catch for checking the socket connection
        {System.out.println("Socket: " + e.getMessage());
        }catch (IOException e) //Checking for input output exceptions
        {System.out.println("IO: " + e.getMessage());
        }finally //If socket is not null and request is done, close the
socket
        {if(aSocket != null) aSocket.close();}
    }
}
```

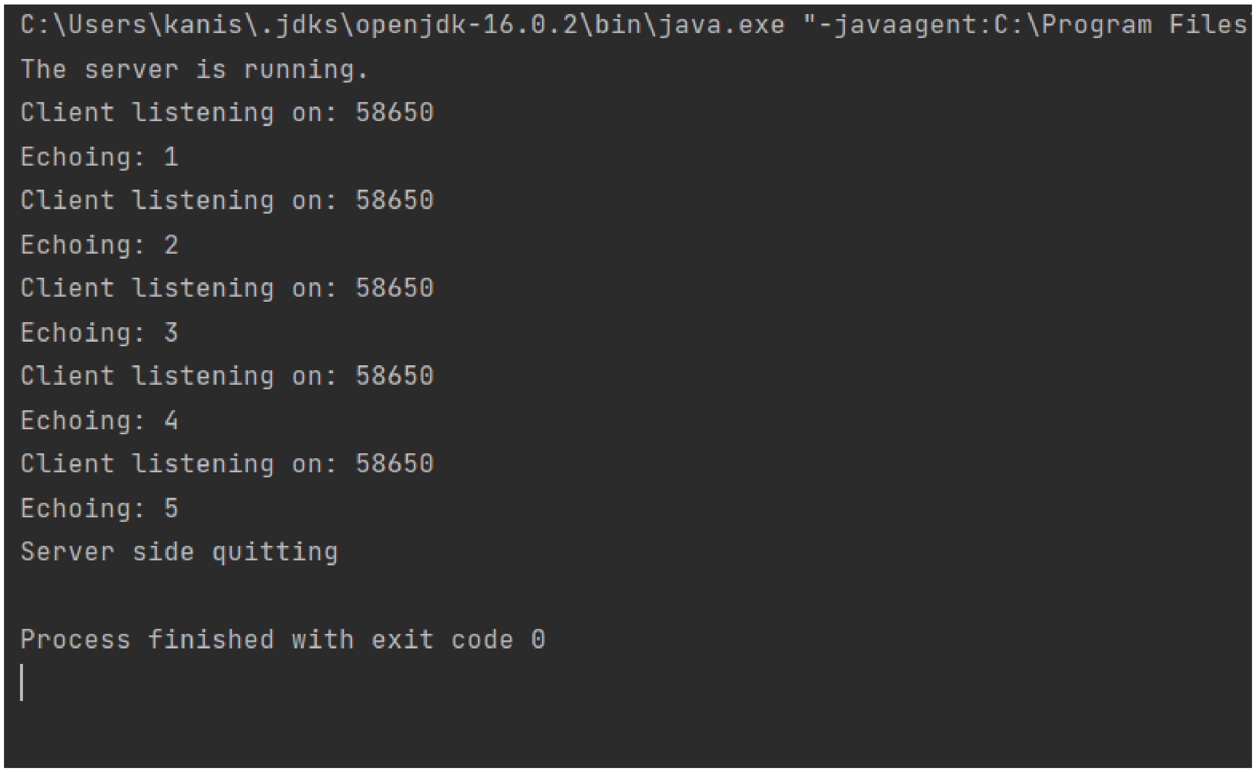
Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu

<u>Project2Task1ClientScreen:</u>



```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The client is running.
1
Reply: 1
2
Reply: 2
3
Reply: 3
4
Reply: 4
5
Reply: 5
halt!
Client side quitting

Process finished with exit code 0
```

<u>Project2Task1ServerScreen:</u>



```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The server is running.
Client listening on: 58650
Echoing: 1
Client listening on: 58650
Echoing: 2
Client listening on: 58650
Echoing: 3
Client listening on: 58650
Echoing: 4
Client listening on: 58650
Echoing: 5
Server side quitting

Process finished with exit code 0
```

Project2Task2

Project2Task2Client:

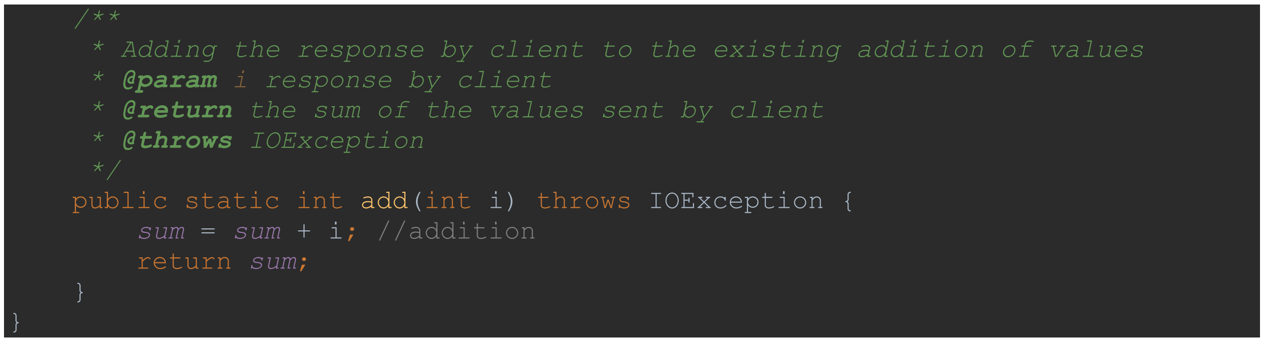
```java
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */

import java.net.*;
import java.io.*;
import java.nio.ByteBuffer;
import java.util.Arrays;

/**
 * class AddingClientUDP
 * UDP Client for sending requests to the server
 */
public class AddingClientUDP{
    static DatagramSocket aSocket = null;
    /**
     * Connecting and parsing requests to the server
     * @param args
     */
    public static void main(String args[]){
        // args give message contents and server hostname
        try {
            String nextLine;
            //BufferedReader to get the input from the user
            BufferedReader typed = new BufferedReader(new
InputStreamReader(System.in));
            System.out.println("The client is running.");
            //While loop till the user keeps sending data
            while ((nextLine = typed.readLine()) != null) {
                //If the user sends halt
                if(nextLine.equalsIgnoreCase("halt!"))
                {
                    //Client quits if user sends halt
                    System.out.println("Client side quitting");
                    break; //breaking out of the code
                }
                else {
                    //Sending the integer to the server
                    int nextInt = Integer.parseInt(nextLine);
                    System.out.println("The server returned: " +
add(nextInt));
                }
            }

        }catch (SocketException e) //Socket Exception if connection issues
        {System.out.println("Socket: " + e.getMessage());
        }catch (IOException e) //input output exception for the user
        {System.out.println("IO: " + e.getMessage());
        }finally //after the request is done, if socket if not null then
close it
        {if(aSocket != null) aSocket.close();}
    }

    /**
     * function add()
     * To send the integer values to server, sent my user to add
     * and get back the reply
     * @param i int that needs to be added
     * @return result from the server
     * @throws IOException
     */
    public static int add(int i) throws IOException {
        // Initializing DatagramSocket
        aSocket = new DatagramSocket();
        //Getting the address of the server
        InetAddress aHost = InetAddress.getByName("localhost");
        //Server port
        int serverPort = 6789;
        //Converting int to byte array
        byte[] m = ByteBuffer.allocate(4).putInt(i).array();
        //DatagramPacket to the server
        DatagramPacket request = new DatagramPacket(m, m.length, aHost,
serverPort);
        //Sending the datagram packet to the server
        aSocket.send(request);
        //Size of the byte array
        byte[] buffer = new byte[1000];
        //Replying back to the server
        DatagramPacket reply = new DatagramPacket(buffer, buffer.length);
        //Receive reply from the server
        aSocket.receive(reply);
        //Byte array in the length og the data that was recieved - truncating
the extra length
        byte[] reply_bytes = Arrays.copyOf(reply.getData(),
reply.getLength());
        //Converting byte array result to int
        int result = ByteBuffer.wrap(reply_bytes).getInt();
        //returning result
        return result;
    }
}
```

Project2Task2Server:

```
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */

import java.net.*;
import java.io.*;
import java.nio.ByteBuffer;
import java.util.Arrays;
```

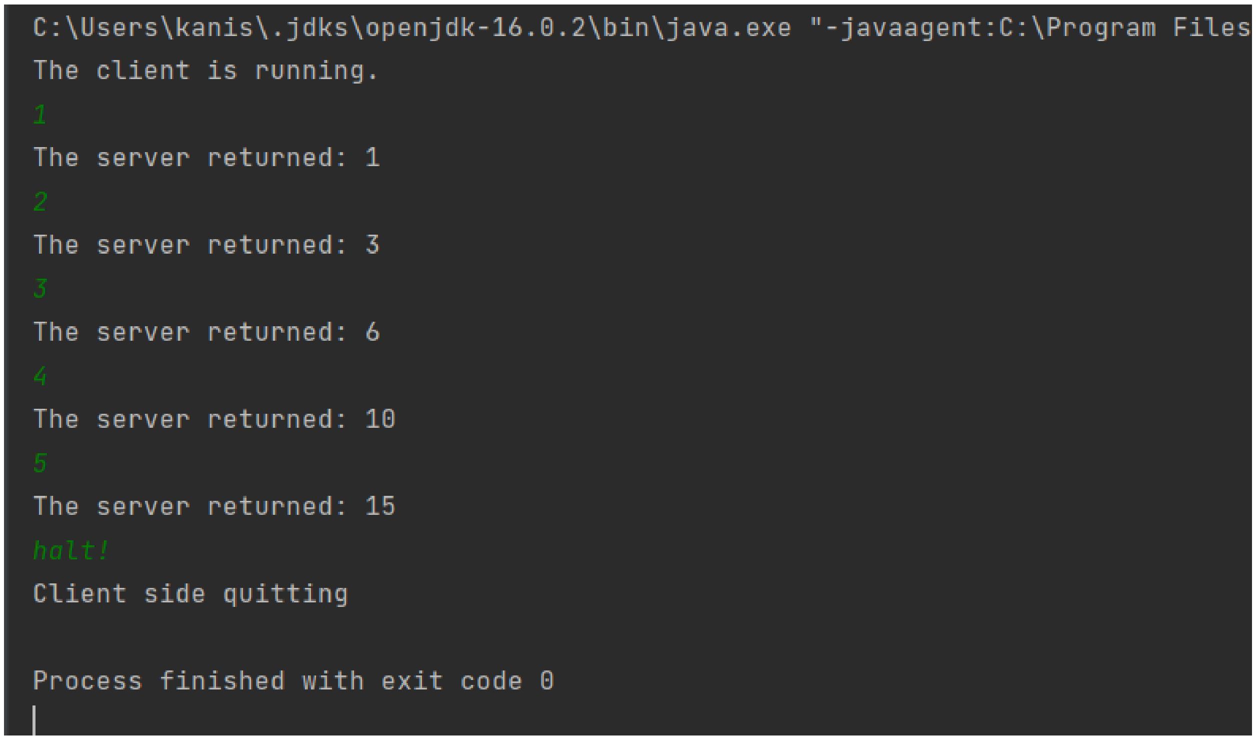
```
/**
 * class AddingServerUDP
 * UDP Server for echoing and replying to the client
 */
public class AddingServerUDP{
    static int sum = 0;
    /**
     * Connection and parsing of messages using UDP
     * @param args no arguments
     */
    public static void main(String args[]){
        //Assigning null to UDP socket
        DatagramSocket aSocket = null;
        //Byte array with 1000 length
        byte[] buffer = new byte[1000];
        try{
            //connection to the socket with the port 6789
            aSocket = new DatagramSocket(6789);
            //Creating a datagram packet to send the data
            DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
            System.out.println("The server is running.");
            //while loop for checking the connection and printing the recie
            while(true){
                //Recieving the request from client
                aSocket.receive(request);
                byte[] response =
Arrays.copyOf(request.getData(),request.getLength());
                String requestString = new String(response);
                if(!(requestString.equalsIgnoreCase("halt!")))
                {
                    //Converting byte array to int
                    int sum_value = ByteBuffer.wrap(response).getInt();
                    System.out.println( "Adding "+ sum_value + " to " + sum);
                    //add function to add the values
                    int result = add(sum_value);
                    //Converting int to byte array
                    byte[] m = ByteBuffer.allocate(4).putInt(result).array();
                    //Replying back to the client
                    DatagramPacket reply_client = new DatagramPacket(m,
m.length, request.getAddress(), request.getPort());
                    //Sending the reply to the client
                    aSocket.send(reply_client);
                    //Printing the sum
                    System.out.println("Returning sum of " + result + " to
client.");
                }
            }
        }catch (SocketException e)  //catch for checking the socket conne
        {System.out.println("Socket: " + e.getMessage());
        }catch (IOException e) //Checking for input output exceptions
        {System.out.println("IO: " + e.getMessage());
        }finally //If socket is not null and request is done, close the
socket
        {if(aSocket != null) aSocket.close();}
    }
```

Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu



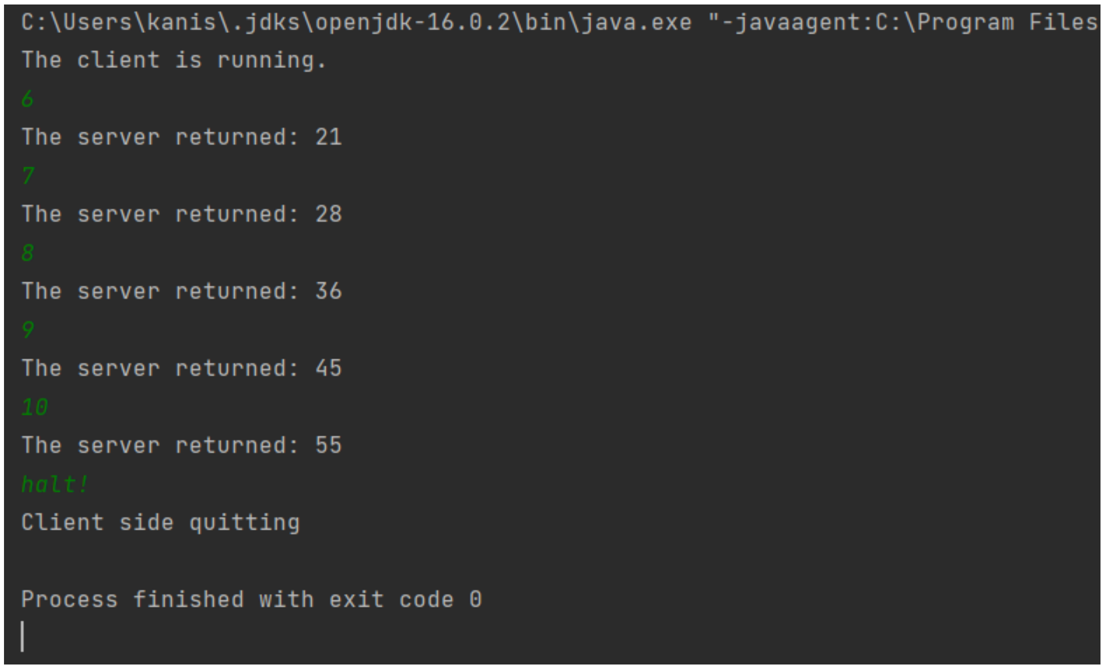
```
    /**
     * Adding the response by client to the existing addition of values
     * @param i response by client
     * @return the sum of the values sent by client
     * @throws IOException
     */
    public static int add(int i) throws IOException {
        sum = sum + i; //addition
        return sum;
    }
}
```

<u>Project2Task2ClientScreen:</u>



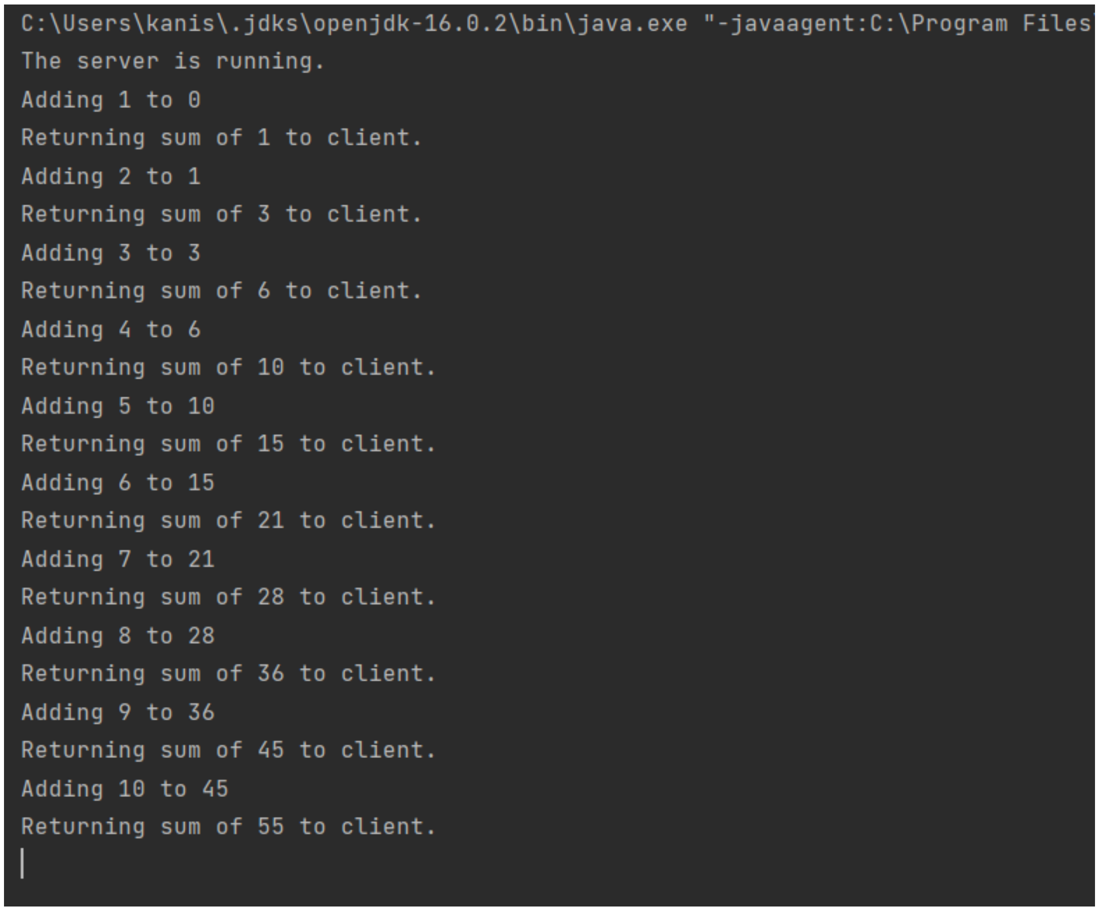
```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The client is running.
1
The server returned: 1
2
The server returned: 3
3
The server returned: 6
4
The server returned: 10
5
The server returned: 15
halt!
Client side quitting

