ChatRoomTCP:

Client:

```java
package ChatRoomTCP;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.Socket;
import java.util.Scanner;

public class Client {
	private InetAddress host;
	private int port;

	public Client(InetAddress host, int port) {
		this.host = host;
		this.port = port;
	}

	private void execute() throws IOException {
		//Phần bổ sung
		Scanner sc = new Scanner(System.in);
		System.out.print("Nhập vào tên của bạn: ");
		String name = sc.nextLine();

		Socket client = new Socket(host, port);
		ReadClient read = new ReadClient(client);
		read.start();
		WriteClient write = new WriteClient(client, name);
		write.start();
	}


	public static void main(String[] args) throws IOException {
		Client client = new Client(InetAddress.getLocalHost(), 15797);
		client.execute();
	}
}

class ReadClient extends Thread{
	private Socket client;

	public ReadClient(Socket client) {
		this.client = client;
	}

```

```
46          @Override
47          public void run() {
48                  DataInputStream dis = null;
49                  try {
50                          dis = new DataInputStream(client.getInputStream());
51                          while(true) {
52                                  String sms = dis.readUTF();
53                                  System.out.println(sms);
54                          }
55                  } catch (Exception e) {
56                          try {
57                                  dis.close();
58                                  client.close();
59                          } catch (IOException ex) {
60                                  System.out.println("Ngắt kết nối Server");
61                          }
62                  }
63          }
64  }
65
66  class WriteClient extends Thread{
67          private Socket client;
68          private String name;
69
70          public WriteClient(Socket client, String name) {
71                  this.client = client;
72                  this.name = name;
73          }
74
75          @Override
76          public void run() {
77                  DataOutputStream dos = null;
78                  Scanner sc = null;
79                  try {
80                          dos = new DataOutputStream(client.getOutputStream());
81                          sc = new Scanner(System.in);
82                          while(true) {
83                                  String sms = sc.nextLine();
84                                  dos.writeUTF(name + ": " + sms);
85                          }
86                  } catch (Exception e) {
87                          try {
88                                  dos.close();
89                                  client.close();
90                          } catch (IOException ex) {
91                                  System.out.println("Ngắt kết nối Server");
92                          }
93                  }
94          }
95
96  }
```

Server:

```java
package ChatRoomTCP;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.Scanner;

public class Server {
	private int port;
	public static ArrayList<Socket> listSK;

	public Server(int port) {
		this.port = port;
	}

	private void execute() throws IOException {
		ServerSocket server = new ServerSocket(port);
		WriteServer write = new WriteServer();
		write.start();
		System.out.println("Server is listening...");
		while (true) {
			Socket socket = server.accept();
			System.out.println("Đã kết nối với " + socket);
			Server.listSK.add(socket);
			ReadServer read = new ReadServer(socket);
			read.start();
		}
	}

	public static void main(String[] args) throws IOException {
		Server.listSK = new ArrayList<>();
		Server server = new Server(15797);
		server.execute();
	}

}

class ReadServer extends Thread {
	private Socket socket;

	public ReadServer(Socket socket) {
		this.socket = socket;
	}

	@Override
```

```
48          @Override
49          public void run() {
50                  try {
51                          DataInputStream dis = new DataInputStream(socket.getInputStream());
52                          while (true) {
53                                  String sms = dis.readUTF();
54                                  if(sms.contains("exit")) {
55                                          Server.listSK.remove(socket);
56                                          System.out.println("Đã ngắt kết nối với " + socket);
57                                          dis.close();
58                                          socket.close();
59                                          continue; //Ngắt kết nối rồi
60                                  }
61                                  for (Socket item : Server.listSK) {
62                                          if(item.getPort() != socket.getPort()) {
63                                                  DataOutputStream dos = new DataOutputStream(item.getOutputStream());
64                                                  dos.writeUTF(sms);
65                                          }
66                                  }
67                                  System.out.println(sms);
68                          }
69                  } catch (Exception e) {
70                          try {
71                                  socket.close();
72                          } catch (IOException ex) {
73                                  System.out.println("Ngắt kết nối Server");
74                          }
75                  }
76          }
77  }
78
79  class WriteServer extends Thread {
80
81          @Override
82          public void run() {
83                  DataOutputStream dos = null;
84                  Scanner sc = new Scanner(System.in);
85                  while (true) {
86                          String sms = sc.nextLine();     //Đang đợi Server nhập dữ liệu
87                          try {
88                                  for (Socket item : Server.listSK) {
89                                          dos = new DataOutputStream(item.getOutputStream());
90                                          dos.writeUTF("Server: " + sms);
91                                  }
92                          } catch (IOException e) {
93                                  e.printStackTrace();
94                          }
95
96                  }
97          }
98
99  }
```

Giải PT bậc 1:

Client:

```
package GiaiPTBac1;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.Socket;
import java.util.Scanner;

public class Client {
	private InetAddress host;
	private int port;
	private Scanner sc = new Scanner(System.in);

	public Client(InetAddress host, int port) {
		this.host = host;
		this.port = port;
	}

	private void execute() throws IOException {
		//Cấu hình Client
		Socket client = new Socket(host, port);
		//Nhập dữ liệu
		int a = input("Nhập vào a: ");
		int b = input("Nhập vào b: ");

		DataInputStream dis = new DataInputStream(client.getInputStream());
		DataOutputStream dos = new DataOutputStream(client.getOutputStream());

		//Gửi dữ liệu đến Server
		dos.writeInt(a);
		dos.writeInt(b);

		//Nhận dữ liệu từ Server
		System.out.println(dis.readUTF());

		dis.close();
		dos.close();
		client.close();
	}

	public static void main(String[] args) throws IOException {
		Client client = new Client(InetAddress.getLocalHost(), 15797);
		client.execute();
	}

	private int input(String request) {
		int number = 0;
		boolean flag = true;
		do {
			try {
				System.out.print(request);
				number = Integer.parseInt(sc.nextLine());
				flag = false;
			} catch (Exception e) {
				System.out.println("Dữ liệu không đúng định dạng. Vui lòng nhập lại: ");
			}
		} while (flag);
		return number;
	}
}
```

