

Name: Muyu Tong

Email: muyut@andrew.cmu.edu

Project 2 Task 0

Project2Task0Client

```
1  //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2  public class EchoClientUDP{
3      public static void main(String args[]){
4          // args give message contents and server hostname
5          System.out.println("The client is running");
6          DatagramSocket aSocket = null;
7          try {
8              InetAddress aHost =
9              InetAddress.getBy_name("localhost");
10
11              //prompt the port number
12              Scanner scanner = new Scanner(System.in);
13              System.out.println("Enter the port number: ");
14              int serverPort = scanner.nextInt();
15
16              aSocket = new DatagramSocket();
17              String nextLine;
18              BufferedReader typed = new BufferedReader(new
19              InputStreamReader(System.in));
20              while ((nextLine = typed.readLine()) != null) {
21                  //each loop read from the console and send it to
22                  the server
23                  byte [] m = nextLine.getBytes();
24                  DatagramPacket request = new DatagramPacket(m,
25                  m.length, aHost, serverPort);
26                  aSocket.send(request);
27                  byte[] buffer = new byte[1000];
```

```

24         DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
25         aSocket.receive(reply);
26         byte[] message = new byte[reply.getLength()];
27         System.arraycopy(reply.getData(), 0, message, 0,
message.length);
28         System.out.println("Reply: " + new
String(message));
29         if (new String(message).equals("halt!")) {
30             System.out.println("Client side quitting");
31             break;
32         }
33     }
34
35     }catch (SocketException e) {System.out.println("Socket: "
+ e.getMessage());
36     }catch (IOException e){System.out.println("IO: " +
e.getMessage());
37     }finally {if(aSocket != null) aSocket.close();}
38 }
39 }
40

```

Project2Task0Server

```

1 //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2 public class EchoServerUDP{
3     public static void main(String args[]){
4         System.out.println("The server is running");
5         DatagramSocket aSocket = null;
6         byte[] buffer = new byte[1000];
7         try{
8             //prompt the port number
9             Scanner scanner = new Scanner(System.in);

```

```

10         System.out.println("Enter the port number: ");
11         int serverPort = scanner.nextInt();
12         scanner.close();
13
14         aSocket = new DatagramSocket(serverPort);
15         DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
16         while(true){
17             //each round receive a request from the client
18             aSocket.receive(request);
19             DatagramPacket reply = new
DatagramPacket(request.getData(),
20                 request.getLength(), request.getAddress(),
request.getPort());
21             byte[] message = new byte[request.getLength()];
22             System.arraycopy(request.getData(), 0, message, 0,
message.length);
23             String requestString = new String(message);
24             System.out.println("Echoing: "+requestString);
25             aSocket.send(reply);
26             if (requestString.equals("halt!")) {
27                 System.out.println("Server side quitting");
28                 break;
29             }
30         }
31     }catch (SocketException e){System.out.println("Socket: " +
e.getMessage());
32     }catch (IOException e) {System.out.println("IO: " +
e.getMessage());
33     }finally {if(aSocket != null) aSocket.close();}
34 }
35 }

```

Project2Task0ClientConsole

```
The client is running
Enter the port number:
6789
1
Reply: 1
2
Reply: 2
3
Reply: 3
4
Reply: 4
5
Reply: 5
halt!
Reply: halt!
Client side quitting

Process finished with exit code 0
```

Project2Task0ServerConsole

```
The server is running
Enter the port number:
6789
Echoing: 1
Echoing: 2
Echoing: 3
Echoing: 4
Echoing: 5
Echoing: halt!
Server side quitting

Process finished with exit code 0
```