Process finished with exit code 0
```

Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu



```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The client is running.
6
The server returned: 21
7
The server returned: 28
8
The server returned: 36
9
The server returned: 45
10
The server returned: 55
halt!
Client side quitting

Process finished with exit code 0
```

Project2Task2ServerScreen:



```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The server is running.
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 6 to client.
Adding 4 to 6
Returning sum of 10 to client.
Adding 5 to 10
Returning sum of 15 to client.
Adding 6 to 15
Returning sum of 21 to client.
Adding 7 to 21
Returning sum of 28 to client.
Adding 8 to 28
Returning sum of 36 to client.
Adding 9 to 36
Returning sum of 45 to client.
Adding 10 to 45
Returning sum of 55 to client.
```

Project2Task3

Project2Task3Client:

```
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */
import java.net.*;
import java.io.*;
import java.nio.ByteBuffer;
import java.util.Arrays;
import java.util.Scanner;

/**
 * class RemoteVariableClientUDP
 * UDP Client for sending requests to the server
 */
public class RemoteVariableClientUDP {
    //Assigning null to datagram socket
    static DatagramSocket aSocket = null;

    /**
     * Connecting and parsing requests to the server
     * @param args
     */
    public static void main(String args[]) {
        try {
            //client is running
            System.out.println("The client is running.");
            //function to address concern of separation
            menu();
        } catch (SocketException e) //Socket Exception if connection issues
        {
            System.out.println("Socket: " + e.getMessage());
        } catch (IOException e) //input output exception for the user
        {
            System.out.println("IO: " + e.getMessage());
        } finally //after the request is done, if socket if not null then
close it
        {
            if (aSocket != null) aSocket.close();
        }
    }

    /**
     * The menu function to provide the choice to the user
     * send those choices with the values to the server along with the client
id
     * @throws IOException
     */
    public static void menu() throws IOException {
        int choice, sum_value, id; //choice, the value sent by user and the
id of the user
```

```
        Scanner input = new Scanner(System.in);
        do {
            // Initializing DatagramSocket
            aSocket = new DatagramSocket();
            //Getting the address of the server
            InetAddress aHost = InetAddress.getByName("localhost");
            int serverPort = 6789;
            //Menu choices
            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");
            choice = input.nextInt();
            //An array to store choice, operand and id sent to the user
            int[] message = new int[3];
            message[0] = choice;
            switch (choice) {
                case 1: //If the user wants to add
                    System.out.println("Enter value to add:");
                    sum_value = input.nextInt();
                    message[1] = sum_value; //value of addition
                    System.out.println("Enter your ID (between 1000-1999):");
                    id = input.nextInt(); //id of the user
                    message[2] = id;
                    send_operation(id, aHost, serverPort, message);
                    break;
                case 2: //If the user wants to subtract
                    System.out.println("Enter value to subtract:");
                    sum_value = input.nextInt();
                    message[1] = sum_value;
                    System.out.println("Enter your ID:");
                    id = input.nextInt();
                    message[2] = id;
                    send_operation(id, aHost, serverPort, message);
                    break;
                case 3: //If the user wants to get the sum
                    message[1] = 0;
                    System.out.println("Enter your ID:");
                    id = input.nextInt();
                    message[2] = id;
                    send_operation(id, aHost, serverPort, message);
                    break;
                case 4: //When client quits
                    System.out.println("Client side quitting. The remote
variable server is still running.");
                    break;
                default: System.out.println("Wrong Choice.Try again");
                    break;
            }
    }while(choice < 4);

    }

    /**
     * function send_operation()
     * To send all the data required to compute to the server
     * @param id - client id
     * @param aHost - server host
     * @param serverPort - server port
     * @param message - the message array that needs to be sent
     * @throws IOException
     */
    private static void send_operation(int id, InetAddress aHost, int
serverPort, int[] message) throws IOException {
        byte[] m;
        if (id < 1000 || id > 1999) {
            System.out.println("Wrong id. Please Enter data again.");
        } else {
            //function to conver int array to byte array
            m = convertIntArrayToByteArray(message);
            //DatagramPacket to the server
            DatagramPacket request = new DatagramPacket(m, m.length, aHost,
serverPort);
            aSocket.send(request);
            //Size of the byte array
            byte[] buffer = new byte[1000];
            //Replying back to the server
            DatagramPacket reply = new DatagramPacket(buffer, buffer.length);
            //Receive reply from the server
            aSocket.receive(reply);
            byte[] reply_bytes = Arrays.copyOf(reply.getData(),
reply.getLength());
            //Converting byte array result to int
            int result = ByteBuffer.wrap(reply_bytes).getInt();
            System.out.println("The result is " + result + "\n");
        }
    }

    /**
     * function convertIntArraytoByteArray()
     * Function to convert the array of integers to array of bytes
     * @param data the int array
     * @return the byte array
     */
    private static byte[] convertIntArrayToByteArray(int[] data) {
        if (data == null) return null;
        // ----------
        byte[] bytes = new byte[data.length * 4];
        for (int i = 0; i < data.length; i++)
            System.arraycopy(ByteBuffer.allocate(4).putInt(data[i]).array(),
0, bytes, i * 4, 4);
        return bytes;
    }
}
```

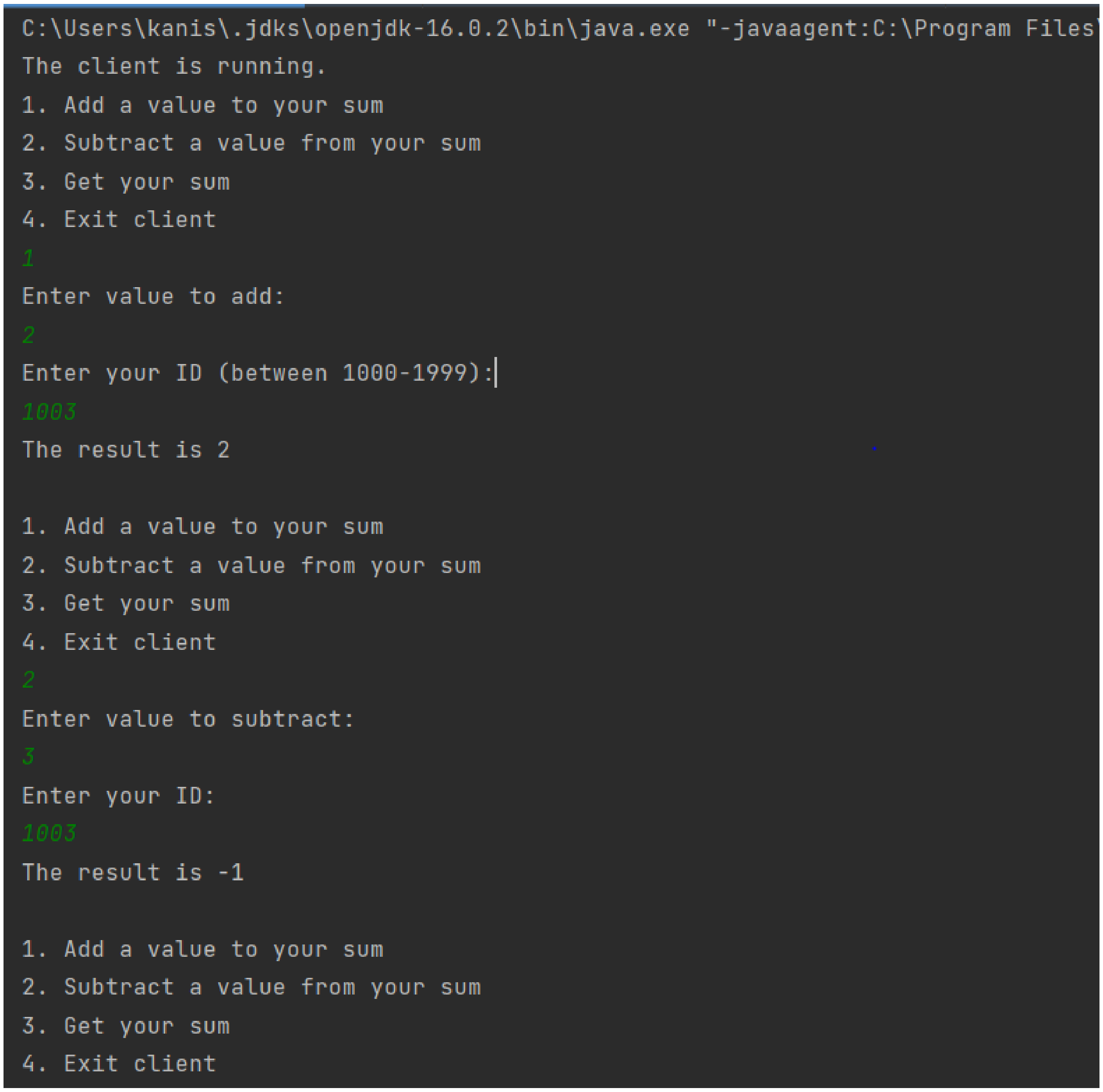
Project2Task3Server:

```java
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */

import java.net.*;
import java.io.*;
import java.nio.ByteBuffer;
import java.util.Arrays;
import java.util.HashMap;
import java.util.Map;

/**
 * class RemoteVariableServerUDP
 * UDP Server for echoing and replying to the client
 */
public class RemoteVariableServerUDP {
    //HashMap to store the client id and the value
    static Map<Integer,Integer> store_message = new HashMap<>();
    /**
     * Connection and parsing of messages using UDP
     * @param args no arguments
     */
    public static void main(String args[]){
        //Assigning null to UDP socket
        DatagramSocket aSocket = null;
        //Byte array with 1000 length
        byte[] buffer = new byte[1000];
        try{
            //connection to the socket with the port 6789
            aSocket = new DatagramSocket(6789);
            //Creating a datagram packet to send the data
            DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
            System.out.println("The server is running.");
            while(true){
                //Recieving the request from client
                aSocket.receive(request);
                byte[] response =
Arrays.copyOf(request.getData(),request.getLength());
                //converting byte array to int array
                int[] message = convertByteArrayToIntArray(response);
                //function to add, subtract or get the values
                int result = compute_values(message);
                System.out.println("The client " + message[2] + " requested "
+ operationId_to_operation(message[0]) + " and the result is: " + result);
                //converting result int to byte array
                byte[] m = ByteBuffer.allocate(4).putInt(result).array();
                //Replying back to the client with datagram packet
                DatagramPacket reply_client = new DatagramPacket(m, m.length,
request.getAddress(), request.getPort());
                aSocket.send(reply_client);

            }
        }catch (SocketException e) //catch for checking the socket connection
        {System.out.println("Socket: " + e.getMessage());
        }catch (IOException e) //Checking for input output exceptions
        {System.out.println("IO: " + e.getMessage());
        }finally //If socket is not null and request is done, close the
socket
        {if(aSocket != null) aSocket.close();}
    }

/**
 * To return the value of the operation based on the choice
 * @param i the int choice for the operation
 * @return the value of the operation
 */
    private static String operationId_to_operation(int i) {
        if(i == 1)
        {
            return "addition";
        }
        else if(i==2){
            return "subtraction";
        }
        else
        {
            return "get";
        }
    }


    /**
     * function convertByteArraytoIntArray()
     * Function to convert the array of integers to array of bytes
     * @param data the int array
     * @return the byte array
     */
    public static int[] convertByteArrayToIntArray(byte[] data) {
        if (data == null || data.length % 4 != 0) return null;
        // ----------
        int[] ints = new int[data.length / 4];
        for (int i = 0; i < ints.length; i++)
            ints[i] = ( convertByteArrayToInt(new byte[] {
                    data[(i*4)], //data and the conversion to 4 bits
                    data[(i*4)+1],
                    data[(i*4)+2],
                    data[(i*4)+3],
            } ));
        return ints;
    }

    /**
     * function convertByteArraytoInt
     * converting a sine 4 bit byte array to int
     * @param intBytes byte array
     * @return
     */
    private static int convertByteArrayToInt(byte[] intBytes){
        ByteBuffer byteBuffer = ByteBuffer.wrap(intBytes);
        return byteBuffer.getInt();
```

```java
    }

    /**
     * function compute_values
     * The function is used to check whether an id already exists or not.
     * To add one if does not exists and add the sum or subtract
     * If exists then add or subtract to the existing values
     * @param message
     * @return
     */
    public static int compute_values(int[] message) {
        //checking if Id exists
        if(store_message.get(message[2])!= null)
        {
            //if the choice is 1 then add
            if(message[0] == 1)
            {
                store_message.put(message[2],store_message.get(message[2]) +
message[1]);
            }
            //else subtract
            else if(message[0] == 2)
            {
                store_message.put(message[2], store_message.get(message[2]) -
message[1]);
            }
        }
        else //if if does not exists
        {
            //if choice is 1 add
            if(message[0] == 1)
            {
                store_message.put(message[2], message[1]);
            }
            //if choice is 2 add -ve
            else if(message[0] == 2)
            {
                store_message.put(message[2], -message[1]);
            }
            //if choice is 3 and id doesn't exist then return 0
            else
            {
                store_message.put(message[2], 0);
            }
        }
        //get the result of the id
        int result = store_message.get(message[2]);
        //return result
        return result;

    }
}
```

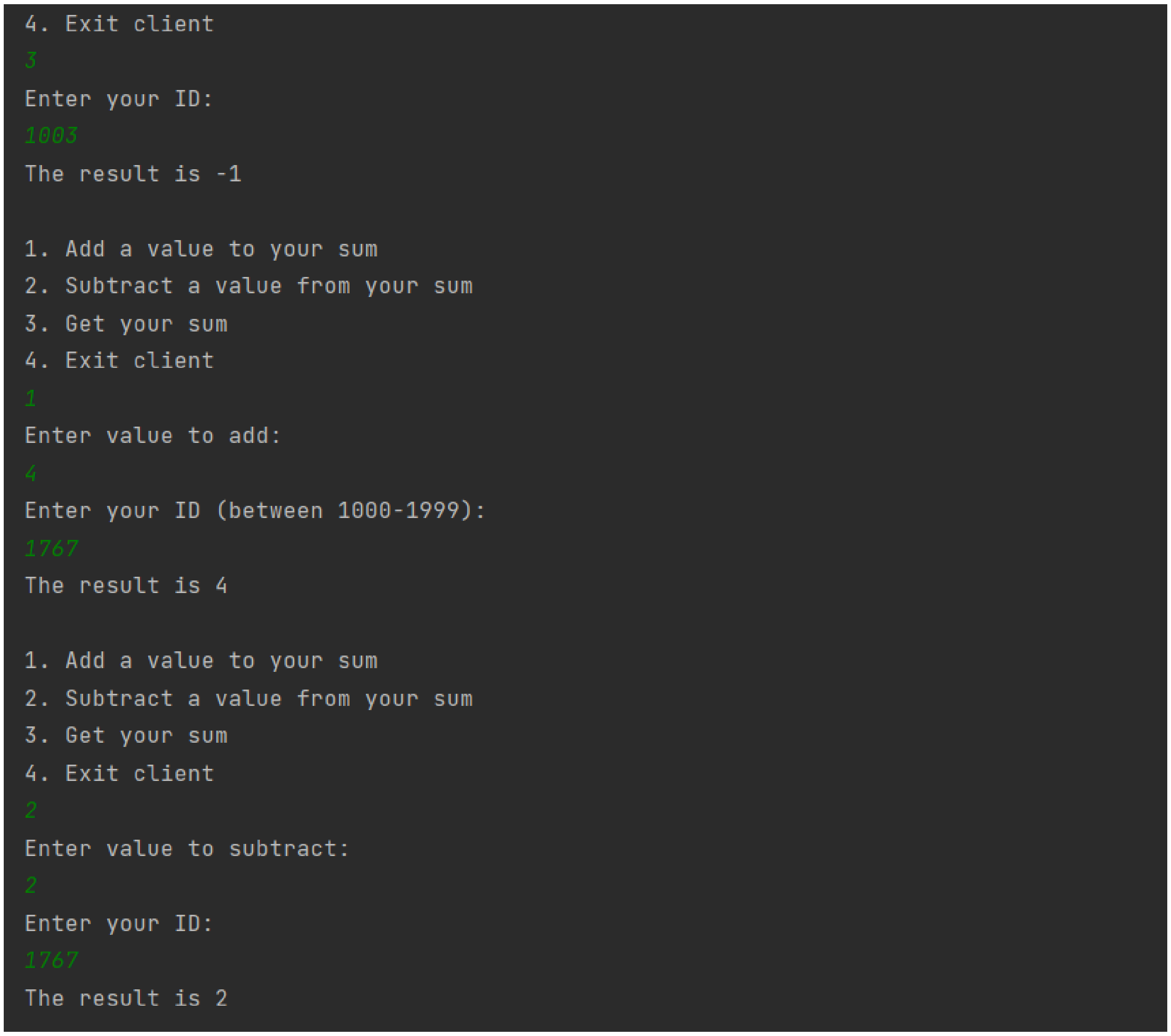
Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu

Project2Task3ClientScreen:

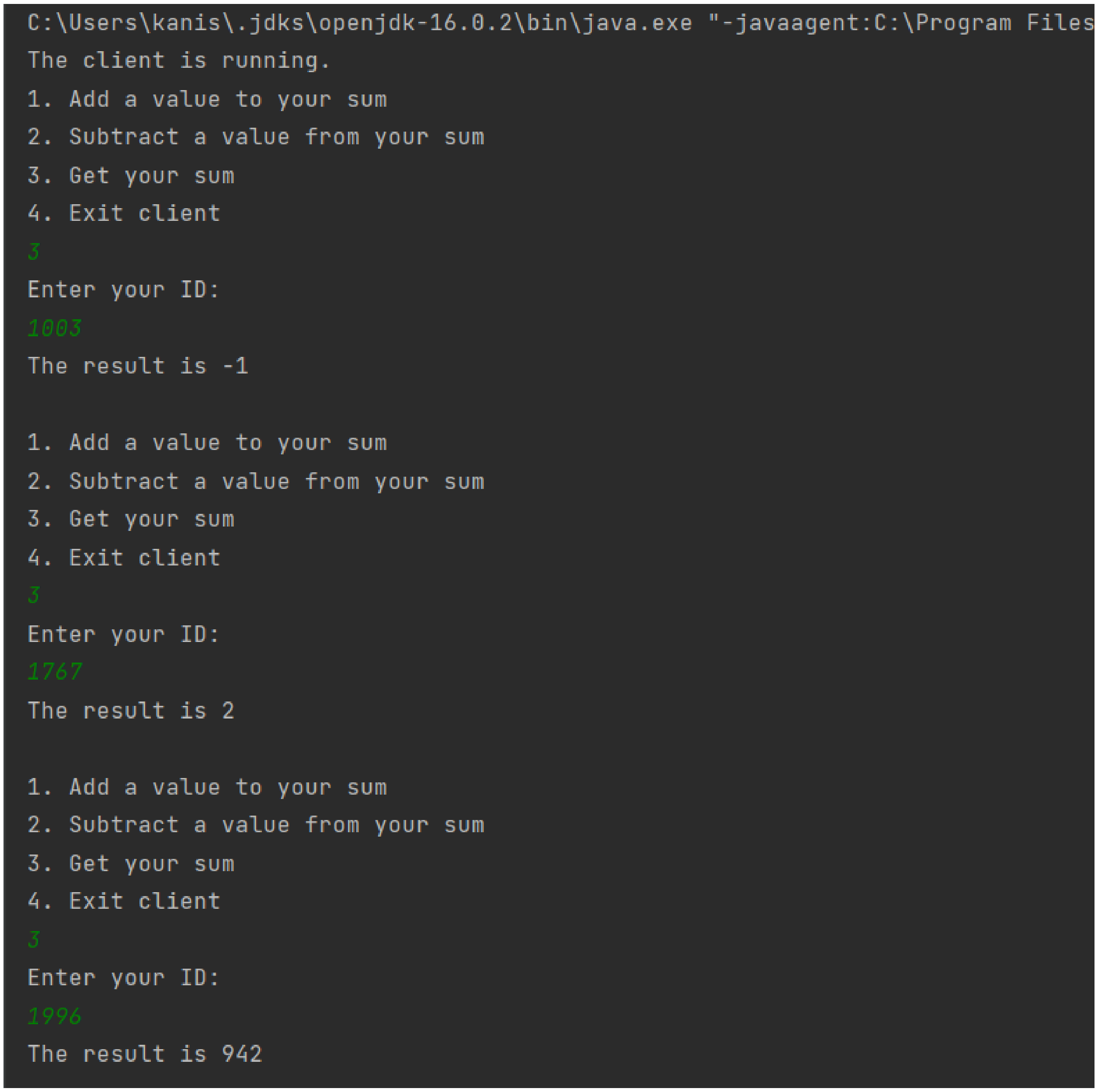


```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The client is running.
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
1
Enter value to add:
2
Enter your ID (between 1000-1999):
1003
The result is 2

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
3
Enter your ID:
1003
The result is -1