Server:

```java
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

public class Server {
	private int port;

	public Server(int port) {
		this.port = port;
	}

	private void execute() throws IOException {
		ServerSocket server = new ServerSocket(port);
		System.out.println("Server is listening...");

		while(true) {
			Socket socket = server.accept();
			PTBac1 ptBac1 = new PTBac1(socket);
			ptBac1.start();
		}
	}

	public static void main(String[] args) throws IOException {
		Server server = new Server(15797);
		server.execute();
	}
}

class PTBac1 extends Thread{
	private Socket socket;

	public PTBac1(Socket socket) {
		this.socket = socket;
	}

	@Override
	public void run() {
		try {
			DataInputStream dis = new DataInputStream(socket.getInputStream());
			DataOutputStream dos = new DataOutputStream(socket.getOutputStream());

			int a = dis.readInt();
			int b = dis.readInt();
			if(a == 0) {
				if(b == 0) {
					dos.writeUTF("Phương trình vô số nghiệm");
				}else {
					dos.writeUTF("Phương trình vô nghiệm");
				}
			}else {
				double result = (double)-b/a;
				dos.writeUTF("Phương trình có nghiệm = " + result);
			}
			dis.close();
			dos.close();
		} catch (Exception e) {
			e.printStackTrace();
		}
	}
```

Giải Pt bậc 2- - UDP

Client:

```java
package GiaiPTBac2_UDP;

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


public class Client {
	private InetAddress host;
	private int port;
	private Scanner sc = new Scanner(System.in);

	public Client(InetAddress host, int port) {
		this.host = host;
		this.port = port;
	}

	private void execute() throws IOException {
		//Cấu hình Client
		DatagramSocket client = new DatagramSocket();

		//Nhập dữ liệu
		int a = input("Nhập vào a: ");
		int b = input("Nhập vào b: ");
		int c = input("Nhập vào c: ");

		//Đóng gói thành Packet và gửi đến Server
		DatagramPacket a_DP = createPacket(a, 1);
		DatagramPacket b_DP = createPacket(b, 2);
		DatagramPacket c_DP = createPacket(c, 3);

		client.send(a_DP);
		client.send(b_DP);
		client.send(c_DP);

		//Nhận dữ liệu từ Server
		String [] result = receiveData(client).split("_");
		double value = Double.parseDouble(result[0]);
		int key          = Integer.parseInt(result[1]);
		if(key == -1) {
			System.out.println("Phương trình vô nghiệm");
		}else if(key == -2) {
			System.out.println("Phương trình vô số nghiệm");
		}else if(key == -3) {
			System.out.println("Phương trình có 1 nghiệm = " + value);
		}else if(key == -4) {
			String [] result1 = receiveData(client).split("_");
			double value_1 = Double.parseDouble(result1[0]);
			System.out.println("Phương trình có 2 nghiệm phân biệt: ");
			System.out.println("x1 = " + value);
			System.out.println("x2 = " + value_1);
		}
		client.close();
	}

```

```
58          public static void main(String[] args) throws IOException {
59                  Client client = new Client(InetAddress.getLocalHost(), 15797);
60                  client.execute();
61          }
62
63          private String receiveData(DatagramSocket server) throws IOException {
64                  byte[] temp = new byte[1024];
65                  DatagramPacket recieve_Packet = new DatagramPacket(temp, temp.length);
66                  server.receive(recieve_Packet);
67                  return new String(recieve_Packet.getData()).trim();
68          }
69
70          private DatagramPacket createPacket(int value, int index) {
71                  String str = String.valueOf(value) + "_" + index;
72                  byte[] arrData = str.getBytes();
73                  return new DatagramPacket(arrData, arrData.length, host, port);
74          }
75
76          private int input(String request) {
77                  int number = 0;
78                  boolean flag = true;
79                  do {
80                          try {
81                                  System.out.print(request);
82                                  number = Integer.parseInt(sc.nextLine());
83                                  flag = false;
84                          } catch (Exception e) {
85                                  System.out.println("Dữ liệu không đúng định dạng. Vui lòng nhập lại: ");
86                          }
87                  } while (flag);
88                  return number;
89          }
90
91
92  }
```