Project 2 Task 1

EavesdropperUDP.java

```
1  //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2  public class EavesdropperUDP {
3      DatagramSocket aSocket;
4      DatagramSocket bSocket;
5      int masPort;
6
7      public EavesdropperUDP() throws SocketException {
8          System.out.println("Eavesdropper is running");
9          //prompt the port number
10         Scanner scanner = new Scanner(System.in);
```

```
11         System.out.println("Enter the masquerading port number:");
12         masPort = scanner.nextInt(); scanner.nextLine();
13         System.out.println("Enter the port number: ");
14         int portNum = scanner.nextInt(); scanner.nextLine();
15         scanner.close();
16
17         aSocket = new DatagramSocket(portNum); //socket listen to
18         6798
19         bSocket = new DatagramSocket();
20
21         byte[] buffer = new byte[1000];
22         while (true) {
23             //get request from client
24             DatagramPacket request = new DatagramPacket(buffer,
25             buffer.length);
26             try {
27                 aSocket.receive(request);
28                 byte[] message = new byte[request.getLength()];
29                 System.arraycopy(request.getData(), 0, message, 0,
30                 message.length);
31                 System.out.println("Eavesdrop from client: " + new
32                 String(message));
33                 } catch (IOException e) {
34                     e.printStackTrace();
35                 }
36
37             //transmit the data to server
38             DatagramPacket transmit = new
39             DatagramPacket(request.getData(),
40             request.getLength(), request.getAddress(),
41             masPort);
42             try {
43                 bSocket.send(transmit);
44             } catch (IOException ioException) {
45                 ioException.printStackTrace();
46             }
```

```
40         }
41
42         //receive reply from the server and sent it back to
the client
43         DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
44         try {
45             bSocket.receive(reply);
46             byte[] message = new byte[reply.getLength()];
47             System.arraycopy(reply.getData(), 0, message, 0,
message.length);
48             System.out.println("Eavesdrop from server: " + new
String(message));
49             reply.setPort(request.getPort());
50             aSocket.send(reply);
51         } catch (IOException ioException) {
52             ioException.printStackTrace();
53         }
54     }
55 }
56
57 public static void main(String[] args) {
58     try {
59         new EavesdropperUDP();
60     } catch (SocketException e) {
61         e.printStackTrace();
62     }
63 }
64
65 }
```


Project2Task1ThreeConsoles

```
The server is running
Enter the port number:
6789
Echoing: hi
Echoing: hello
Echoing: halt!
Server side quitting
```

```
Process finished with exit code 0
```

```
!
```

```
The client is running
Enter the port number:
6798
hi
Reply: hi
hello
Reply: hello
halt!
Reply: halt!
Client side quitting
```

```
Process finished with exit code 0
```

```
Eavesdropper is running
Enter the masquerading port number:
6789
Enter the port number:
6798
Eavesdrop from client: hi
Eavesdrop from server: hi
Eavesdrop from client: hello
Eavesdrop from server: hello
Eavesdrop from client: halt!
Eavesdrop from server: halt!
|
```

Project 2 Task 2

Project2Task2Client

```
1 //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2 public class AddingClientUDP {
3     public static DatagramSocket socket;
4     public static int portNum;
5     public static void main(String[] args) {
6         System.out.println("The client is running.");
7         try {
8             socket = new DatagramSocket();
9             System.out.println("Please enter server port: ");
10            Scanner scanner = new Scanner(System.in);
11            portNum = scanner.nextInt();scanner.nextLine();
12
13            String s;
14            while (scanner.hasNextLine()) {
```

```

15         s = scanner.nextLine();
16         if (s.equals("halt!")) {
17             System.out.println("Client side quitting.");
18             break;
19         }
20         int num = Integer.parseInt(s);
21         int res = add(num);
22         System.out.println("The server returned " + res +
".");
23     }
24     } catch (SocketException e) {
25         e.printStackTrace();
26     }
27 }
28 public static int add(int i) {
29     String num = String.valueOf(i);
30     byte[] m = num.getBytes();
31     byte[] buffer = new byte[1000];
32     try {
33         DatagramPacket request = new DatagramPacket(m,
m.length, InetAddress.getByName("localhost"), portNum);
34         socket.send(request);
35         DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
36         socket.receive(reply);
37         byte[] message = new byte[reply.getLength()];
38         System.arraycopy(reply.getData(), 0, message, 0,
reply.getLength());
39         String res = new String(message);
40         return Integer.parseInt(res);
41     } catch (IOException e) {
42         e.printStackTrace();
43     }
44     return 0;
45 }
46 }

```

Project2Task2Server

```
1  //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2  public class AddingServerUDP {
3      public static int sum = 0;
4      public static void main(String[] args) {
5          DatagramSocket socket = null;
6          try {
7              socket = new DatagramSocket(6789);
8          } catch (SocketException e) {
9              e.printStackTrace();
10         }
11         while (true) {
12             byte[] buffer = new byte[1000];
13             DatagramPacket request = new DatagramPacket(buffer,
14 buffer.length);
15             try {
16                 socket.receive(request);
17             } catch (IOException e) {
18                 e.printStackTrace();
19             }
20             byte[] message = new byte[request.getLength()];
21             System.arraycopy(request.getData(), 0, message, 0,
22 request.getLength());
23             int num = Integer.parseInt(new String(message));
24             System.out.printf("Adding %d to %d\n", num, sum);
25             add(num);
26             System.out.printf("Returning sum of %d to client\n",
27 sum);
28             byte[] res = String.valueOf(sum).getBytes();
29             DatagramPacket reply = new DatagramPacket(res,
30 res.length, request.getAddress(), request.getPort());
31             try {
32                 socket.send(reply);
33             } catch (IOException e) {
34                 e.printStackTrace();
35             }
36         }
37     }
38 }
```

```
30         } catch (IOException e) {
31             e.printStackTrace();
32         }
33     }
34 }
35 public static void add(int i) {
36     sum += i;
37 }
38 }
```

Project2Task2ClientConsole

The client is running.