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
```

Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu

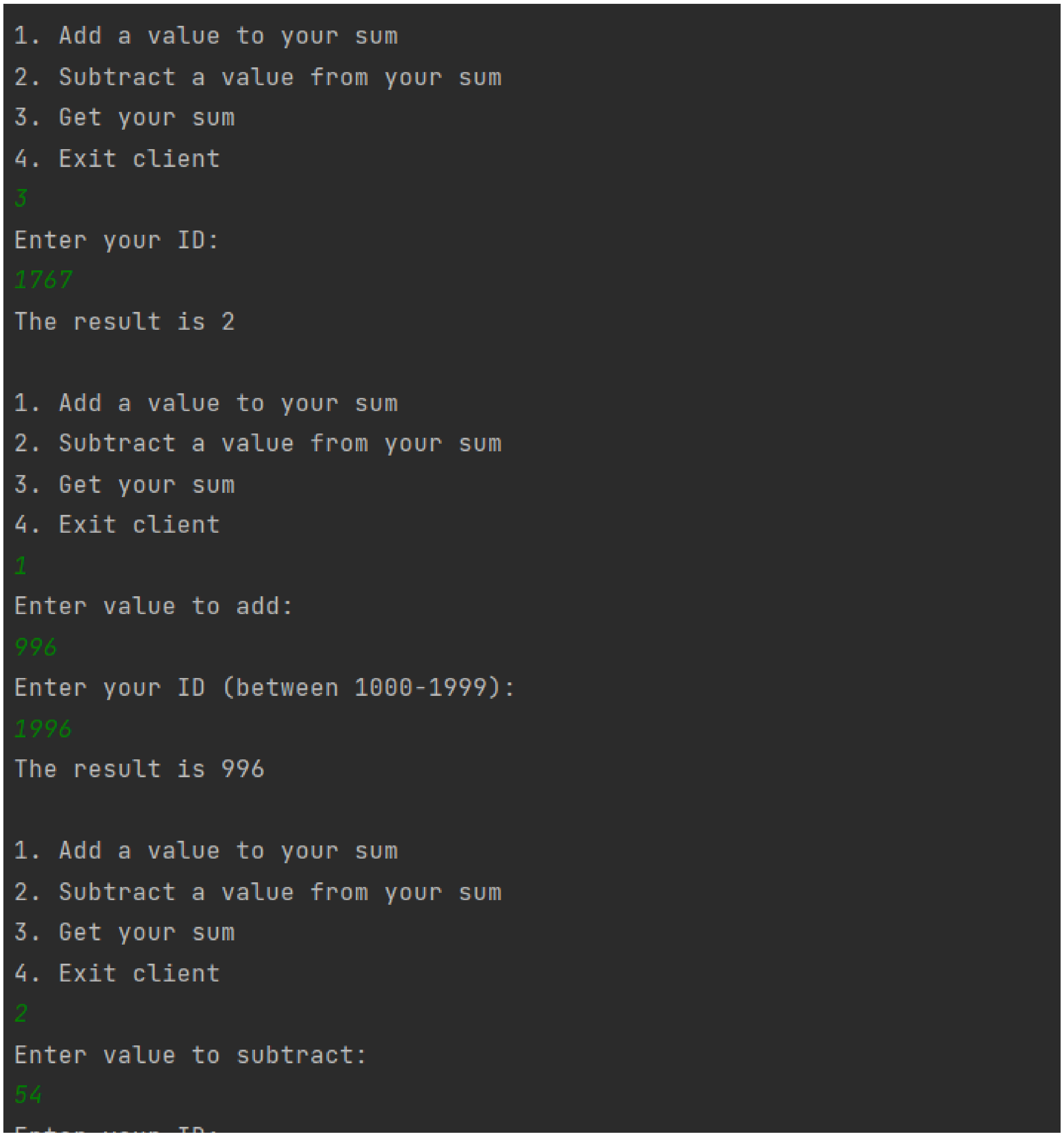


```
4. Exit client
3
Enter your ID:
1003
The result is -1

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
1
Enter value to add:
4
Enter your ID (between 1000-1999):
1767
The result is 4

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
2
Enter your ID:
1767
The result is 2
```

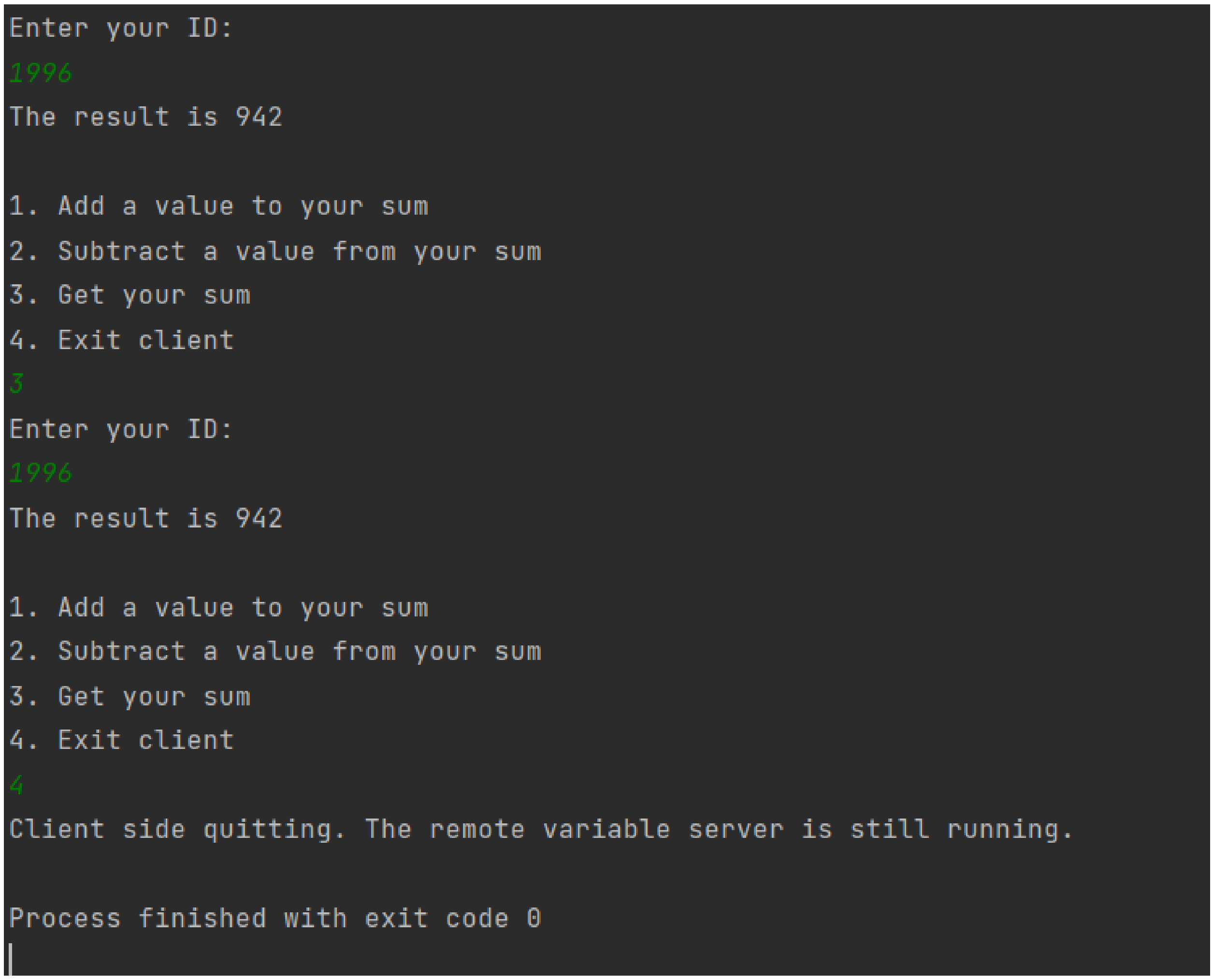


```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The client is running.
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
3
Enter your ID:
1003
The result is -1

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:
1767
The result is 2

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:
1996
The result is 942
```

Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu

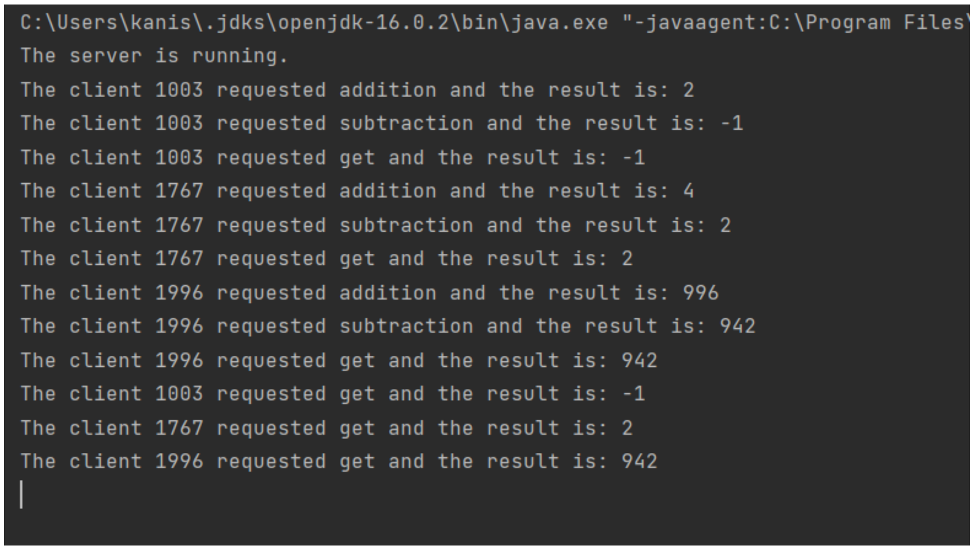


```
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
3
Enter your ID:
1767
The result is 2

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
1
Enter value to add:
996
Enter your ID (between 1000-1999):
1996
The result is 996

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
54
```



```
Enter your ID:
1996
The result is 942

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:
1996
The result is 942

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. The remote variable server is still running.

Process finished with exit code 0
```

<u>Project2Task3ServerScreen:</u>



```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The server is running.
The client 1003 requested addition and the result is: 2
The client 1003 requested subtraction and the result is: -1
The client 1003 requested get and the result is: -1
The client 1767 requested addition and the result is: 4
The client 1767 requested subtraction and the result is: 2
The client 1767 requested get and the result is: 2
The client 1996 requested addition and the result is: 996
The client 1996 requested subtraction and the result is: 942
The client 1996 requested get and the result is: 942
The client 1003 requested get and the result is: -1
The client 1767 requested get and the result is: 2
The client 1996 requested get and the result is: 942
```

Project2Task4

Project2Task4Client:

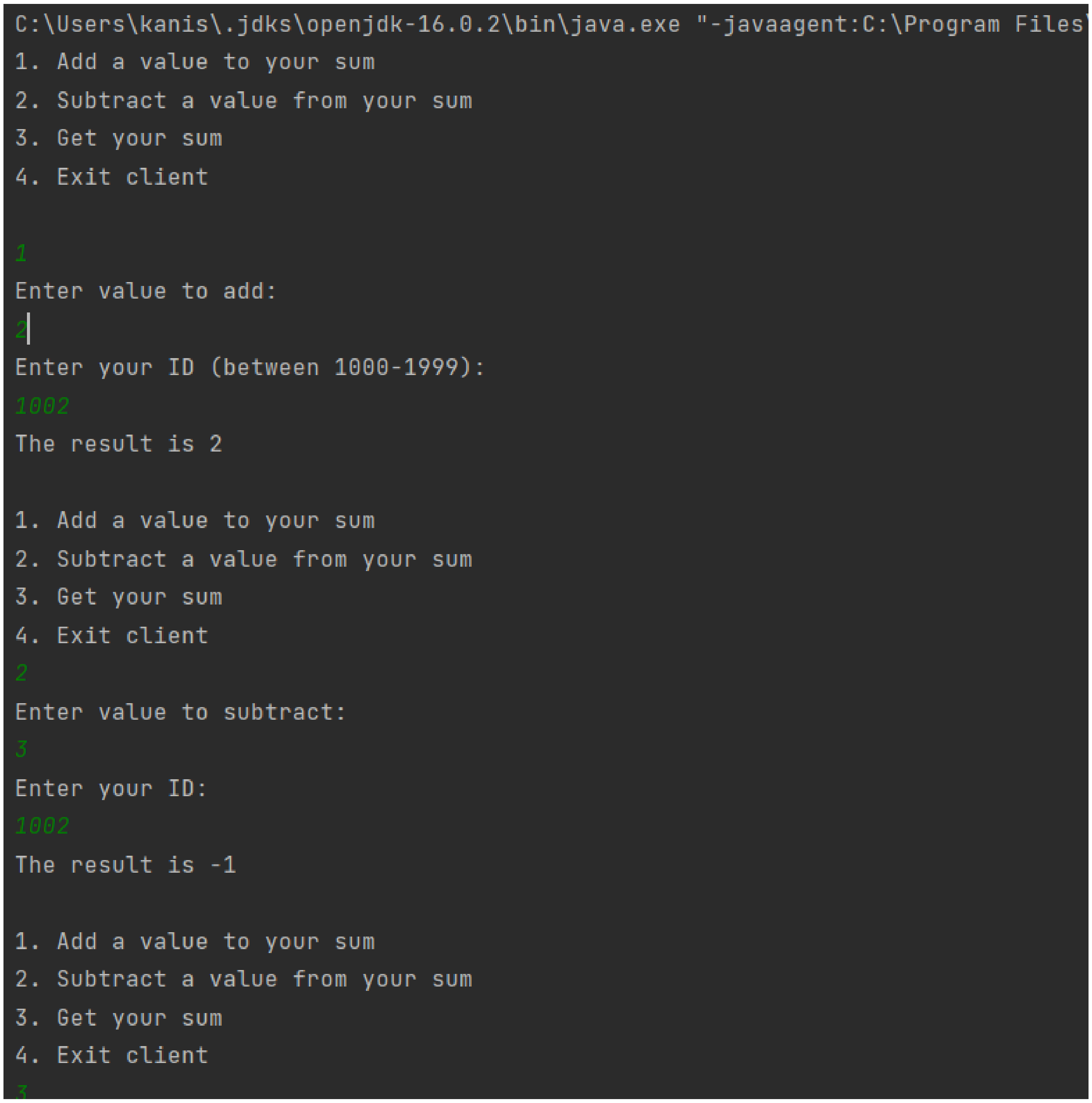
```java
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */
import java.net.*;
import java.io.*;
import java.nio.ByteBuffer;
import java.util.Arrays;
import java.util.Scanner;