Server:

```
1   package GiaiPTBac2_UDP;
2
3   import java.io.IOException;
4   import java.net.DatagramPacket;
5   import java.net.DatagramSocket;
6   import java.net.InetAddress;
7   import java.net.SocketException;
8   import java.util.ArrayList;
9
10  public class Server {
11          private int port;
12          private InetAddress clientIP;
13          private int clientPort;
14          public static ArrayList<Data> listSK;
15
16          public Server(int port) {
17                  this.port = port;
18          }
19
```

```
20          private void execute() throws IOException {
21                  // Tạo Server Socket
22                  DatagramSocket server = new DatagramSocket(port);
23                  System.out.println("Server is listening....");
24                  while (true) {
25                          String[] arrData = recieveData(server).split("_");
26                          int key = Integer.parseInt(arrData[1]);
27                          int value = Integer.parseInt(arrData[0]);
28                          if (checkClientExist(clientIP, clientPort) == false) { // Client này kết nối lần đầu
29                                  Data data = new Data(clientIP, clientPort);
30                                  // Gán dữ liệu vào
31                                  if (key == 1) {
32                                          data.setA(value);
33                                  } else if (key == 2) {
34                                          data.setB(value);
35                                  } else if (key == 3) {
36                                          data.setC(value);
37                                  }
38                                  listSK.add(data);
39                          } else { // Đã tồn tại
40                                  for (Data item : listSK) {
41                                          if (item.getHost().equals(clientIP) && item.getPort() == clientPort) {
42                                                  // Gán dữ liệu vào
43                                                  if (key == 1) {
44                                                          item.setA(value);
45                                                  } else if (key == 2) {
46                                                          item.setB(value);
47                                                  } else if (key == 3) {
48                                                          item.setC(value);
49                                                  }
50                                          }
51
52                                          // Kiểm tra xem Client này đã đưa đủ dữ liệu thì giải PT
53                                          if(item.getCount() == 3) {
54                                                  giaiPTBac2(server, item);
55                                                  //Sau khi tính toán cũng xóa Client trong danh sách ra
56                                                  listSK.remove(item);
57                                                  break;
58                                          }
59                                  }
60                          }
61                  }
62          }
63
64          public static void main(String[] args) throws IOException {
65                  Server.listSK = new ArrayList<>();
66                  Server server = new Server(15797);
67                  server.execute();
68          }
69
70          private void giaiPTBac2(DatagramSocket server, Data data) throws IOException {
71                  /*
72                   * -1: vô nghiệm        -2: vô số nghiệm        -3: 1 nghiệm kép        -4: 2 nghiệm
73                   * */
74                  int a = data.getA();
75                  int b = data.getB();
76                  int c = data.getC();
77
78                  //Xử lý và gửi về Client
79                  if(a == 0) {
```

```
78                  //Xử lý và gửi về Client
79                  if(a == 0) {
80                          if(b == 0) {
81                                  if(c == 0) {    //Phương trình vô số nghiệm
82                                          sendData(0, -2, server, clientIP, clientPort);
83                                  }else {                 //Phương trình vô nghiệm
84                                          sendData(0, -1, server, clientIP, clientPort);
85                                  }
86                          }else { //Phương trình có 1 nghiệm
87                                  sendData((-c/b), -3, server, clientIP, clientPort);
88                          }
89                          return;
90                  }
91
92                  double delta = (b*b) - (4*a*c);
93                  double x1;
94                  double x2;
95                  if(delta > 0) {                 //Phương trình có 2 nghiệm
96                          x1 = (double) ((-b + Math.sqrt(delta)) / (2 * a));
97                          x2 = (double) ((-b - Math.sqrt(delta)) / (2 * a));
98                          sendData(x1, -4, server, clientIP, clientPort);
99                          sendData(x2, -4, server, clientIP, clientPort);
100                 }else if(delta == 0) {  //Phương trình có 1 nghiệm
101                         x1 = (-b / (2 * a));
102                         sendData(x1, -3, server, clientIP, clientPort);
103                 }else {                                 //Phương trình vô nghiệm
104                         sendData(0, -1, server, clientIP, clientPort);
105                 }
106
107         }
108
109         private void sendData(double value, int index, DatagramSocket server, InetAddress clientIP, int clientPort) throws IOException {
110                 byte[] temp = new byte[1024];
111                 String str = String.valueOf(value) + "_" + index;
112                 temp = str.getBytes();
113                 DatagramPacket send_result_Packet = new DatagramPacket(temp, temp.length, clientIP, clientPort);
114                 server.send(send_result_Packet);
115         }
116
117         private String recieveData(DatagramSocket server) throws IOException {
118                 byte[] temp = new byte[1024];
119                 DatagramPacket recieve_Packet = new DatagramPacket(temp, temp.length);
120                 server.receive(recieve_Packet);
121                 clientIP = recieve_Packet.getAddress();
122                 clientPort = recieve_Packet.getPort();
123                 return new String(recieve_Packet.getData()).trim();
124         }
125
126         private boolean checkClientExist(InetAddress clientIP, int clientPort) {
127                 for (Data item : listSK) {
128                         if (item.getHost().equals(clientIP) && item.getPort() == clientPort) {
129                                 return true;
130                         }
131                 }
132                 return false;
133         }
134
135 }
136
137 class Data {

public InetAddress getHost() {
        return host;
}

public void setHost(InetAddress host) {
        this.host = host;
}

public void setPort(int port) {
        this.port = port;
}
```

```
137 class Data {
138         private InetAddress host;
139         private int port;
140         private int a;
141         private int b;
142         private int c;
143         private int count;
144
145         public Data(InetAddress host, int port) {
146                 this.host = host;
147                 this.port = port;
148                 this.a = 0;
149                 this.b = 0;
150                 this.c = 0;
151                 this.count = 0;
152         }
153
```

Login

Client:

```java
package LoginTCP;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.Scanner;

public class Client {
	private InetAddress host;
	private int port;
	public Client(InetAddress host, int port) {
		this.host = host;
		this.port = port;
	}

	private void execute() throws IOException {
		Socket client = new Socket(host, port);
		DataInputStream dis = new DataInputStream(client.getInputStream());
		DataOutputStream dos = new DataOutputStream(client.getOutputStream());

		Scanner sc = new Scanner(System.in);

		int id = 0;
		String pass = "";
		boolean flag = true;
		int result = 0;
		do {
			System.out.print("Nhập vào id: ");
			try {
				id = Integer.parseInt(sc.nextLine());
			} catch (NumberFormatException e) {
				System.out.println("id phải là số nguyên. Vui lòng nhập lại!");
				System.out.print("Nhập vào id: ");
				id = Integer.parseInt(sc.nextLine());
			}

			System.out.print("Nhập vào password: ");
			pass = sc.nextLine();
			dos.writeInt(id);
			dos.writeUTF(pass);
			result = dis.readInt();
			if(result == -1) {
				System.out.println("Sai password. Vui lòng nhập lại");
			}else if(result == -2) {
				System.out.println("User không tồn tại. Vui lòng nhập lại");
			}else {
				System.out.println("Số tiền của bạn là: " + result);
				flag = false;
			}

		} while (flag);
		dis.close();
		dos.close();
		client.close();
	}

	public static void main(String[] args) throws IOException {
		Client client = new Client(InetAddress.getLocalHost(), 15797);
		client.execute();
	}
}
```

Server:

```java
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Server {
	private int port;

	public Server(int port) {
		this.port = port;
	}

	private void execute() throws IOException, SQLException {
		ServerSocket server = new ServerSocket(port);
		System.out.println("Server is listening....");
		Socket socket = server.accept();
		DataInputStream dis = new DataInputStream(socket.getInputStream());
		DataOutputStream dos = new DataOutputStream(socket.getOutputStream());

		int id = 0;
		String pass = "";
		boolean flag = true;
		int result = 0;
		do {
			id = dis.readInt();
			pass = dis.readUTF();
			Connection connect = DBConnection.getConnection();
			String query = "SELECT dbo.FN_Login(?, ?) AS result";
			PreparedStatement ps = connect.prepareStatement(query);
			ps.setInt(1, id);
			ps.setString(2, pass);
			ResultSet rs = ps.executeQuery();
			rs.next();
			result = rs.getInt("result");
			if(result == -1) {
				dos.writeInt(-1);
			}else if(result == -2) {
				dos.writeInt(-2);
			}else {
				dos.writeInt(result);
				flag = false;
			}
			DBConnection.closeConnection(connect);
		} while (flag);
		dis.close();
		dos.close();
		socket.close();
		server.close();
	}
	public static void main(String[] args) throws IOException, SQLException {
		Server server = new Server(15797);
		server.execute();

	}

}
```