Please enter server port:

6789

1

The server returned 1.

2

The server returned 3.

-3

The server returned 0.

4

The server returned 4.

5

The server returned 9.

halt!

Client side quitting.

Process finished with exit code 0

.

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

Project2Task2ServerConsole

```
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
```


Project 2 Task 3

Project2Task3Client

```
1  //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2  public class RemoteVariableClientUDP {
3      public static DatagramSocket socket;
4      public static int portNum;
5
6      public RemoteVariableClientUDP() {
7      }
8
9      public static void main(String[] args) {
10         System.out.println("The client is running.");
11
12         try {
13             socket = new DatagramSocket();
14             System.out.println("Please enter server port: ");
15             Scanner scanner = new Scanner(System.in);
16             portNum = scanner.nextInt();
17             scanner.nextLine();
18
19             while(true) {
20                 System.out.println("1. Add a value to your
sum.\n2. Subtract a value from your sum.\n3. Get your sum.\n4.
Exit client");
21                 String s = scanner.nextLine();
22                 int choice = Integer.parseInt(s);
23                 int num;
24                 int res;
25                 int id;
26                 if (choice == 1) {
27                     System.out.println("Enter the value to add:
");
28                     num = Integer.parseInt(scanner.nextLine());
```

```
29         System.out.println("Enter your ID: ");
30         id = Integer.parseInt(scanner.nextLine());
31         res = add(num, id);
32     } else if (choice == 2) {
33         System.out.println("Enter the value to
subtract:");
34         num = Integer.parseInt(scanner.nextLine());
35         System.out.println("Enter your ID: ");
36         id = Integer.parseInt(scanner.nextLine());
37         res = subtract(num, id);
38     } else {
39         if (choice != 3) {
40             return;
41         }
42
43         System.out.println("Enter your ID: ");
44         id = Integer.parseInt(scanner.nextLine());
45         res = get(id);
46     }
47
48     System.out.println("The result is: " + res);
49     System.out.println();
50 }
51 } catch (SocketException var7) {
52     var7.printStackTrace();
53 }
54 }
55
56 public static int add(int i, int id) {
57     String request = "\"add\"," + i + "," + id;
58     return request(request);
59 }
60
61 public static int subtract(int i, int id) {
62     String request = "\"subtract\"," + i + "," + id;
63     return request(request);
```

```

64     }
65
66     public static int get(int id) {
67         String request = "\"get\",0,\" + id;
68         return request(request);
69     }
70
71     public static int request(String requestBody) {
72         byte[] m = requestBody.getBytes();
73         byte[] buffer = new byte[1000];
74
75         try {
76             DatagramPacket request = new DatagramPacket(m,
77 m.length, InetAddress.getByName("localhost"), portNum);
78             socket.send(request);
79             DatagramPacket reply = new DatagramPacket(buffer,
80 buffer.length);
81             socket.receive(reply);
82             byte[] message = new byte[reply.getLength()];
83             System.arraycopy(reply.getData(), 0, message, 0,
84 reply.getLength());
85             String res = new String(message);
86             System.out.println(res);
87             return Integer.parseInt(res);
88         } catch (IOException var7) {
89             var7.printStackTrace();
90             return 0;
91         }
92     }
93 }
94
95