/**
 * class RemoteVariableClientTCP
 * TCP Client for sending requests to the server
 */
public class RemoteVariableClientTCP {

    /**
     * Connecting and parsing requests to the server
     * @param args
     */
    public static void main(String args[]) {
        // arguments supply hostname
        Socket clientSocket = null;
        try {
            int serverPort = 7777;
            //connection with socket
            clientSocket = new Socket("localhost", serverPort);
            //Input from the server
            BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
            //output from the server
            PrintWriter out = new PrintWriter(new BufferedWriter(new
OutputStreamWriter(clientSocket.getOutputStream())));
            //function to address concern of separation
            menu(out, in);
        } catch (IOException e) { //input output exception for the user
            System.out.println("IO Exception:" + e.getMessage());
        } finally { //after the request is done, if socket if not null then
close it
            try {
                if (clientSocket != null) {
                    clientSocket.close();
                }
            } catch (IOException e) {
                // ignore exception on close
            }
        }
    }

    /**
```

```
     * The menu function to provide the choice to the user
     * send those choices with the values to the server along with the client
id
     * @throws IOException
     */
    public static void menu(PrintWriter out, BufferedReader in) throws
IOException {
        int choice, sum_value, id;
        //input scanner for user
        Scanner input = new Scanner(System.in);
        do {
            //Menu choices
            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");
            choice = input.nextInt();
            int[] message = new int[3];
            //An array to store choice, operand and id sent to the user
            message[0] = choice;
            switch (choice) {
                case 1: //If the user wants to add
                    System.out.println("Enter value to add:");
                    sum_value = input.nextInt();
                    message[1] = sum_value; //value of addition
                    System.out.println("Enter your ID (between 1000-1999):");
                    id = input.nextInt();
                    message[2] = id;  //id of the user
                    //operation to send the data to the server
                    send_operation(out, in, id, message);
                    break;
                case 2: //If the user wants to subtract
                    System.out.println("Enter value to subtract:");
                    sum_value = input.nextInt();
                    message[1] = sum_value;
                    System.out.println("Enter your ID:");
                    id = input.nextInt();
                    message[2] = id;
                    send_operation(out, in, id, message);
                    break;
                case 3: //If the user wants to get the sum
                    message[1] = 0;
                    System.out.println("Enter your ID:");
                    id = input.nextInt();
                    message[2] = id;
                    send_operation(out, in, id, message);
                    break;
                case 4: //When client quits
                    System.out.println("Client side quitting. The remote
variable server is still running.");
                    break;
                default: System.out.println("Wrong Choice.Try again");
                    break;
            }
        }while(choice < 4);

    }
    /**
     * function send_operation()
     * To send all the data required to compute to the server
     * @param out the out scanner for server
     * @param in the input scanner for server
     * @param id id of the client
     * @param message message that needs to be sent to the server
     * @throws IOException
     */
    private static void send_operation(PrintWriter out, BufferedReader in,
int id, int[] message) throws IOException {
        if (id < 1000 || id > 1999) {
            System.out.println("Wrong id. Please Enter data again.");
        } else {
            //Sending message to the server
            out.println(Arrays.toString(message));
            out.flush();
            //Receive reply from the server
            String result = in.readLine();
            System.out.println("The result is " + result + "\n");
        }
    }
}
```

Project2Task4Server:

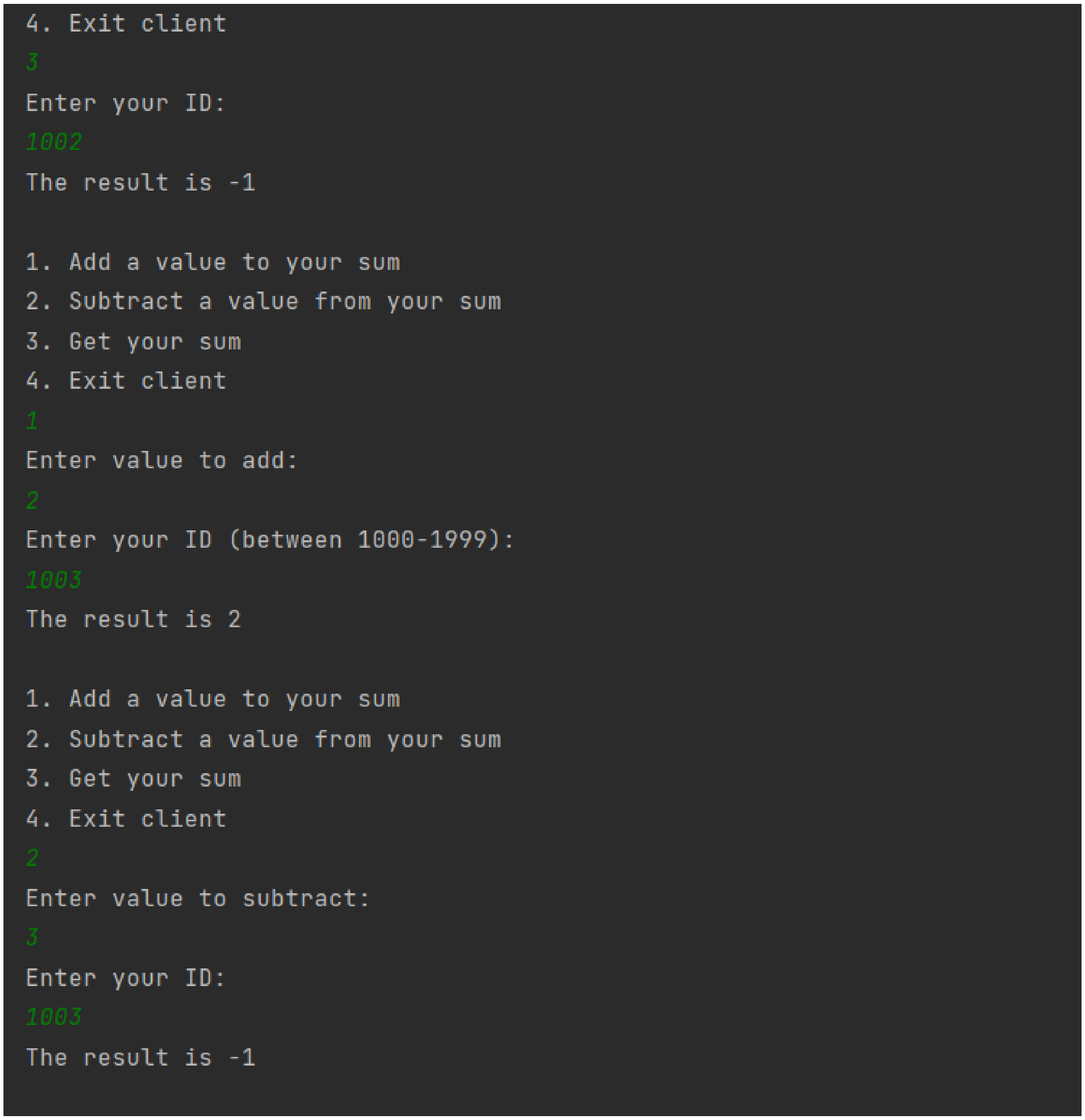
```
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */
import java.net.*;
import java.io.*;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

/**
 * class RemoteVariableServerTCP
 * TCP Server for echoing and replying to the client
 */
public class RemoteVariableServerTCP {
    //HashMap to store the client id and the value
    static Map<Integer,Integer> store_message = new HashMap<>();
    /**
     * Connection and parsing of messages using UDP
     * @param args no arguments
     */
    public static void main(String args[]) {
        Socket clientSocket = null;
            try {
                int serverPort = 7777; // the server port we are using

                // Create a new server socket
                ServerSocket listenSocket = new ServerSocket(serverPort);
                while (true) {
                    //Checking if there is a client or not, if not then
listening for a client
                    if((clientSocket == null)||
(clientSocket.getInputStream().read() == -1)) {
                        clientSocket = listenSocket.accept();
                    }

                    //"in" to read from the client socket
                    Scanner in;
                    in = new Scanner(clientSocket.getInputStream());

                    //"out" to write to the client socket
                    PrintWriter out;
                    out = new PrintWriter(new BufferedWriter(new
OutputStreamWriter(clientSocket.getOutputStream())));

                    //converting the message recieved to string array and
then to int array
                    String[] message_string = in.nextLine().replaceAll("\\[",
"")

                            .replaceAll("]", "")
                            .replaceAll(" ", "")
                            .split(",");
                    //Converting the message to int array
                    int[] message = new int[message_string.length];
                    for (int i = 0; i < message_string.length; i++) {
                        message[i] = Integer.valueOf(message_string[i]);
                    }
                    //getting the result after computing the values
                    String result = String.valueOf(compute_values(message));
                    //sending back the result
                    out.println(result);
                    //printing the result
                    System.out.println("The result of " +
operationId_to_operation(message[0]) + " for id " + message[2] + " is " +
result);

                    out.flush();
                }

                // Handle exceptions
            } catch (IOException e) { //Checking for input output exceptions
                System.out.println("IO Exception:" + e.getMessage());
            } finally { //If socket is not null and request is done, close
the socket
                try {
                    if (clientSocket != null) {
                        clientSocket.close();
                    }
                } catch (IOException e) {
                    // ignore exception on close
                }
            }
    }

    /**
     * To return the value of the operation based on the choice
     * @param i the int choice for the operation
     * @return the value of the operation
     */
    private static String operationId_to_operation(int i) {
        if(i == 1)
        {
            return "addition";
        }
        else if(i==2){
            return "subtraction";
        }
        else
        {
            return "get";
        }
    }

    /**
     * function compute_values
     * The function is used to check whether an id already exists or not.
     * To add one if does not exists and add the sum or subtract
     * If exists then add or subtract to the existing values
     * @param message
     * @return result
     */
    public static int compute_values(int[] message) {
        //checking if Id exists
        if(store_message.get(message[2])!= null)
        {
            //if the choice is 1 then add
            if(message[0] == 1)
            {
                store_message.put(message[2],store_message.get(message[2]) +
message[1]);
            }
            //else subtract
            else if(message[0] == 2)
            {
                store_message.put(message[2], store_message.get(message[2]) -
message[1]);
            }
        }
        else //if if does not exists
        {
            //if choice is 1 add
            if(message[0] == 1)
            {
                store_message.put(message[2], message[1]);
            }
            //if choice is 2 add -ve
            else if(message[0] == 2)
            {
                store_message.put(message[2], -message[1]);
            }
            //if choice is 3 and id doesn't exist then return 0
            else
            {
                store_message.put(message[2], 0);
            }
        }
        //get the result of the id
        int result = store_message.get(message[2]);
        return result;

    }
}
```

Project2Task4ClientScreen:



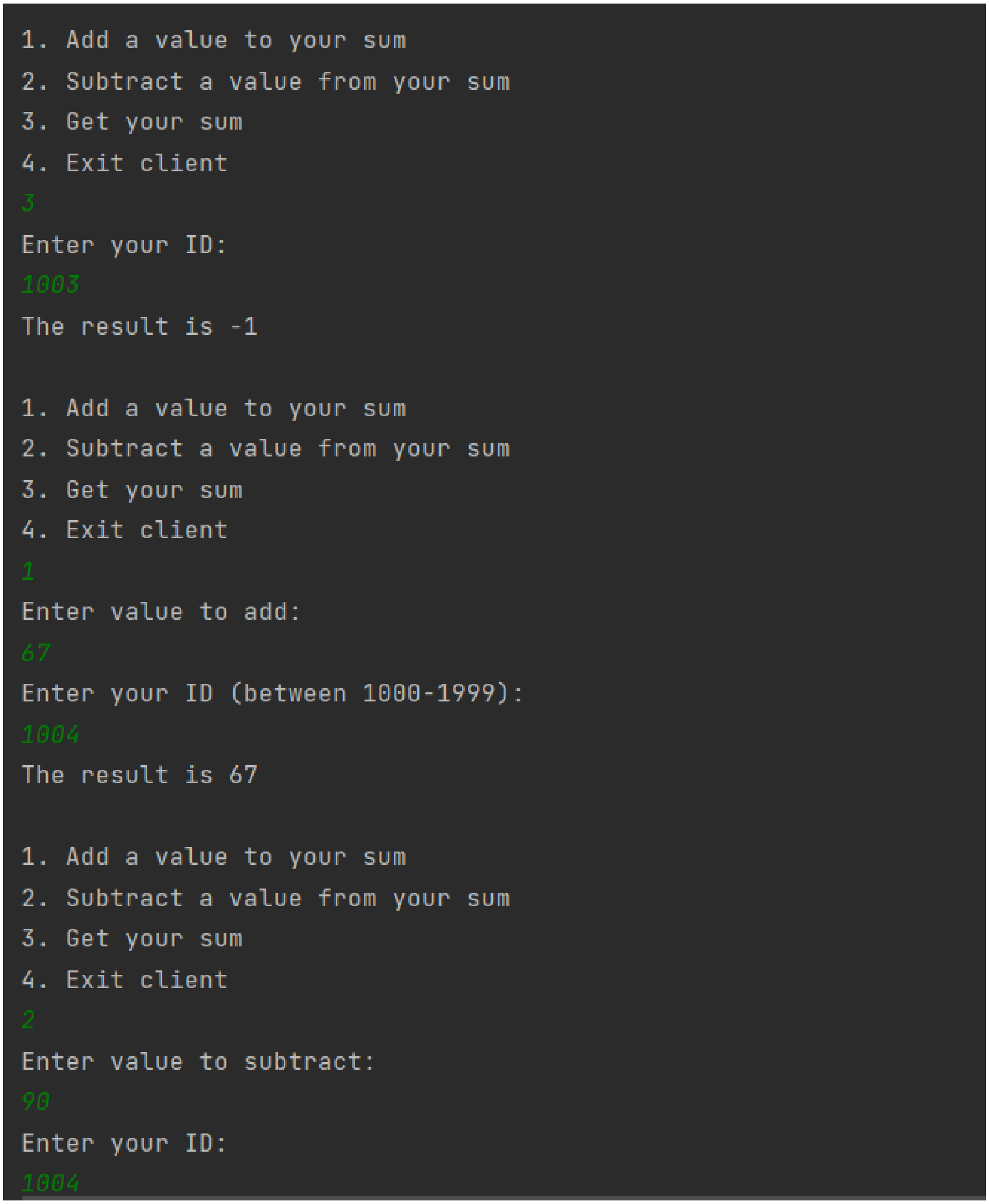
```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client