Database connection:

```java
package LoginTCP;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DBConnection {
        public static Connection getConnection(){
        Connection connection = null;
       try {
           Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
           String url = "jdbc:sqlserver://localhost:1433; databaseName=NhanVien";
           String user = "sa";
           String pass = "123456";
           connection = DriverManager.getConnection(url, user, pass);
       } catch (Exception ex) {
           ex.printStackTrace();
       }
       return connection;
    }

    public static void closeConnection(Connection con){
        if(con != null){
            try {
                con.close();
            } catch (SQLException ex) {
                ex.printStackTrace();
            }
        }
    }
}
```

List các câu về số học:

Client:

```java
package menuTCP;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.Socket;
import java.util.Scanner;

public class Client {
        private InetAddress host;
        private int port;
        private int a;
        private int b;
        private int c;
        private int d;
        private Scanner sc = new Scanner(System.in);

        public Client(InetAddress host, int port) {
                this.host = host;
                this.port = port;
        }

```

```
24          private void execute() throws IOException {
25                  Socket client = new Socket(host, port);
26                  DataInputStream dis = new DataInputStream(client.getInputStream());
27                  DataOutputStream dos = new DataOutputStream(client.getOutputStream());
28
29                  // Nhập dữ liệu
30                  a = input("Nhập vào số a: ");
31                  b = input("Nhập vào số b: ");
32                  c = input("Nhập vào số c: ");
33                  d = input("Nhập vào số d: ");
34                  dos.writeInt(a);
35                  dos.writeInt(b);
36                  dos.writeInt(c);
37                  dos.writeInt(d);
38
39                  while (true) {
40                          // Nhận menu và in ra màn hình
41                          System.out.println(dis.readUTF());
42
43                          int choose = input("Nhập vào tùy chọn của bạn: ");
44                          //Gửi tùy chọn đến Server
45                          dos.writeInt(choose);
46                          switch (choose) {
47                          case 1:
48                                  System.out.println("Ước chung lớn nhất = " + dis.readInt());
49                                  break;
50                          case 2:
51                                  System.out.println("Bội chung nhỏ nhất = " + dis.readInt());
52                                  break;
53                          case 3:
54                                  System.out.print("Danh sách sau khi sắp xếp tăng dần: ");
55                                  System.out.print(dis.readInt() + " ");
56                                  System.out.print(dis.readInt() + " ");
57                                  System.out.print(dis.readInt() + " ");
58                                  System.out.print(dis.readInt() + " ");
59                                  break;
60                          case 4:
61                                  System.out.print("Danh sách sau khi sắp xếp giảm dần: ");
62                                  System.out.print(dis.readInt() + " ");
63                                  System.out.print(dis.readInt() + " ");
64                                  System.out.print(dis.readInt() + " ");
65                                  System.out.print(dis.readInt() + " ");
66                                  break;
67                          case 5:
68                                  System.out.println("Tổng các số lẻ = " + dis.readInt());
69                                  break;
70                          case 6:
71                                  System.out.println("Bội chung số chẵn = " + dis.readInt());
72                                  break;
73                          case 7:
74                                  System.out.println("Các số nguyên tố: ");
75                                  int songuyento = dis.readInt();
76                                  while(songuyento != -1) {
77                                          System.out.print(songuyento + " ");
78                                          songuyento = dis.readInt();
79                                  }
80                                  break;
```

```
82                          case 8:
83                                  System.out.println("Các số chính phương: ");
84                                  int sochinhphuong = dis.readInt();
85                                  while(sochinhphuong != -1) {
86                                          System.out.print(sochinhphuong + " ");
87                                          sochinhphuong = dis.readInt();
88                                  }
89                                  break;
90
91                          case 9:
92                                  System.out.println("Các số hoàn hảo: ");
93                                  int sohoanhao = dis.readInt();
94                                  while(sohoanhao != -1) {
95                                          System.out.print(sohoanhao + " ");
96                                          sohoanhao = dis.readInt();
97                                  }
98                                  break;
99                          case 10:
100                                 break;
101
102                         default:
103                                 System.out.println("Vui lòng chỉ nhập số trong menu!");
104                                 break;
105                         }
106                         if(choose == 10) break; // Bằng 10 thoát
107                 }
108                 dis.close();
109                 dos.close();
110                 client.close();
111         }
112
113         public static void main(String[] args) throws IOException {
114                 Client client = new Client(InetAddress.getLocalHost(), 15797);
115                 client.execute();
116         }
117
118         private int input(String request) {
119                 int number = 0;
120                 boolean flag = true;
121                 do {
122                         try {
123                                 System.out.println(request);
124                                 number = Integer.parseInt(sc.nextLine());
125                                 flag = false;
126                         } catch (Exception e) {
127                                 System.out.println("Dữ liệu không đúng định dạng. Vui lòng nhập lại!");
128                         }
129                 } while (flag);
130                 return number;
131         }
132 }
```