```

Project2Task3Server

```
1 //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2 public class RemoteVariableServerUDP {
3     public static TreeMap<Integer, Integer> map = new TreeMap<>();
4     public static void main(String[] args) {
5         DatagramSocket socket = null;
6         try {
7             socket = new DatagramSocket(6789);
8         } catch (SocketException e) {
9             e.printStackTrace();
10        }
11        while (true) {
12            byte[] buffer = new byte[1000];
13            DatagramPacket request = new DatagramPacket(buffer,
14            buffer.length);
15            try {
16                socket.receive(request);
17            } catch (IOException e) {
18                e.printStackTrace();
19            }
20            byte[] message = new byte[request.getLength()];
21            System.arraycopy(request.getData(), 0, message, 0,
22            request.getLength());
23            String requestBody = new String(message);
24            String[] params = requestBody.split(",");
25            String method = params[0];
26            int num = Integer.parseInt(params[1]);
27            int id = Integer.parseInt(params[2]);
28            int sum = exec(method, id, num);
29
30            byte[] res = String.valueOf(sum).getBytes();
31            DatagramPacket reply = new DatagramPacket(res,
32            res.length, request.getAddress(), request.getPort());
33            try {
```

```

33         socket.send(reply);
34     } catch (IOException e) {
35         e.printStackTrace();
36     }
37 }
38 }
39
40 public static int exec(String method, int id, int num) {
41     if (!map.containsKey(id)) map.put(id, 0);
42     int res;
43     if (method.equals("get")) {
44         res = get(id);
45     } else if (method.equals("add")) {
46         res = add(id, num);
47     } else {
48         res = subtract(id, num);
49     }
50     System.out.printf("Visitor id: %d, method: %s, returned
value: %d\n", id, method, res);
51     return map.get(id);
52 }
53
54 public static int add(int id, int num) {
55     map.put(id, map.get(id)+num);
56     return map.get(id);
57 }
58
59 public static int subtract(int id, int num) {
60     map.put(id, map.get(id)-num);
61     return map.get(id);
62 }
63
64 public static int get(int id) {
65     return map.get(id);
66 }
67 }

```

Project2Task3ClientConsole

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

1

Enter the value to add:

1

Enter your ID:

1

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 the value to subtract:

2

Enter your ID:

1

-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

3

Enter your ID:

1

-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

4

Process finished with exit code 0

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

1

Enter the value to add:

5

Enter your ID:

2

5

The result is: 5

1. Add a value to your sum.
2. Subtract a value from your sum.
3. Get your sum.

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

1

Enter the value to add:

10

Enter your ID:

3

10

The result is: 10

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:

20

Enter your ID:

3

-10

The result is: -10

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:

3

-10

The result is: -10

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

4

Process finished with exit code 0

```

3. Get your sum.
4. Exit client
2
Enter the value to subtract:
10
Enter your ID:
2
-5
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:
2
-5
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

Process finished with exit code 0

```

```

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:
1
-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
4

Process finished with exit code 0

```

```

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:
3
-10
The result is: -10

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

Process finished with exit code 0

```

```

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:
2
-5
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

Process finished with exit code 0

```


Project2Task3ServerConsole

```
Visitor id: 1, method: add, returned value: 1
Visitor id: 1, method: subtract, returned value: -1
Visitor id: 1, method: get, returned value: -1
Visitor id: 2, method: add, returned value: 5
Visitor id: 2, method: subtract, returned value: -5
Visitor id: 2, method: get, returned value: -5
Visitor id: 3, method: add, returned value: 10
Visitor id: 3, method: subtract, returned value: -10
Visitor id: 3, method: get, returned value: -10
Visitor id: 1, method: get, returned value: -1
Visitor id: 2, method: get, returned value: -5
Visitor id: 3, method: get, returned value: -10
```

Project 2 Task 4

Project2Task4Client

```
1 //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2 public class RemoteVariableClientTCP {
3     public static Socket socket;
4     public static int portNum = 0;
5     public static void main(String[] args) {
6         System.out.println("The client is running.");
7         System.out.println("Please enter server port: ");
8         Scanner scanner = new Scanner(System.in);
9         portNum = scanner.nextInt();
10        scanner.nextLine();
11
12        String s;
13        while (true) {
14            System.out.println("""
15                1. Add a value to your sum.
16                2. Subtract a value from your sum.
17                3. Get your sum.
```

```
18         4. Exit client""");
19         s = scanner.nextLine();
20         int choice = Integer.parseInt(s);
21
22         int num;
23         int res;
24         int id;
25         if (choice == 1) {
26             System.out.println("Enter the value to add: ");
27             num = Integer.parseInt(scanner.nextLine());
28             System.out.println("Enter your ID: ");
29             id = Integer.parseInt(scanner.nextLine());
30             res = add(num, id);
31         } else if (choice == 2) {
32             System.out.println("Enter the value to
subtract:");
33             num = Integer.parseInt(scanner.nextLine());
34             System.out.println("Enter your ID: ");
35             id = Integer.parseInt(scanner.nextLine());
36             res = subtract(num, id);
37         } else if (choice == 3) {
38             System.out.println("Enter your ID: ");
39             id = Integer.parseInt(scanner.nextLine());
40             res = get(id);
41         } else {
42             return;
43         }
44
45         System.out.println("The result is: " + res);
46         System.out.println();
47     }
48 }
49
50 public static int add(int i, int id) {
51     String request = "add,"+i+","+id;
52     return request(request);
```

```

53     }
54
55     public static int subtract(int i, int id) {
56         String request = "subtract,"+i+", "+id;
57         return request(request);
58     }
59
60     public static int get(int id) {
61         String request = "get,"+0+", "+id;
62         return request(request);
63     }
64
65     public static int request(String requestBody) {
66         try {
67             socket = new Socket("localhost", portNum);
68             BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
69             PrintWriter out = new PrintWriter(new
BufferedWriter(new OutputStreamWriter(socket.getOutputStream())));
70             out.println(requestBody);
71             out.println();
72             out.flush();
73             String res = in.readLine();
74             System.out.println(res);
75             return Integer.parseInt(res);
76         } catch (IOException e) {
77             e.printStackTrace();
78         }
79         return 0;
80     }
81 }
82