1
Enter value to add:
2
Enter your ID (between 1000-1999):
1002
The result is 2

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
3
Enter your ID:
1002
The result is -1

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
3
```

Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu



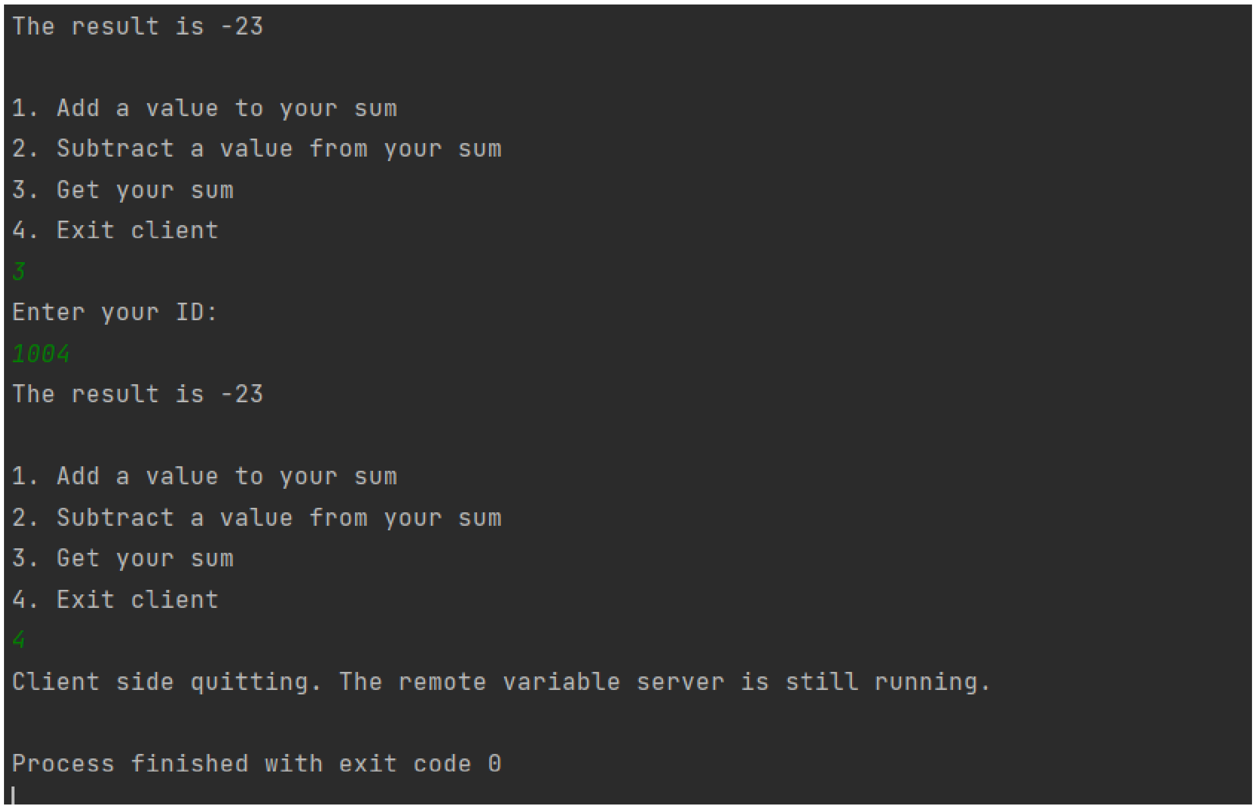
```
4. Exit client
3
Enter your ID:
1002
The result is -1

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
1
Enter value to add:
2
Enter your ID (between 1000-1999):
1003
The result is 2

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
3
Enter your ID:
1003
The result is -1
```

Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu

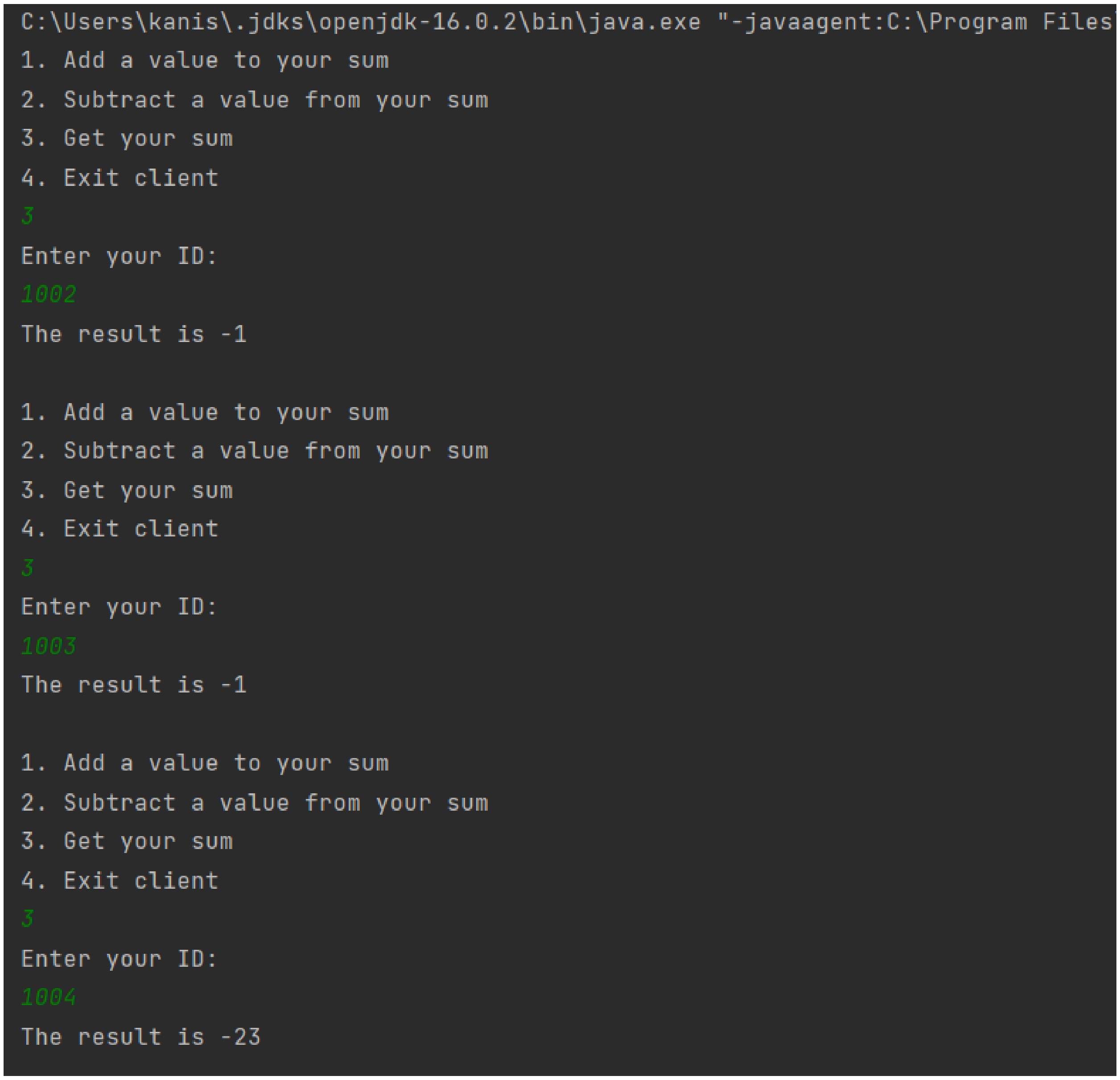


```
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
3
Enter your ID:
1003
The result is -1

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
1
Enter value to add:
67
Enter your ID (between 1000-1999):
1004
The result is 67

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
90
Enter your ID:
1004
```

Name: Kanishka Bhambhani
Email id: kbhambha@andrew.cmu.edu

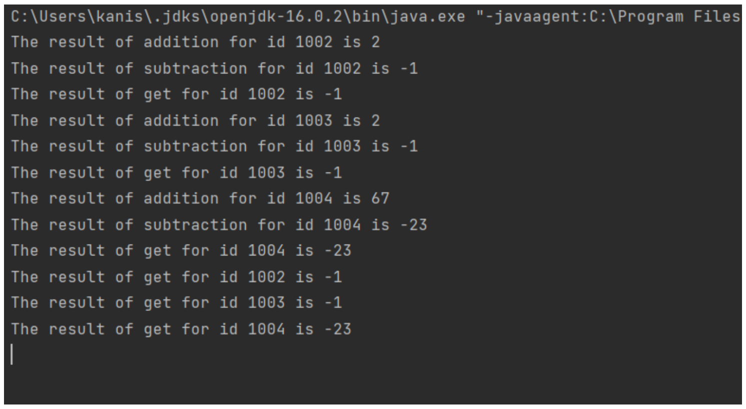


```
The result is -23

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:
1004
The result is -23

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. The remote variable server is still running.

Process finished with exit code 0
```



```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
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:
1002
The result is -1

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:
1003
The result is -1

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:
1004
The result is -23
```

<u>Project2Task4ServerScreen:</u>



```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files
The result of addition for id 1002 is 2
The result of subtraction for id 1002 is -1
The result of get for id 1002 is -1
The result of addition for id 1003 is 2
The result of subtraction for id 1003 is -1
The result of get for id 1003 is -1
The result of addition for id 1004 is 67
The result of subtraction for id 1004 is -23
The result of get for id 1004 is -23
The result of get for id 1002 is -1
The result of get for id 1003 is -1
The result of get for id 1004 is -23
```

Project2Task5

Project2Task5Client:

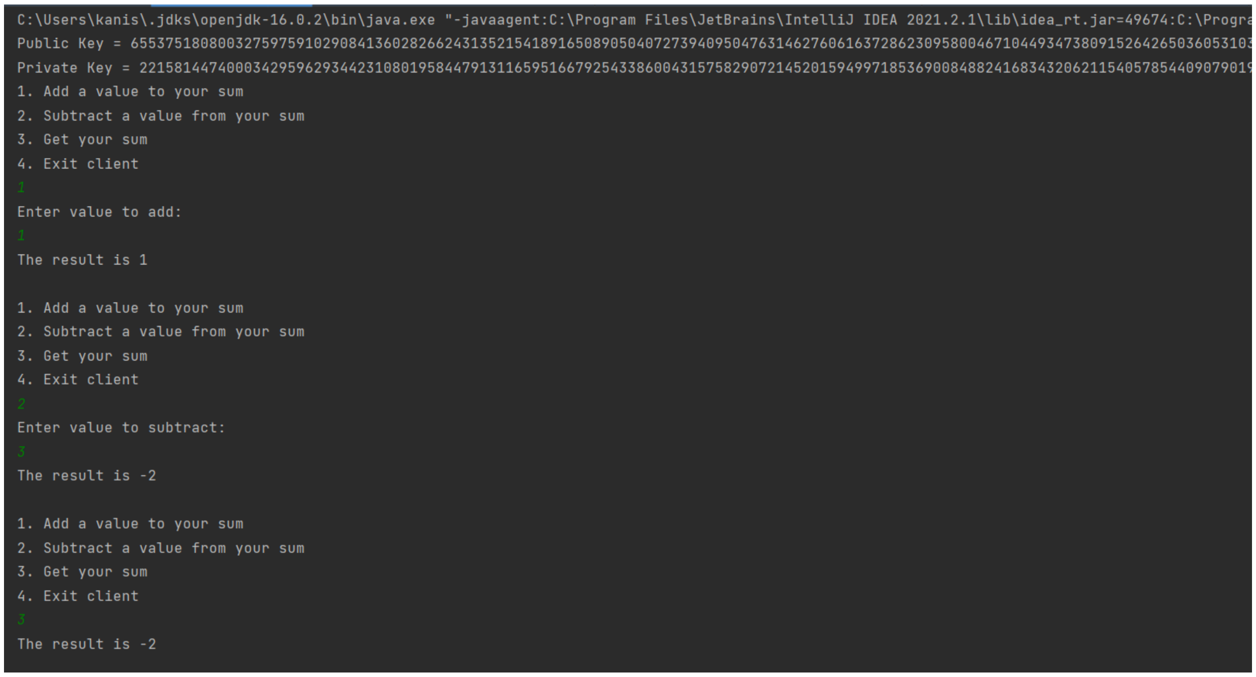
```
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */

import java.math.BigInteger;
import java.net.*;
import java.io.*;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import java.util.Random;
import java.util.Scanner;