Server:

```
3  import java.io.DataInputStream;
4  import java.io.DataOutputStream;
5  import java.io.IOException;
6  import java.net.ServerSocket;
7  import java.net.Socket;
8  import java.sql.SQLException;
9  import java.util.ArrayList;
10 import java.util.Collections;
11
12 public class Server {
13         private int port;
14         private int a;
15         private int b;
16         private int c;
17         private int d;
18
19         public Server(int port) {
20                 this.port = port;
21         }
22
23         private void execute() throws IOException{
24                 System.out.println("Server is listening...");
25                 //Cấu hình Server
26                 ServerSocket server = new ServerSocket(port);
27                 Socket socket = server.accept();
28                 DataInputStream dis = new DataInputStream(socket.getInputStream());
29                 DataOutputStream dos = new DataOutputStream(socket.getOutputStream());
30
31                 //Nhận dữ liệu từ Client
32                 a = dis.readInt();
33                 b = dis.readInt();
34                 c = dis.readInt();
35                 d = dis.readInt();
36                 int [] arr = {a,b,c,d};
37
38                 while(true) {
39                         //Gửi menu về Client
40                         dos.writeUTF(menu());
41                         //Nhận tùy chọn menu
42                         int choose = dis.readInt();
43                         switch (choose) {
44                         case 1:
45                                 int UCLN = GCD(arr[0], GCD(arr[1], GCD(arr[2], arr[3])));
46                                 dos.writeInt(UCLN);
47                                 break;
48                         case 2:
49                                 dos.writeInt(LCM(arr));
50                                 break;
51                         case 3:
52                                 ArrayList<Integer> arrASC = ASC(arr);   //Mảng sau khi sắp xếp
53                                 dos.writeInt(arrASC.get(0));
54                                 dos.writeInt(arrASC.get(1));
55                                 dos.writeInt(arrASC.get(2));
56                                 dos.writeInt(arrASC.get(3));
57                                 break;
```

```
58                      case 4:
59                              ArrayList<Integer> arrDESC = DESC(arr); //Mảng sau khi sắp xếp
60                              dos.writeInt(arrDESC.get(0));
61                              dos.writeInt(arrDESC.get(1));
62                              dos.writeInt(arrDESC.get(2));
63                              dos.writeInt(arrDESC.get(3));
64                              break;
65                      case 5:
66                              dos.writeInt(sumOfOdd(arr));
67                              break;
68                      case 6:
69                              dos.writeInt(sumOfEven(arr));
70                              break;
71                      case 7:
72                              for (Integer item : arr) {
73                                      if(soNguyenTo(item) == true) {
74                                              dos.writeInt(item);
75                                      }
76                              }
77                              dos.writeInt(-1);       //Đánh dấu kết thúc
78                              break;
79
80                      case 8:
81                              for (Integer item : arr) {
82                                      if(soChinhPhuong(item) == true) {
83                                              dos.writeInt(item);
84                                      }
85                              }
86                              dos.writeInt(-1);       //Đánh dấu kết thúc
87                              break;
88
89                      case 9:
90                              for (Integer item : arr) {
91                                      if(soHoanHao(item) == true) {
92                                              dos.writeInt(item);
93                                      }
94                              }
95                              dos.writeInt(-1);       //Đánh dấu kết thúc
96                              break;
97                      case 10:
98                              break;
99
100                     default:
101                             break;
102                     }
103                     if(choose == 10) break;
104             }
105             dis.close();
106             dos.close();
107             server.close();
108     }
109     private String menu() {
110             String menu = "\n\n============MENU============\n"
111                                     + "1. Tìm ước chung lớn nhất\n"
112                                     + "2. Tìm bội chung nhỏ nhất\n"
113                                     + "3. Sắp xếp tăng dần\n"
114                                     + "4. Sắp xếp giảm dần\n"
115                                     + "5. Tổng tất cả các số lẻ\n"
116                                     + "6. Tổng tất cả các số chẵn\n"
117                                     + "7. Các số nguyên tố\n"
118                                     + "8. Các số chính phương\n"
119                                     + "9. Các số hoàn hảo\n"
120                                     + "10. Thoát";
121             return menu;
122     }
123
```

```
124         private int GCD(int a, int b) {
125                 while (a != b) {
126             if (a > b)
127                   a = a - b;
128             else
129                   b = b - a;
130              }
131              return a;
132         }
133
134         private int LCM(int [] arr)
135         {
136                int max = 0, k = 1;
137         for (int i = 0; i < arr.length; i++) {  //Tìm số lớn nhất trong mảng
138               if (max < arr[i]) {
139                     max = arr[i];
140               }
141         }
142         int temp = max;
143         for (int i = 0; i < arr.length; i++) {
144               while (max % arr[i] != 0) {
145                     k++;
146                     max = temp * k;
147               }
148         }
149         return max;
150         }
151
152         private ArrayList<Integer> ASC (int [] arr) {
153                 ArrayList<Integer> list = new ArrayList<>();
154                 for (int i : arr) {
155                         list.add(i);
156                 }
157                 Collections.sort(list);
158                 return list;
159         }
160
161         private ArrayList<Integer> DESC (int [] arr) {
162                 ArrayList<Integer> list = new ArrayList<>();
163                 for (int i : arr) {
164                         list.add(i);
165                 }
166                 Collections.sort(list);
167                 Collections.reverse(list);
168                 return list;
169         }
170
171         private int sumOfOdd(int [] arr) {
172                 int sum = 0;
173                 for (int i : arr) {
174                         if(i % 2 == 1) {
175                                 sum += i;
176                         }
177                 }
178                 return sum;
179         }
180
```

```
181         private int sumOfEven(int [] arr) {
182                 int sum = 0;
183                 for (int i : arr) {
184                         if(i % 2 == 0) {
185                                 sum += i;
186                         }
187                 }
188                 return sum;
189         }
190
191         private boolean soNguyenTo(int n) {
192                 if(n < 2) return false;
193                 for (int i = 2; i <= Math.sqrt(n); i++) {
194                         if((n%i == 0) && (n != i)) return false;
195                 }
196                 return true;
197         }
198
199         private boolean soChinhPhuong(int n) {
200                 int temp = (int)Math.sqrt(n);
201                 return (temp*temp == n);
202         }
203
204         private boolean soHoanHao(int n) {
205                 int tong = 0;
206         for (int i = 1; i < n ; i ++){
207             if ( n % i == 0 ) tong = tong + i;
208         }
209         return (tong == n);
210         }
211
212         public static void main(String[] args) throws IOException {
213                 Server server = new Server(15797);
214                 server.execute();
215         }
216
217 }
```

Từ điển:

Client :

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class TCPClient {

    private static Socket socket;
    private static BufferedReader in;
    private static BufferedWriter out;
    private static BufferedReader stdIn;
    private static boolean loop = true;

    public TCPClient() {
    }

    //Hàm xử lý
    public static void handle() {
        try {
            socket = new Socket("localhost", 2001);
            System.out.println("Khach hang da ket noi...");