```

```
1 //source https://github.com/CMU-Heinz-95702/Project-2-Client-Server
2 public class RemoteVariableServerTCP {
3     public static TreeMap<Integer, Integer> map = new TreeMap<>();
4     public static void main(String[] args) {
5         Socket socket = null;
6         ServerSocket listenSocket = null;
7         try {
8             listenSocket = new ServerSocket(6789);
9         } catch (IOException e) {
10             e.printStackTrace();
11         }
12
13         while (true) {
14             try {
15                 socket = listenSocket.accept();
16                 Scanner in = new Scanner(socket.getInputStream());
17                 StringBuilder request = new StringBuilder();
18                 String s;
19                 while (in.hasNextLine()) {
20                     s = in.nextLine();
21                     if (s.equals("")) {
22                         String requestBody = request.toString();
23
24                         int sum = exec(requestBody);
25
26                         byte[] res =
27 String.valueOf(sum).getBytes();
28
29                         PrintWriter out = new PrintWriter(new
30 BufferedWriter(new OutputStreamWriter(socket.getOutputStream())));
31
32                         out.println(new String(res));
33                         out.flush();
34                         socket.close();
35                     } else {
36                         System.out.println(s);
37                         request.append(s);
38                     }
39                 }
40             } catch (Exception e) {
41                 e.printStackTrace();
42             }
43         }
44     }
45 }
```

```

34         }
35     }
36     } catch (IOException e) {
37         e.printStackTrace();
38     }
39 }
40 }
41
42 public static int exec(String requestBody) {
43     String[] params = requestBody.split(",");
44     String method = params[0];
45     int num = Integer.parseInt(params[1]);
46     int id = Integer.parseInt(params[2]);
47     if (!map.containsKey(id)) map.put(id, 0);
48     int res;
49     if (method.equals("get")) {
50         res = get(id);
51     } else if (method.equals("add")) {
52         res = add(id, num);
53     } else {
54         res = subtract(id, num);
55     }
56     System.out.printf("Visitor id: %d, method: %s, returned
57 value: %d\n", id, method, res);
58     return map.get(id);
59 }
60
61 public static int add(int id, int num) {
62     map.put(id, map.get(id)+num);
63     return map.get(id);
64 }
65
66 public static int subtract(int id, int num) {
67     map.put(id, map.get(id)-num);
68     return map.get(id);
69 }

```

```

69
70     public static int get(int id) {
71         return map.get(id);
72     }
73 }
74

```

Project2Task4ClientConsole

6789

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:

2

Enter your ID:

1

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 the value to subtract:

4

Enter your ID:

1

-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

3

Enter your ID:

1

-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

4

Process finished with exit code 0

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

1

Enter the value to add:

6

Enter your ID:

3

6

The result is: 6

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:

12

Enter your ID:

3

-6

The result is: -6

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:

3

-6

The result is: -6

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

4

Process finished with exit code 0

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

1

Enter the value to add:

4

Enter your ID:

2

4

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 the value to subtract:

8

Enter your ID:

2

-4

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

3

Enter your ID:

2

-4

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

4

Process finished with exit code 0

<pre>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: 3 -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 4 Process finished with exit code 0</pre>	<pre>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: 3 -6 The result is: -6 1. Add a value to your sum. 2. Subtract a value from your sum. 3. Get your sum. 4. Exit client 4 Process finished with exit code 0</pre>	<pre>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: 2 -4 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 4 Process finished with exit code 0</pre>
--	--	--