/**
 * class RemoteVariableClientTCP
 * TCP Client for sending requests to the server
 */
public class SigningClientTCP {
    /**
     * Connecting and parsing requests to the server
     * @param args
     */
    public static void main(String args[]) {
        BigInteger[] rsa_result = calculateRSA(); //Calculation for public
and private key
        String public_key =
rsa_result[0].toString()+rsa_result[2].toString(); //Public key -
concatenation of e+n
        byte[] hashed_public_key = new byte[0];
        try {
            hashed_public_key = compute_hash(public_key); //hashing the
public key
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace(); //error if algorithm does not exist
        }
        //getting the client id i.e least 20 significant bytes of the hash
        byte[] clientId = Arrays.copyOfRange(hashed_public_key,
hashed_public_key.length-20, hashed_public_key.length);
        BigInteger client_id = new BigInteger(clientId); //converting to
BigInteger
        //storing e, n and d values
        BigInteger e = rsa_result[0];
        BigInteger d = rsa_result[1];
        BigInteger n = rsa_result[2];
        //calling the menu method were data is sent to server
        menu(client_id, e, d, n);
        }
    public static void menu(BigInteger client_id, BigInteger e, BigInteger d,
BigInteger n) {
        int choice = 0, sum_value = 0; //operation and operand
        String result; //result returned by server computation
        String message_encrypt = null; //message that needs to be encrypted
        Scanner input = new Scanner(System.in);
        do {
            Socket clientSocket = null;
            try {
                //server connection establishment
                int serverPort = 7777;
                clientSocket = new Socket("localhost", serverPort);
                BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
                PrintWriter out = new PrintWriter(new BufferedWriter(new
OutputStreamWriter(clientSocket.getOutputStream())));
                //menu choices
                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");
                choice = Integer.parseInt(input.nextLine());
                switch (choice) {
                    case 1: //The message to send if the choice is add
                        System.out.println("Enter value to add:");
                        sum_value = Integer.parseInt(input.nextLine());
                        message_encrypt = client_id + "" + e + "" + n +
                                "" + choice + "" + sum_value;
                        break;
                    case 2: //The message to send if the choice is subtract
                        System.out.println("Enter value to subtract:");
                        sum_value = Integer.parseInt(input.nextLine());
                        message_encrypt = client_id + "" + e + "" + n +
                                "" + choice + "" + sum_value;
                        break;
                    case 3: //The message to send if the choice is get
                        message_encrypt = client_id + "" + e + "" + n +
                                "" + choice;
                        break;
                    case 4: //quitting if user wants to exit
                        System.out.println("Client side quitting. The remote
variable server is still running.");
                        System.exit(0);
                        break;
                    default: System.out.println("Wrong Choice.Try again");
                        continue;
                }
                //hashing and then encrypting the message
                BigInteger encrypted_sign = sign(message_encrypt, d, n);
                //Sending encrypted sign, client id, e, n, choice and value
if choice is 1 or 2 to the server
                out.println(encrypted_sign);
                out.flush();
                out.println(client_id);
                out.flush();
                out.println(e);
                out.flush();
                out.println(n);
                out.flush();
                out.println(choice);
                out.flush();
                if(choice == 1 || choice == 2) {
                    out.println(sum_value);
                    out.flush();
                }
                //recieving the result from the server
                result = in.readLine();
                if(!(result.equalsIgnoreCase("Error in request.")))
                    System.out.println("The result is " + result + "\n");
                else
                    System.out.println("Error in Request.");

            } catch (IOException | NoSuchAlgorithmException ex) { //Exception
for input output
                System.out.println("IO Exception:" + ex.getMessage());
            } finally { //If operation is done and client is not null/closed
                try {
                    if (clientSocket != null) {
                        clientSocket.close();
                    }
                } catch (IOException ex) {
                    // ignore exception on close
                }
            }

        }while(choice < 4);

    }

    /**
     * function calculateRSA()
     * Used the one given in the lab
     * To calculate public and private keys
     * @return e,d,n - to create public and private keys
     */
    public static BigInteger[] calculateRSA()
    {
        BigInteger n; // n is the modulus for both the private and public
keys
        BigInteger e; // e is the exponent of the public key
        BigInteger d; // d is the exponent of the private key
        BigInteger[] rsa = new BigInteger[3];
        Random rnd = new Random();

        //Generating two large random primes.
        BigInteger p = new BigInteger(400, 100, rnd);
        BigInteger q = new BigInteger(400, 100, rnd);

        // Compute n by the equation n = p * q.
        n = p.multiply(q);

        // Compute phi(n) = (p-1) * (q-1)
        BigInteger phi =
(p.subtract(BigInteger.ONE)).multiply(q.subtract(BigInteger.ONE));
```

```
        // Select a small odd integer e that is relatively prime to phi(n).
        e = new BigInteger("65537");

        //Compute d as the multiplicative inverse of e modulo phi(n).
        d = e.modInverse(phi);

        System.out.println("Public Key = " + e+n);  // (e,n) is the RSA
public key
        System.out.println("Private Key = " + d+n);  // (d,n) is the RSA
private keySystem.out.println(" n = " + n);  // Modulus for both keys
        rsa[0] = e;
        rsa[1] = d;
        rsa[2] = n;

        return rsa;
    }

    /**
     * function compute_hash()
     * It is used to hash the values using SHA-256
     * @param public_key_hash
     * @return hash using SHA-256
     * @throws NoSuchAlgorithmException
     */
    public static byte[] compute_hash(String public_key_hash) throws
NoSuchAlgorithmException {
        MessageDigest digest;
        digest = MessageDigest.getInstance("SHA-256");
        //encoding with SHA-256
        byte[] encodedhash = digest.digest(
                public_key_hash.getBytes(StandardCharsets.UTF_8));
        return encodedhash;
    }

    /**
     * function sign()
     * Creating a signature from the client
     * @param message_encrypt message to be encrypted
     * @param d exponent of private key
     * @param n modulus of public and private keys
     * @return the sign of the message almong with the private key
     * @throws NoSuchAlgorithmException
     */
    public static BigInteger sign(String message_encrypt, BigInteger d,
BigInteger n) throws NoSuchAlgorithmException {
        byte[] signed_hash = compute_hash(message_encrypt); //Hashing the
message
        byte[] positive_signed_hash = new byte[signed_hash.length + 1];
        positive_signed_hash[0] = 0;
        //Adding a byte 0 in the start to not have -ve values for RSA
        for(int i = 0, j = 1; i < signed_hash.length; i++, j++)
        {
            positive_signed_hash[j] = signed_hash[i];
        }
        BigInteger m = new BigInteger(positive_signed_hash);
        return  m.modPow(d,n); //Encrypting the message
    }
}
```

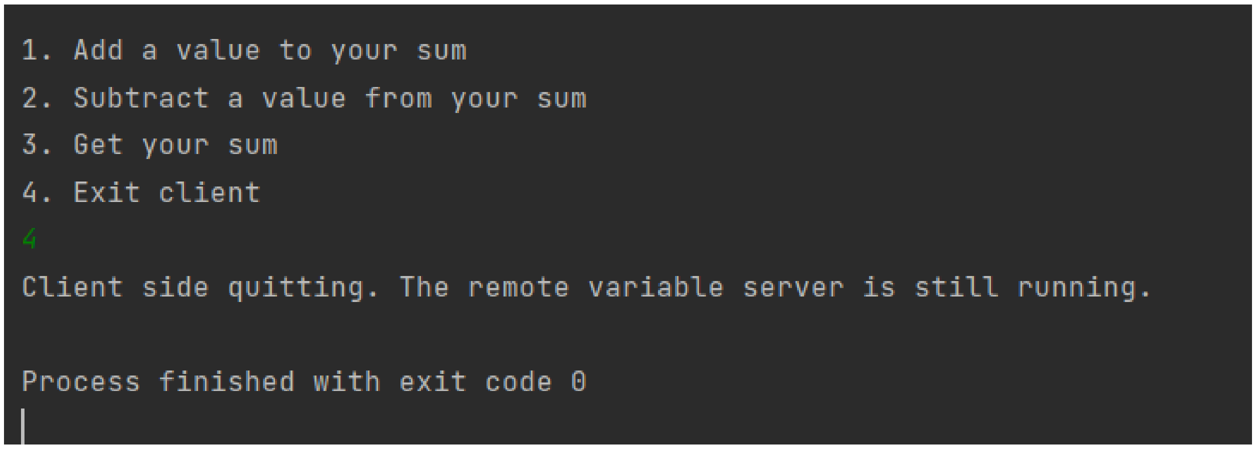
Project2Task5Server:

```
/**
 * Andrew id: kbhambha
 * Author : Kanishka Bhambhani
 */

import java.math.BigInteger;
import java.net.*;
import java.io.*;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

/**
 * class VerifyingServerTCP
 * TCP Server for echoing and replying to the client
 */

public class VerifyingServerTCP {
    //HashMap to store the client id and the value
    public static Map<BigInteger,Integer> store_message = new HashMap<>();
    /**
     * Connection and parsing of messages using UDP
     * @param args no arguments
     */
    public static void main(String args[]) {
        Socket clientSocket = null;
            try {
                int serverPort = 7777; // the server port we are using

                // Create a new server socket
                ServerSocket listenSocket = new ServerSocket(serverPort);
                while (true) {
                    try {
                        clientSocket = listenSocket.accept();
                        //"in" to read from the client socket
                        Scanner in = new
Scanner(clientSocket.getInputStream());
                        //"out" to write to the client socket
                        PrintWriter out;
                        out = new PrintWriter(new BufferedWriter(new
OutputStreamWriter(clientSocket.getOutputStream())));
                        //verifying the message and returning the computation
result
                        verify(in, out);
                    }
                    //In order for the client to continue listening
                    catch(Exception e)
                    {
                        main(null);
                    }
                }

                // Handle exceptions
            } catch (IOException e) { //Checking for input output exceptions
            } finally { //If socket is not null and request is done, close
the socket
                try {
                    if (clientSocket != null) {
                        clientSocket.close();
                    }
                } catch (IOException e) {
                    // ignore exception on close
                }
            }
    }

    /**
     * function compute_result()
     * The function is used to verify whether the public key's least 20 bits
are equal to the client Id
     * The encrypted sign is also decrypted and the message is hashed to
verify if the message is by the same person or not.
     * @param in client's input
     * @param out ouput to client
     * @throws NoSuchAlgorithmException
     */
    public static void verify(Scanner in,PrintWriter out) throws
NoSuchAlgorithmException {
        int sum_value = 0;
        //Getting the message and encrypted sign from the client and storing
them in BigInteger
        String encryption_string = in.nextLine();
        BigInteger encrypted_sign = new BigInteger(encryption_string);
        BigInteger clientId = new BigInteger(in.nextLine());
        String e_string = in.nextLine();
        String n_string = in.nextLine();
        int choice = Integer.parseInt(in.nextLine());
        BigInteger e = new BigInteger(e_string);
        BigInteger n = new BigInteger(n_string);
        //Concatenating e and n for Public key
        String public_key = e_string + n_string;
        //Hashing the public key
        byte[] hashed_public_key = compute_hash(public_key);
        //Substringing the least 20 bits of the public key
        byte[] computed_clientId = Arrays.copyOfRange(hashed_public_key,
hashed_public_key.length - 20, hashed_public_key.length);
        //Storing it in BigInteger
        BigInteger client_id = new BigInteger(computed_clientId);
        String message_encrypt;
        //The message will have the value to be added to the operation based
on the choice
        if (choice == 1 || choice == 2) {
            sum_value = Integer.parseInt(in.nextLine()); //If + or -
            message_encrypt = client_id + "" + e + "" + n +
                    "" + choice + "" + sum_value;
        } else {
            message_encrypt = client_id + "" + e + "" + n + //If get
                    "" + choice;
        }
        //Decrypting the sign
        BigInteger decrypted_sign = (encrypted_sign.modPow(e, n));
        //Hashing the message derived above
        BigInteger hashed_message = hash_message(message_encrypt);
        //Comparing the sign and hashed message, client Id and computed
client ID
        if ((decrypted_sign.equals(hashed_message)) &&
(clientId.equals(client_id))) {
            //If true, performing computation
            int result = compute_values(client_id, choice, sum_value);
            //Sending it back to client
            out.println(result);
            System.out.println("The result of "+
operationId_to_operation(choice) + " for id " + clientId + " is " + result);
        } else { //if not, sending error in request
            out.println("Error in request.");
            System.out.println("Error in request.");
        }
        //flushing the data after sent
        out.flush();
    }

    /**
     * To return the value of the operation based on the choice
     * @param i the int choice for the operation
     * @return the value of the operation
     */
    private static String operationId_to_operation(int i) {
        if(i == 1)
        {
            return "addition";
        }
        else if(i==2){
            return "subtraction";
        }
        else
        {
            return "get";
        }
    }

    /**
     * function compute_values
     * The function is used to check whether an id already exists or not.
     * To add one if does not exists and add the sum or subtract
     * If exists then add or subtract to the existing values
     * @param clientId the id of the client
     * @param choice the operation chosen by client
     * @param sum_value the operand valye
     * @return
     */
    public static int compute_values(BigInteger clientId, Integer choice,
Integer sum_value) {
        //checking if Id exists
        if(store_message.get(clientId)!= null)
        {
            //if the choice is 1 then add
            if(choice == 1)
            {
                store_message.put(clientId,store_message.get(clientId) +
sum_value);
            }
            //else subtract
            else if(choice == 2)
            {
                store_message.put(clientId,store_message.get(clientId) -
sum_value);
            }
        }
        else //if if does not exists
        {
            //if choice is 1 add
            if(choice  == 1)
            {
                store_message.put(clientId, sum_value);
            }
            //if choice is 2 add -ve
            else if(choice == 2)
            {
                store_message.put(clientId, -sum_value);
            }
            //if choice is 3 and id doesn't exist then return 0
            else
            {
                store_message.put(clientId, sum_value);
            }
        }
        //get the result of the id
        int result = store_message.get(clientId);
        return result;