            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));

            stdIn = new BufferedReader(new InputStreamReader(System.in));

            while (loop) {
                System.out.println("Vui long nhap tu tra cuu: ");
                String input = stdIn.readLine();
                out.write(input);
                out.newLine();
                out.flush();

                String message = in.readLine();
                if (message.equals("Ngat ket noi....")) {
                    loop = false;
                }
                System.out.println(message);
            }
            in.close();
            stdIn.close();
            out.close();
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        TCPClient.handle();
    }
}
```

Server:

```
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;
import java.nio.ByteBuffer;
import java.nio.charset.StandardCharsets;
import java.nio.file.Paths;
import java.util.*;

public class TCPServer {

    private static ServerSocket serverSocket;
    private static Socket socket;
    private static BufferedReader in;
    private static BufferedWriter out;
    private static LinkedHashMap<String, String> dictionaries = new LinkedHashMap<>();
    private static boolean loop = true;
    private static final String URL_FILE = "./src/bai1/dictionary.txt";


    private static void readFile(String filePath) {
        try {
            Scanner readFile = new Scanner(new File(filePath));
            while (readFile.hasNextLine()) {
                String[] line = readFile.nextLine().split(";");
                String key = line[0];
                String value = line[1];
                dictionaries.putIfAbsent(key, value);
            }
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        TCPServer.readFile(URL_FILE);
        try {
            serverSocket = new ServerSocket(2001);
            System.out.println("May chu dang chay tren cong 2001...");
            socket = serverSocket.accept();
            System.out.println("Khach hang da ket noi!");

            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));

            while (loop) {
                String[] line = in.readLine().split(";");
                String note = line[0];
                System.out.println("Khach hang muon tra cuu: " + note);

                switch (note.toUpperCase()) {
                    case "BYE":
                        loop = false;
                        out.write("Ngat ket noi....");
                        out.newLine();
                        out.flush();
                        break;
                    case "ADD":
                        String x = line[1];
                        String y = line[2];
                        boolean flagAdd = false;
                        for (String key : dictionaries.keySet()) {
                            if (key.toLowerCase().equals(x.toLowerCase())) {
                                out.write("Da ton tai trong tu dien");
                                out.newLine();
                                out.flush();
                                flagAdd = true;
                                break;
                            }
```

```
                    }
                    if (!flagAdd) {
                        dictionaries.putIfAbsent(x, y);
                        PrintWriter printWriter = new PrintWriter(new FileWriter(URL_FILE), true);
                        for (String key : dictionaries.keySet()) {
                            printWriter.write(key + ";" + dictionaries.get(key) + "\n");
                        }
                        printWriter.flush();
                        readFile(URL_FILE);
                        out.write("Them vao tu dien thanh cong");
                        out.newLine();
                        out.flush();
                    }
                    break;
                case "DEL":
                    if (dictionaries.containsKey(line[1])) {
                        dictionaries.remove(line[1]);
                        PrintWriter printWriter = new PrintWriter(new FileWriter(URL_FILE), true);
                        for (String key : dictionaries.keySet()) {
                            printWriter.write(key + ";" + dictionaries.get(key) + "\n");
                        }
                        printWriter.flush();
                        readFile(URL_FILE);
                        out.write("Xoa thanh cong");
                        out.newLine();
                        out.flush();
                    } else {
                        out.write("Khong ton tai tu do trong tu dien");
                        out.newLine();
                        out.flush();
                    }

                    break;

                default:
                    //Tieng anh sang tieng viet
                    boolean flag = false;
                    for (String key : dictionaries.keySet()) {
                        if (note.equalsIgnoreCase(key)) {
                            String value = dictionaries.get(key);
                            out.write("dich ra tieng anh:" + value);
                            out.newLine();
                            out.flush();
                            flag = true;
                            break;
                        }
                    }
                    //Tieng viet sang tieng anh
                    if (!flag) {
                        for (String key : dictionaries.keySet()) {
                            String value = dictionaries.get(key);
                            if (note.equalsIgnoreCase(value)) {
                                out.write("dich ra tieng anh:" + key);
                                out.newLine();
                                out.flush();
                                flag = true;
                                break;
                            }
                        }
                    }
                    if (!flag) {
                        out.write("Khong tim thay tu can tra cuu! Vui long nhap lai.");
                        out.newLine();
                        out.flush();
                        break;
                    }
            }
        }
        in.close();
        out.close();
        socket.close();
        serverSocket.close();
    } catch (IOException e) {
        System.out.println(e.getMessage());
    }
  }
}
```

Bài 1:

Viết chương trình gửi tin nhắn hai chiều giữa client-server sử dụng TCP socket.

Client gửi 1 chuỗi ký tự bất kỳ đến server

Server nhận và gửi chuỗi đảo ngược về client

Client xuất kết quả ra console, chương trình kết thúc khi client gửi chuỗi bye.

Client:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class client {
    Socket socket =null;
    BufferedWriter out = null;
    BufferedReader in = null;
    BufferedReader stdIn = null;

    public client(String host, int port){
        try{
            socket = new Socket(host, port);
            System.out.println("Client connected");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line="";
            while(!line.equals("bye")){
                try{
                    //nhập chuỗi
                    line = stdIn.readLine();
                    System.out.println("Client sent:" + line);
                    out.write(line);
                    out.newLine();
                    out.flush();// đẩy dữu liệu qua server

                    //nhận kết qua đảo chuỗi từ server
                    String da= in.readLine();
                    System.out.println("Result:" +da);
                }catch(Exception e){
                    System.err.println(e);
                }
            }
            in.close();
            stdIn.close();
            out.close();
            socket.close();
        }catch(Exception e){
            System.out.println(e);
        }
    }

    public static void main(String[] args){
            client client = new client("localhost",6000);
    }
}
```