Project2Task4ServerConsole

```
Visitor id: 1, method: add, returned value: 2
Visitor id: 1, method: subtract, returned value: -2
Visitor id: 1, method: get, returned value: -2
Visitor id: 2, method: add, returned value: 4
Visitor id: 2, method: subtract, returned value: -4
Visitor id: 2, method: get, returned value: -4
Visitor id: 3, method: add, returned value: 6
Visitor id: 3, method: subtract, returned value: -6
Visitor id: 3, method: get, returned value: -6
Visitor id: 1, method: get, returned value: -2
Visitor id: 2, method: get, returned value: -4
Visitor id: 3, method: get, returned value: -6
```

Project 2 Task 5

Project2Task5Client

```
1 public class SigningClientTCP {
2     public Socket socket;
3     public int portNum;
```

```

4      RSA rsa;
5      String id;
6      Scanner scanner;
7      public SigningClientTCP() {
8          System.out.println("The client is running.");
9          rsa = new RSA(); // create a rsa key pair
10         System.out.println("Please enter server port: ");
11         scanner = new Scanner(System.in);
12         portNum = scanner.nextInt();
13         scanner.nextLine();
14
15         try {
16             id = Utils.getID(rsa.getE().toString()+
17             rsa.getN().toString());
18             } catch (NoSuchAlgorithmException e) {
19                 e.printStackTrace();
20             }
21     }
22     public void init() { //start the tcp client
23         String s;
24         while (true) {
25             System.out.println("""
26                 1. Add a value to your sum.
27                 2. Subtract a value from your sum.
28                 3. Get your sum.
29                 4. Exit client""");
30             s = scanner.nextLine();
31             int choice = Integer.parseInt(s);
32
33             int num;
34             int res;
35
36             if (choice == 1) {
37                 System.out.println("Enter the value to add: ");
38                 num = Integer.parseInt(scanner.nextLine());
39                 res = add(num, id);

```



```
39         } else if (choice == 2) {
40             System.out.println("Enter the value to
subtract:");
41             num = Integer.parseInt(scanner.nextLine());
42             res = subtract(num, id);
43         } else if (choice == 3) {
44             System.out.println("Enter your ID: ");
45             res = get(id);
46         } else {
47             return;
48         }
49
50         System.out.println("The result is: " + res);
51         System.out.println();
52     }
53 }
54 public static void main(String[] args) {
55     SigningClientTCP client = new SigningClientTCP();
56     client.init();
57 }
58
59 public int add(int i, String id) {
60     String request = "add,"+i+","+id;
61     return request(request);
62 }
63
64 public int subtract(int i, String id) {
65     String request = "subtract,"+i+","+id;
66     return request(request);
67 }
68
69 public int get(String id) {
70     String request = "get,"+0+","+id;
71     return request(request);
72 }
73
```

```

74     public int request(String requestBody) {
75         BigInteger signature = rsa.sign(requestBody);
76         String request = requestBody
77         +";"+signature+";"+rsa.publicKey();
78         try {
79             socket = new Socket("localhost", portNum);
80             BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
81             PrintWriter out = new PrintWriter(new
BufferedWriter(new
OutputStreamWriter(socket.getOutputStream())));
82             out.println(request);
83             out.println();
84             out.flush();
85             String res = in.readLine();
86             System.out.println(res);
87             return Integer.parseInt(res);
88         } catch (IOException e) {
89             e.printStackTrace();
90         }
91     }
92 }
93
94
95 //source: https://github.com/CMU-Heinz-95702/Project-2-Client-Server
96 public class RSA {
97     private BigInteger n; // n is the modulus for both the
private and public keys
98     private BigInteger e; // e is the exponent of the public key
99     private BigInteger d; // d is the exponent of the private key
100     private MessageDigest md;
101     public BigInteger getN() {
102         return n;
103     }