    }

    /**
     * function compute_hash()
     * It is used to hash the values using SHA-256
     * @param public_key_hash
     * @return hash using SHA-256
     * @throws NoSuchAlgorithmException
     */
    public static byte[] compute_hash(String public_key_hash) throws
NoSuchAlgorithmException {
        MessageDigest digest;
        //encoding with SHA-256
        digest = MessageDigest.getInstance("SHA-256");
        byte[] encodedhash = digest.digest(
                public_key_hash.getBytes(StandardCharsets.UTF_8));
        return encodedhash;
    }
    /**
     * function varify()
     * Creating a signature from the client
     * @param message_encrypt message to be hashed and then added the zero
byte to
     * @return the hashed message
     * @throws NoSuchAlgorithmException
     */
    public static BigInteger hash_message(String message_encrypt) throws
NoSuchAlgorithmException {
        byte[] signed_hash = compute_hash(message_encrypt); //Hashing the
message
        byte[] positive_signed_hash = new byte[signed_hash.length + 1];
        positive_signed_hash[0] = 0;
        //Adding a byte 0 in the start to not have -ve values for RSA
        for(int i = 0, j = 1; i < signed_hash.length; i++, j++)
        {
            positive_signed_hash[j] = signed_hash[i];
        }
        BigInteger m = new BigInteger(positive_signed_hash);
        return m;
    }
}
```

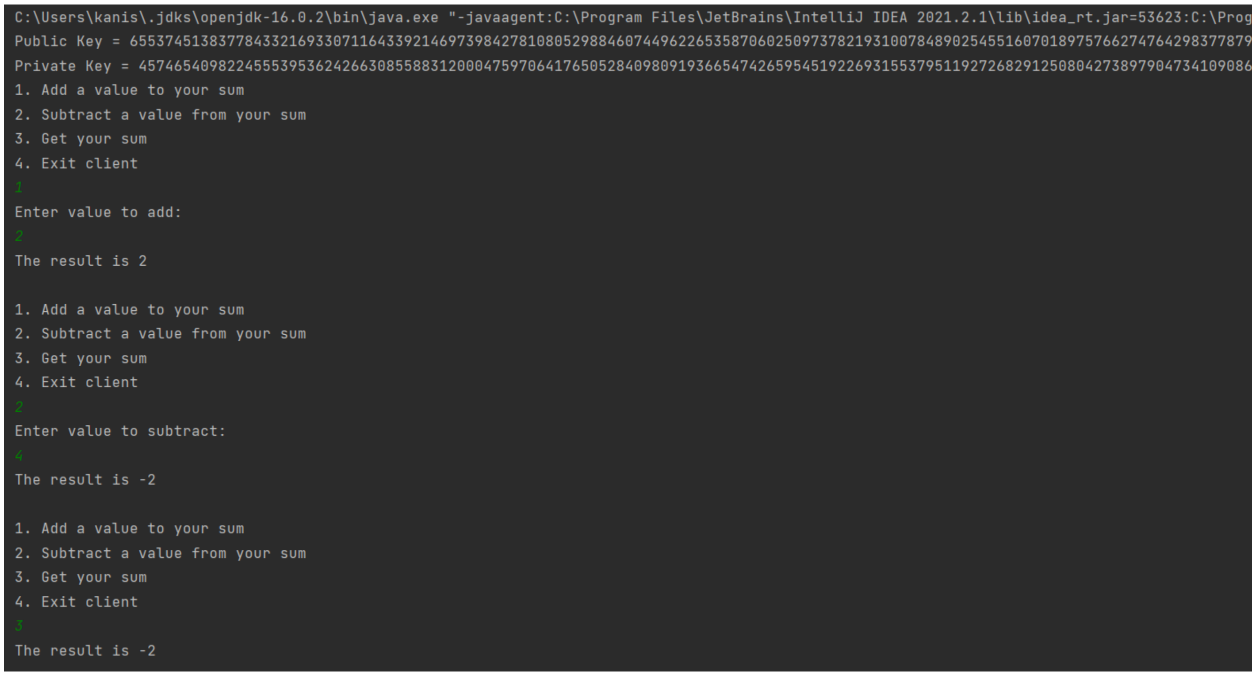
Project2Task5ClientScreen:



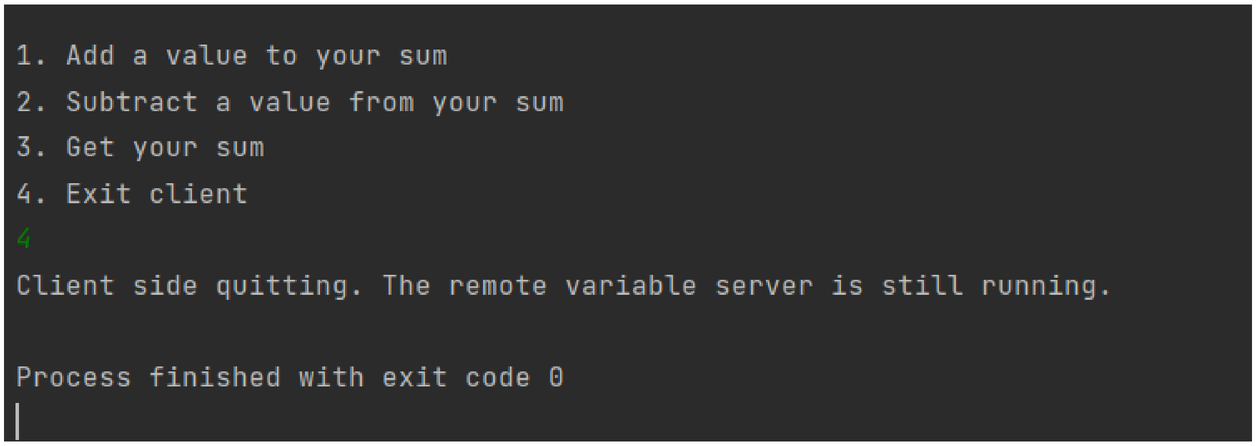
```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021.2.1\lib\idea_rt.jar=49674:C:\Progra
Public Key = 655375180800327597591029084136028266243135215418916508905040727394095047631462760616372862309580046710449347380915264265036053103
Private Key = 22158144740003429596293442310801958447913116595166792543386004315758290721452015949971853690084882416834320621154057854409079019
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
1
Enter value to add:
1
The result is 1

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
3
The result is -2

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
3
The result is -2
```



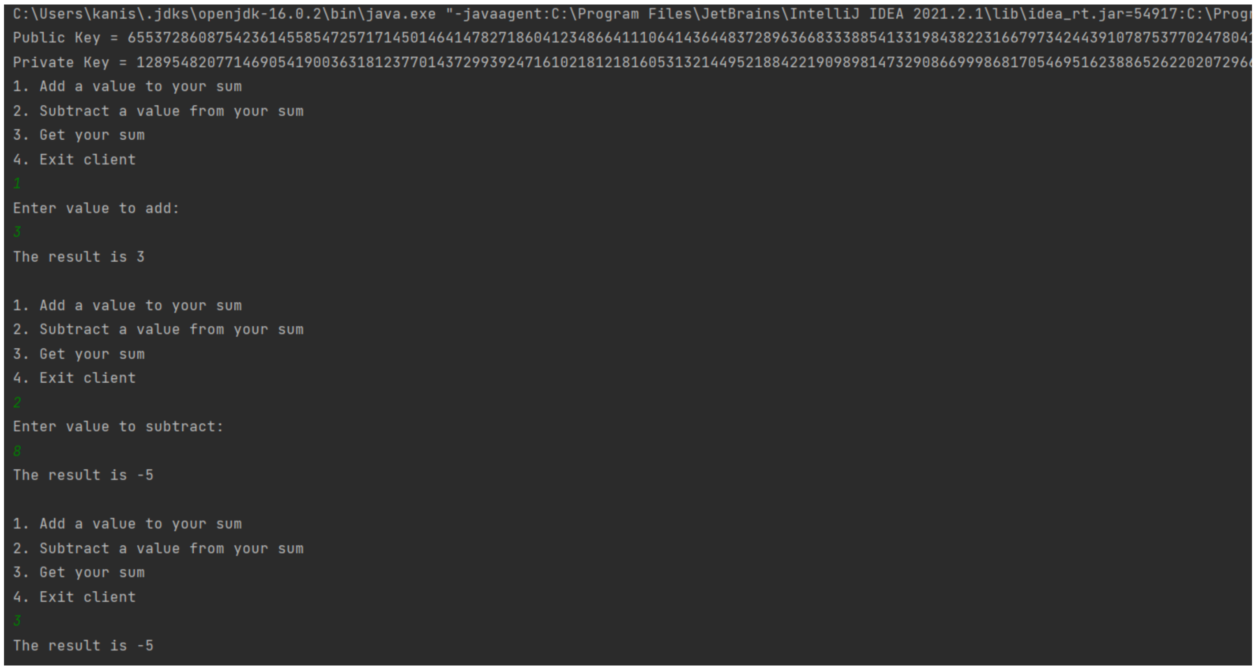
```
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
4
Client side quitting. The remote variable server is still running.

Process finished with exit code 0
```



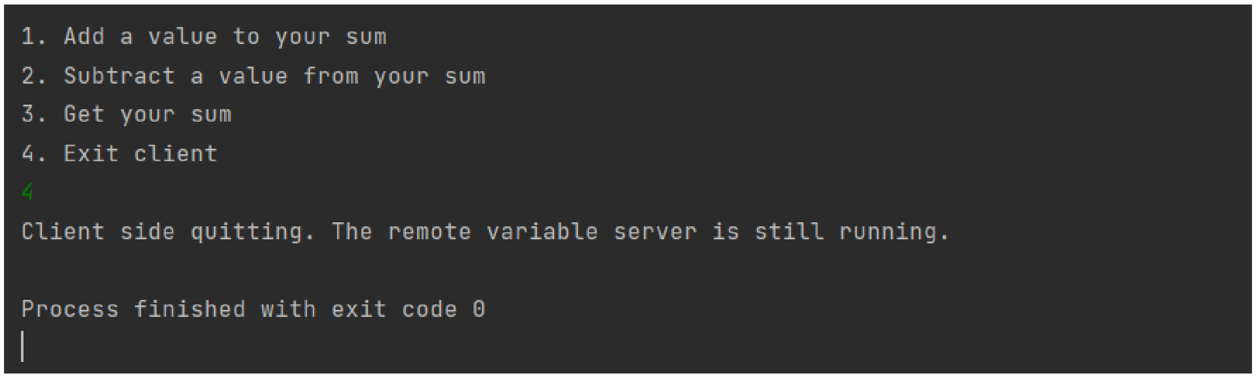
```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021.2.1\lib\idea_rt.jar=53623:C:\Prog
Public Key = 6553745138377843321693307116433921469739842781080529884607449622653587060250973782193100784890254551607018975766274764298377879
Private Key = 457465409822455539536242663085588312000475970641765052840980919366547426595451922693155379511927268291250804273897904734109086
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
1
Enter value to add:
2
The result is 2

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
4
The result is -2

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
3
The result is -2
```



```
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. The remote variable server is still running.

Process finished with exit code 0
```

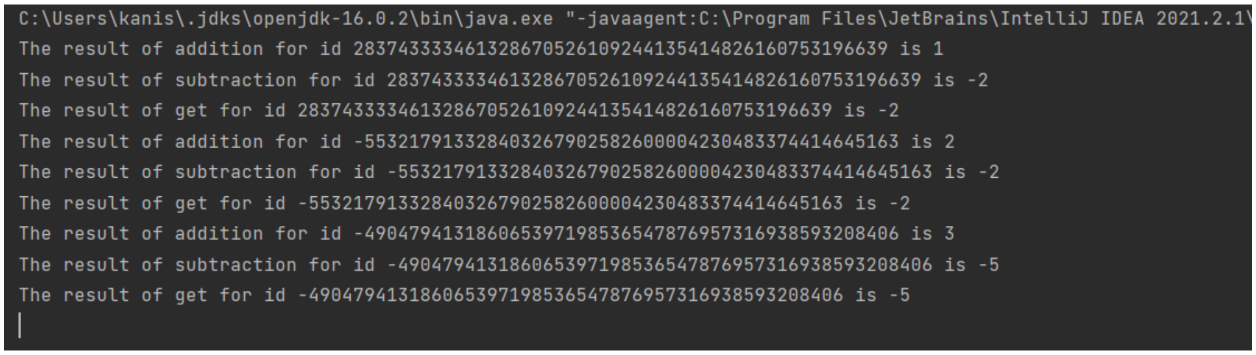


```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021.2.1\lib\idea_rt.jar=54917:C:\Prog
Public Key = 65537286087542361455585472571714501464147827186041234866411106414364483728963668333885413319843822316679734244391078753770247804
Private Key = 1289548207714690541900363181237701437299392471610218121816053132144952188422190989814732908669998681705469516238865262202072966
1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
1
Enter value to add:
3
The result is 3

1. Add a value to your sum
2. Subtract a value from your sum
3. Get your sum
4. Exit client
2
Enter value to subtract:
8
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
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
4
Client side quitting. The remote variable server is still running.

Process finished with exit code 0
```

Project2Task5ServerScreen:



```
C:\Users\kanis\.jdks\openjdk-16.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021.2.1\
The result of addition for id 283743333461328670526109244135414826160753196639 is 1
The result of subtraction for id 283743333461328670526109244135414826160753196639 is -2
The result of get for id 283743333461328670526109244135414826160753196639 is -2
The result of addition for id -553217913328403267902582600004230483374414645163 is 2
The result of subtraction for id -553217913328403267902582600004230483374414645163 is -2
The result of get for id -553217913328403267902582600004230483374414645163 is -2
The result of addition for id -490479413186065397198536547876957316938593208406 is 3
The result of subtraction for id -490479413186065397198536547876957316938593208406 is -5
The result of get for id -490479413186065397198536547876957316938593208406 is -5
```