Server:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.ServerSocket;
import java.net.Socket;

public class server {
    ServerSocket server = null;
    Socket socket =null;
    BufferedWriter out =null;
    BufferedReader in =null;
    StringBuffer stringBuffer= null;
    public server(int port){
        try{
            server =new ServerSocket(port);
            System.out.println("Server started. Waitting for client....");
            socket =server.accept();
            System.out.println("Client" + socket.getInetAddress()+ "is connect");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            String line ="";
            while(!line.equals("bye")){
                try{
                    //nhận chuỗi từ Client
                    line = in.readLine();
                    System.out.println("Server received: " + line);
                    //xử lý đảo chuỗi
                    stringBuffer = new StringBuffer(line);
                    String result = stringBuffer.reverse().toString();
                    out.write(result);
                    out.newLine();
                    out.flush();// đẩy dữu liệu qua
                }catch(Exception e){
                    System.out.println(e);
                }
            }
            in.close();
            out.close();
            socket.close();
            server.close();
        } catch (Exception e){
            System.err.println(e);
        }
    }
    public static void main(String[] args){
        server server = new server(6000);
    }
}
```

Bài 2:

Viết chương trình tìm số hoàn hảo, hoạt động theo mô hình client-server, sử dụng TCP socket

Client gửi 1 số n nguyên dương đến server.

Server kiểm tra n, nếu:

o Là số hoàn hảo: trả kết quả về client và xuất ra màn hình

o Không phải số hoàn hảo: trả về client số hoàn hảo lớn hơn và gần n nhất.

Client:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class client {
    Socket socket =null;
    BufferedWriter out = null;
    BufferedReader in = null;
    BufferedReader stdIn = null;

    public client(String host, int port){
        try{
            socket = new Socket(host, port);
            System.out.println("Client connected");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line="";
            while(!line.equals("bye")){
                try{
                    //nhập số
                    line = stdIn.readLine();
                    System.out.println("Client sent:" + line);
                    out.write(line);
                    out.newLine();
                    out.flush();// đẩy dữu liệu qua server

                    //nhận kết qua đảo chuỗi từ server
                    String da= in.readLine();
                    System.out.println("Result:" +da);
                }catch(Exception e){
                    System.err.println(e);
                }
            }
            in.close();
            stdIn.close();
            out.close();
            socket.close();
        }catch(Exception e){
            System.out.println(e);
        }
    }

    public static void main(String[] args){
            client client = new client("localhost",6000);
    }
}
```

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.ServerSocket;
import java.net.Socket;

public class server {
    ServerSocket server = null;
    Socket socket =null;
    BufferedWriter out =null;
    BufferedReader in =null;
    StringBuffer stringBuffer= null;
    public int ktSoHoanHao(int n){
        int s=0;
        for(int i=1;i<n;i++){
            if(n%i==0)
                s+=i;
        }
        if(s==n)
            return 1;
        return 0;
    }
    public int soHHKeSau(int n){
        for(int i=n+1;i<Integer.MAX_VALUE;i++){
            if(ktSoHoanHao(i)==1)
                return i;
        }
        return 0;
    }
    public server(int port){
        try{
            server =new ServerSocket(port);
            System.out.println("Server started. Waitting for client....");
            socket =server.accept();
            System.out.println("Client" + socket.getInetAddress()+ "is connect");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            String line ="";
            while(!line.equals("bye")){
                try{
                    //nhận số từ Client
                    line = in.readLine();
                    int a = Integer.parseInt(line);
                    System.out.println("Server received: " + a);
                    if(ktSoHoanHao(a)==1){
                        out.write(a +" là số hoàn hảo");
                    }
                    else{
                        out.write(a+" không phải là số hoàn hảo"+a+" nhat la: "+soHHKeSau(a));
                    }
                    out.newLine();
                    out.flush();// đẩy dữu liệu qua
                }catch(Exception e){
                    System.out.println(e);
                }
            }
            in.close();
            out.close();
            socket.close();
            server.close();
        } catch (Exception e){
            System.err.println(e);
        }
    }
    public static void main(String[] args){
        server server = new server(6000);
    }
```

Bài 3:

Viết chương trình phân tích số, hoạt động theo mô hình client-server, sử dụng TCP socket

Client gửi số nguyên dương n >= 10 đến server.

Server phân tích n thành tích các số nguyên tố và gửi trả ngược lại client

Client xuất kết quả ra console

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class client {
    Socket socket =null;
    BufferedWriter out = null;
    BufferedReader in = null;
    BufferedReader stdIn = null;

    public client(String host, int port){
        try{
            socket = new Socket(host, port);
            System.out.println("Client connected");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line="";
            while(!line.equals("bye")){
                try{
                    //nhập số cần phân tích
                    line = stdIn.readLine();
                    int a = Integer.parseInt(line);
                    if(a<=10){
                        System.out.println("Nhập lại:");
                        line = stdIn.readLine();
                    }
                    System.out.println("Client sent:" + line);
                    out.write(line);
                    out.newLine();
                    out.flush();// đẩy dữu liệu qua server

                    //nhận kết qua phân tích từ server
                    String da= in.readLine();
                    System.out.println("Result:" +da);
                }catch(Exception e){
                    System.err.println(e);
                }
            }
            in.close();
            out.close();
            stdIn.close();
            socket.close();
        }catch(Exception e){
            System.out.println(e);
        }
    }

    public static void main(String[] args){
            client client = new client("localhost",6000);
    }
}
```

Server:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import static java.lang.Math.sqrt;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.List;