```

```
104
105     public BigInteger getE() {
106         return e;
107     }
108
109     public BigInteger getD() {
110         return d;
111     }
112
113     public String publicKey() {
114         return e+", "+n;
115     }
116
117     public BigInteger encryptWithPrivate(String message) {
118         //use the private key to encrypt the message, add leading
119         0
120         byte[] messageBytes =
message.getBytes(StandardCharsets.UTF_8);
121         byte[] signBytes = new byte[messageBytes.length+1];
122         signBytes[0] = 0;
123         System.arraycopy(messageBytes, 0, signBytes, 1,
signBytes.length - 1);
124         BigInteger m = new BigInteger(signBytes);
125         return m.modPow(d, n);
126     }
127
128     public BigInteger sign(String message) {
129         //encrypt the message with private key to sign
130         byte[] m = Utils.getHashBytes(message);
131         return encryptWithPrivate(Utils.bytesToHex(m));
132     }
133
134     public String decryptMessage(BigInteger message, BigInteger
e, BigInteger n) {
135         //decrypt the message with key pari e + n
136         byte[] mb = message.modPow(e, n).toByteArray();
```

```

136         return new String(mb);
137     }
138
139     public RSA() {
140         Random rnd = new Random();
141
142         // Generate two large random primes.
143         BigInteger p = new BigInteger(400, 100, rnd);
144         BigInteger q = new BigInteger(400, 100, rnd);
145
146         // Compute n by the equation  $n = p * q$ .
147         n = p.multiply(q);
148
149         // Compute  $\phi(n) = (p-1) * (q-1)$ 
150         BigInteger phi =
151             (p.subtract(BigInteger.ONE)).multiply(q.subtract(BigInteger.ONE))
152             ;
153
154         // Select a small odd integer e that is relatively prime
155         // to phi(n).
156         e = new BigInteger("65537");
157
158         // Compute d as the multiplicative inverse of e modulo
159         // phi(n).
160         d = e.modInverse(phi);
161
162         try { // for hash
163             md = MessageDigest.getInstance("SHA-256");
164         } catch (NoSuchAlgorithmException
165             noSuchAlgorithmException) {
166             noSuchAlgorithmException.printStackTrace();
167         }
168
169         System.out.println(" e = " + e);
170         System.out.println(" d = " + d);
171         System.out.println(" n = " + n);

```

```
167     }
168 }
169
170 //source: https://github.com/CMU-Heinz-95702/Project-2-Client-Server
171 //source: stack overflow
172 public class Utils {
173     private static MessageDigest md;
174
175     static {
176         try {
177             md = MessageDigest.getInstance("SHA-256");
178         } catch (NoSuchAlgorithmException e) {
179             e.printStackTrace();
180         }
181     }
182
183     private static final char[] HEX_ARRAY =
184         "0123456789ABCDEF".toCharArray();
185     public static String bytesToHex(byte[] bytes) { //convert byte
186         to hex string
187         char[] hexChars = new char[bytes.length * 2];
188         for (int j = 0; j < bytes.length; j++) {
189             int v = bytes[j] & 0xFF;
190             hexChars[j * 2] = HEX_ARRAY[v >>> 4];
191             hexChars[j * 2 + 1] = HEX_ARRAY[v & 0x0F];
192         }
193         return new String(hexChars);
194     }
195
196     public static byte[] getHashBytes(String message) { //get the
197         hashed byte of a string
198         return
199         md.digest(message.getBytes(StandardCharsets.UTF_8));
200     }
201 }
```

```

198     public static String getID(String s) throws
        NoSuchAlgorithmException { //get an ID using has
199         byte[] hash = Utils.getHashBytes(s);
200         byte[] idCode = new byte[20];
201         System.arraycopy(hash, 0, idCode, 0, 20);
202         return Utils.bytesToHex(idCode);
203     }
204 }

```

Project2Task5Server

```

1  public class VerifyingServerTCP {
2      public TreeMap<String, Integer> map;
3      Socket socket;
4      ServerSocket listenSocket;
5      RSA rsa;
6
7      public VerifyingServerTCP() {
8          map = new TreeMap<>();
9          socket = null;
10         listenSocket = null;
11         rsa = new RSA();
12
13         try {
14             listenSocket = new ServerSocket(6789);
15         } catch (IOException e) {
16             e.printStackTrace();
17         }
18
19     }
20
21     public void init() { //start the server
22         while (true) {
23             try {
24                 socket = listenSocket.accept();

```

```

25         Scanner in = new
Scanner(socket.getInputStream());
26         StringBuilder request = new StringBuilder();
27         String s;
28         while (in.hasNextLine()) {
29             s = in.nextLine();
30             if (s.equals("")) {
31                 PrintWriter out = new PrintWriter(new
BufferedWriter(new
OutputStreamWriter(socket.getOutputStream())));
32                 String requestBody = request.toString();
33                 requestBody = verify(requestBody); //
verify the message get the real request
34                 if (requestBody.equals("Error in
request")) {
35                     out.println("Error in request");
36                     break;
37                 }
38                 int sum = exec(requestBody);
39
40                 byte[] res =
String.valueOf(sum).getBytes();
41                 out.println(new String(res));
42                 out.flush();
43                 socket.close();
44                 } else {
45                     request.append(s);
46                 }
47             }
48         } catch (IOException e) {
49             e.printStackTrace();
50         }
51     }
52 }
53

