```
1 import java.net.*;
 2 import java.io.*;
 3 import java.util.Scanner;
 5 public class RemoteVariableClientUDP{
       static InetAddress aHost;
7
       static int serverPort;
 8
       static DatagramSocket aSocket;
       static String replyString = "";
 9
10
11
12
       public static void main(String args[]){
13
           // args give message contents and server
   hostname
           System.out.println("The client is running."
14
   );
15
           try {
16
               // set to localhost to host on local
   machine and set port
17
               aHost = InetAddress.getByName("
   localhost");
18
               serverPort = 6789;
19
20
               // input server port to use
21
               System.out.print("Input a server side
   port number: ");
22
               Scanner readline = new Scanner(System.
   in);
23
               serverPort = readline.nextInt();
24
25
26
               String ID = "";
               String operation = "";
27
28
               String value = "";
29
30
               String packet = "";
31
32
               while (true) {
                   System.out.println("\n1. Add a
33
   value to your sum.\n" +
34
                            "2. Subtract a value from
```

```
34 \text{ your sum.} \text{\n"} +
35
                             "3. Get your sum.\n" +
36
                             "4. Exit client");
37
38
                    int in = readline.nextInt();
39
40
                    if(in == 1){
41
                         operation = "1 ";
                        System.out.println("Enter value
42
    to add: ");
                        value = String.valueOf(readline
43
   .nextInt());
                        System.out.println("Enter your
44
   ID: ");
                        ID = String.valueOf(readline.
45
   nextInt()) + " ";
46
47
                        packet += operation + ID +
   value;
48
49
50
                    else if(in == 2){
51
                        operation = "2 ";
52
                        System.out.println("Enter value
    to subtract: ");
53
                        value = String.valueOf(readline
   .nextInt());
54
                        System.out.println("Enter your
   ID: ");
55
                         ID = String.valueOf(readline.
   nextInt()) + " ";
56
57
                        packet += operation + ID +
   value;
58
59
                    }
60
                    else if(in == 3){
61
                        operation = "3 ";
                        System.out.println("Enter your
62
   ID: ");
                        ID = String.valueOf(readline.
63
```

```
63 nextInt()) + " ";
64
                        value = "";
65
66
                        packet += operation + ID +
   value;
                   }
67
68
                   else if(in == 4){
                        System.out.println("Client
69
   side quitting. The remote variable server is still
    running."); // send reply if not halt
70
                        break;
71
                    }
72
                   System.out.println("The result is
73
     " + communicate(packet));
74
75
                   packet = "";
76
77
               }
78
79
               // catch potential exceptions
           }catch (SocketException e) {System.out.
80
   println("Socket: " + e.getMessage());
           }catch (IOException e){System.out.println(
81
   "IO: " + e.getMessage());
           }finally {if(aSocket != null) aSocket.
82
   close();}
83
       }
84
       public static String communicate(String in)
   throws IOException {
85
86
           aSocket = new DatagramSocket();
87
88
           // operation code
89
           byte [] m = in.qetBytes(); // convert
   console in into byte packets
90
           DatagramPacket operationRequest = new
   DatagramPacket(m, m.length, aHost, serverPort);
           aSocket.send(operationRequest); // send
91
   packet
92
```

```
File - /Users/sairajuladevi/IdeaProjects/Project2Task3/src/main/java/RemoteVariableClientUDP.java
              byte[] buffer = new byte[1000]; // create
 93
     a buffer to receiver server packet
              DatagramPacket reply = new DatagramPacket(
 94
     buffer, buffer.length); // format for receiving
     packet
              aSocket.receive(reply); // receive from
 95
     socket
              replyString = new String(reply.getData()).
 96
     substring(0,reply.getLength()); // get proper
     length of string
 97
              return replyString;
 98
         }
 99
100 }
```

```
1 import java.net.*;
 2 import java.io.*;
 3 import java.util.Scanner;
 4 import java.util.*;
 5
 6 public class RemoteVariableServerUDP{
 7
       private static int sum = 0;
       private static TreeMap<Integer, Integer>
   tree_map = new TreeMap<Integer, Integer>();
10
       static String instructions = "1. Add a value to
    your sum.\n" +
11
               "2. Subtract a value from your sum.\n"
               "3. Get your sum.\n" +
12
13
               "4. Exit client";
14
15
       public static void main(String args[]){
           System.out.println("The server is running."
16
   ); // lab instructions
17
           DatagramSocket aSocket = null;
18
           byte[] buffer = new byte[1000]; // set up
   packet buffer for client message
19
           try{
20
               // set up ports
21
               System.out.print("Input a server port
   number to listen on: ");
22
               Scanner readline = new Scanner(System.
   in);
23
               int serverPort = readline.nextInt();
   // convert to int
24
25
               // set up sockets to receive client
   packets
26
               aSocket = new DatagramSocket(serverPort
   );
27
               aSocket.setReuseAddress(true);
28
29
               DatagramPacket request = new
   DatagramPacket(buffer, buffer.length); // syntax
   for buffer
```

```
30
31
               int id;
32
               int value;
33
34
               while(true){ // loop to continue until
    'halt!' is sent
35
36
                   // operation request and reply
37
                   aSocket.receive(request); //
   receive request, and format a reply
38
                    String requestString = new String(
   request.qetData()).substring(0,request.getLength
   ()).strip(); // proper length
39
40
                    String[] arrRequestString =
   requestString.split(" ");
41
42
                    id = Integer.valueOf(
   arrRequestString[1]);
43
44
                    if(!tree_map.containsKey(id)){
45
                        if(arrRequestString[0].equals("
   1")){
46
                            value = Integer.valueOf(
   arrRequestString[2]);
47
                            tree_map.put(id, value);
48
                        }
49
                        else if(arrRequestString[0].
   equals("2")){
50
                            value = -1*Integer.valueOf(
   arrRequestString[2]);
51
                            tree_map.put(id, value);
                        }
52
53
                    }
                    else{
54
55
                        if(arrRequestString[0].equals("
   1")){
56
                            value = Integer.valueOf(
   arrRequestString[2]);
57
                            tree_map.replace(id,
   tree_map.get(id)+value);
```

```
58
59
                        else if(arrRequestString[0].
   equals("2")){
60
                            value = Integer.valueOf(
   arrRequestString[2]);
61
                            tree_map.replace(id,
   tree_map.get(id)-value);
62
                        }
63
                   }
64
65
                   System.out.println("\nVisitorID: "
    + arrRequestString[1]);
                   System.out.println("Operation #: "
66
    + arrRequestString[0]);
                   System.out.println("Return Variable
67
    " + String.valueOf(tree_map.get(id)));
68
69
70
                   byte [] m = (String.valueOf(
   tree_map.get(id))).getBytes(); // convert console
   in into byte packets
71
72
                   DatagramPacket reply = new
   DatagramPacket(m, m.length, request.getAddress(),
   request.getPort());
73
                   aSocket.send(reply); // send back
   the reply
74
75
76
               // catch potential exceptions
77
78
           }catch (SocketException e){System.out.
   println("Socket: " + e.getMessage());
79
           }catch (IOException e) {System.out.println(
   "IO: " + e.getMessage());
           }finally {if(aSocket != null) aSocket.close
80
   ();}
81
       }
82 }
```