```

```

54     public String verify(String request) { //verify the request
and return the real request info
55         //display all the info of client
56         String[] s = request.split(";");
57         String message = s[0];
58         String signature = s[1];
59         String[] pubKey = s[2].split(",");
60         BigInteger e = new BigInteger(pubKey[0]);
61         BigInteger n = new BigInteger(pubKey[1]);
62         System.out.println();
63         System.out.println("message: " + message);
64         System.out.println("signature: " + signature);
65         System.out.println("client's public key: " + e+", "+n);
66
67         String id = message.split(",")[2];
68         String testId = "";
69         try { //use the public key to verify the id
70             testId = Utils.getID(e.toString()+n.toString());
71         } catch (NoSuchAlgorithmException
noSuchAlgorithmException) {
72             noSuchAlgorithmException.printStackTrace();
73         }
74
75         //decrypt the message to get the hashed string
76         String testMessage = rsa.decryptMessage(new
BigInteger(signature), e, n);
77
78         //hashed the message
79         String hashMessage =
Utils.bytesToHex(Utils.getHashBytes(message));
80
81         if (id.equals(testId)) {
82             System.out.println("id: "+ testId + " is valid");
83         } else {
84             return "Error in request";
85         }

```



```

86         if (testMessage.equals(hashMessage)) {
87             System.out.println("Signature is valid");
88         } else {
89             return "Error in request";
90         }
91         return message;
92     }
93
94     public int exec(String requestBody) {
95         String[] params = requestBody.split(",");
96         String method = params[0];
97         int num = Integer.parseInt(params[1]);
98         String id = params[2];
99         if (!map.containsKey(id)) map.put(id, 0);
100        int res;
101        if (method.equals("get")) {
102            res = get(id);
103        } else if (method.equals("add")) {
104            res = add(id, num);
105        } else {
106            res = subtract(id, num);
107        }
108        System.out.printf("Visitor id: %s, method: %s, returned
value: %d\n", id, method, res);
109        return map.get(id);
110    }
111
112    public int add(String id, int num) {
113        map.put(id, map.get(id)+num);
114        return map.get(id);
115    }
116
117    public int subtract(String id, int num) {
118        map.put(id, map.get(id)-num);
119        return map.get(id);
120    }

```

```
121
122     public int get(String id) {
123         return map.get(id);
124     }
125
126     public static void main(String[] args) {
127         VerifyingServerTCP server = new VerifyingServerTCP();
128         server.init();
129     }
130 }
131
```

Project2Task5ClientConsole

```
The client is running.  
e = 65537  
d = 1592175873305199506781984511293115653095376302200547276630103956036227018595  
n = 2099230092518213935179570662414084948939016169097255253153622688288293602867  
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  
1  
Enter the value to add:  
5  
  
5  
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  
2  
Enter the value to subtract:  
3  
  
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  
3  
Enter your ID:  
  
2  
The result is: 2
```

Project2Task5ServerConsole

```
message: add,5,C7FEBE0774D316C5A3A53C3C2537B9B1CD856387
signature: 551483972169504549031797229365034981006034946910875929446542464422814450108917543083
client's public key: 65537,20992300925182139351795706624140849489390161690972552531536226882882
id: C7FEBE0774D316C5A3A53C3C2537B9B1CD856387 is valid
Signature is valid
Visitor id: C7FEBE0774D316C5A3A53C3C2537B9B1CD856387, method: add, returned value: 5

message: subtract,3,C7FEBE0774D316C5A3A53C3C2537B9B1CD856387
signature: 695414892384362915204545171135662614934426736770243143424021427813939997976210951306
client's public key: 65537,20992300925182139351795706624140849489390161690972552531536226882882
id: C7FEBE0774D316C5A3A53C3C2537B9B1CD856387 is valid
Signature is valid
Visitor id: C7FEBE0774D316C5A3A53C3C2537B9B1CD856387, method: subtract, returned value: 2

message: get,0,C7FEBE0774D316C5A3A53C3C2537B9B1CD856387
signature: 488486282630317238859113702275862583662677458562022388116560942307354765098483765917
client's public key: 65537,20992300925182139351795706624140849489390161690972552531536226882882
id: C7FEBE0774D316C5A3A53C3C2537B9B1CD856387 is valid
Signature is valid
Visitor id: C7FEBE0774D316C5A3A53C3C2537B9B1CD856387, method: get, returned value: 2
|
```