public class server {
    ServerSocket server = null;
    Socket socket =null;
    BufferedWriter out =null;
    BufferedReader in =null;
    StringBuffer stringBuffer= null;
    /*public static List<Integer> phantichSNT(int n){
        int i=2;
        List<Integer> list = new ArrayList<Integer>();
        //phân tích
        while(n>1){
            if(n%i==0){
                n=n/i;
                list.add(i);
            }
            else{
                i++;
            }
        }
        //nếu list trống thì add n vào
        if(list.isEmpty()){
            list.add(n);
        }
        return list;
    }*/
    public static boolean ktrsnt(int n){
        if(n<=1)
            return false;
        for(int i=2; i<=sqrt(n);i++)
            if(n%i==0)
                return false;
        return true;
    }
    public server(int port){
        try{
            server =new ServerSocket(port);
            System.out.println("Server started. Waitting for client....");
            socket =server.accept();
            System.out.println("Client" + socket.getInetAddress()+ "is connect");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            String line ="";
            while(!line.equals("bye")){
                    //nhận số từ Client
                    line = in.readLine();
                    int a = Integer.parseInt(line);
                    System.out.println("Server received: " + a);
                    //xử lý phân tích thành các số nguyên tố
                    /*List<Integer> list= phantichSNT(a);
                    int size= list.size();
                    for(int i=0;i<size-1;i++){
                        out.write(list.get(i)+ "x");
                    }*/
                    for(int i=2;i<=a;i++){
                        while(a%i==0 && ktrsnt(i)){
                           out.write(i+"x");
                           a=a/i;
                        }
                    }
                    out.newLine();
                    out.flush();// đẩy dữu liệu qua
                }
                in.close();
                out.close();
                socket.close();
                server.close();
            } catch (Exception e){
                System.err.println(e);
            }
        }
        public static void main(String[] args){
            server server = new server(6000);
        }
    }
```

Bài 4:

Viết chương trình đoán số, hoạt động theo mô hình client-server, sử dụng TCP socket

Khi client kết nối, server tạo sẵn 1 số nguyên ngẫu nhiên n <= 100

Client đoán số do server tạo, nếu không đúng, server cần gợi ý bằng cách cho biết số client gửi lớn hơn hay nhỏ hơn n.

Quá trình lặp liên tục cho tới khi client gửi đúng số = n. Server xuất các thống kê: số lần client đoán, tổng thời gian đoán.

Client::

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class BT4Client {
    Socket socket = null;
    BufferedWriter out = null;
    BufferedReader in = null;
    BufferedReader stdIn = null;

    public BT4Client(String host, int port){
        try{
            socket = new Socket(host, port);
            System.out.println("Client connected");
            System.out.println("Hay doan 1 con so");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line ="";
            String newLine =" ";
            //String random=in.readLine();
            while(!line.equals(newLine)){
                line = stdIn.readLine();
                try{
                    int a=Integer.parseInt(line);
                    if(a<=100){
                        System.out.println("Client sent: "+line);
                        out.write(line);
                        out.newLine();
                        out.flush();

                        newLine=in.readLine();
                        System.out.println("Client received: "+newLine);
                    }
                    else{
                        System.out.println("Vui long nhap so <=100.");
                    }

                }catch(NumberFormatException e){
                    System.out.println("Vui long nhap so <=100.");
                }
            }
            newLine=in.readLine();
            System.out.println("Client received: "+newLine);
            in.close();
            out.close();
            stdIn.close();
            socket.close();

        }catch(Exception e){
            System.err.println(e);
        }
    }

    public static void main (String[] args){
        BT4Client client = new BT4Client("localhost", 6000);
    }
}
```

Server:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import static java.lang.Math.sqrt;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.HashMap;
import java.util.Map;
import java.util.Random;
import java.util.Set;
import java.util.StringTokenizer;

public class BT4Server {
    ServerSocket server = null;
    Socket socket = null;
    BufferedWriter out =null;
    BufferedReader in = null;

    public BT4Server(int port){
        try{
            server = new ServerSocket(port);
            System.out.println("Server started. Waiting for client........");
            socket = server.accept();
            System.out.println("Client " +socket.getInetAddress()+" is connected.");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            Random generator = new Random();
            int n=generator.nextInt(100)+1;
            System.out.println("So ngau nhien: "+n);
            String line="";
            String newLine="";
            while(!line.equals(String.valueOf(n))){
                line = in.readLine();
                int t=Integer.parseInt(line);
                System.out.println("Server received: "+line);
                if(n<t)
                    newLine="Sai. Goi y: So can doan be hon so nay.";
                else
                    newLine="Sai. Goi y: So can doan lon hon so nay.";
                System.out.println("Server sent: "+newLine);
                out.write(newLine);
                out.newLine();
                out.flush();
            }
            newLine="Chinh xac!";
            System.out.println("Server sent: "+newLine);
            /*out.write(newLine);
            out.newLine();
            out.flush();*/
            in.close();
            out.close();
            socket.close();
            server.close();

        }catch(Exception e){
            System.err.println(e);
        }
    }

    public static void main (String[] args){
        BT4Server server = new BT4Server (6000);
    }
}
```

Bài 5:

Viết chương trình tính toán, hoạt động theo mô hình client-server, sử dụng TCP socket

Client gửi 1 chuỗi phép toán gồm 2 số và 1 trong 4 phép toán (+, -, *, /) đến server

Server phân tích chuỗi, tính kết quả và trả lại client hoặc trả thông báo lỗi nếu chuỗi phép toán không đúng format.

Client xuất kết quả ra console

Client:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class BT5Client {
    Socket socket = null;
    BufferedWriter out = null;
    BufferedReader in = null;
    BufferedReader stdIn = null;

    public BT5Client(String host, int port){
        try{
            socket = new Socket(host, port);
            System.out.println("Client connected");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line ="";
            String newLine ="";
            while(!line.equals("bye")){
                line = stdIn.readLine();

                System.out.println("Client sent: "+line);
                out.write(line);
                out.newLine();
                out.flush();

                newLine=in.readLine();
                System.out.println("Client received: "+newLine);

            }
            in.close();
            out.close();
            stdIn.close();
            socket.close();

        }catch(Exception e){
            System.err.println(e);
        }
    }

    public static void main (String[] args){
        BT5Client client = new BT5Client("localhost", 6000);
    }
}
```

Server:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import static java.lang.Math.sqrt;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.HashMap;
import java.util.Map;
import java.util.Set;
import java.util.StringTokenizer;

public class BT5Server {
    ServerSocket server = null;
    Socket socket = null;
    BufferedWriter out =null;
    BufferedReader in = null;

    public BT5Server(int port){
        try{
            server = new ServerSocket(port);
            System.out.println("Server started. Waiting for client........");
            socket = server.accept();
            System.out.println("Client " +socket.getInetAddress()+" is connected.");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            String line="";
            while(!line.equals("bye")){
                line = in.readLine();
                String newLine="";
                System.out.println("Server received: "+line);
                String pattern="\\d{0,6}[*/+-]\\d{0,6}";
                if(line.matches(pattern)==false){
                    newLine="Khong dung format";
                }
                else{
                    int a,b;
                    double result = 0;
                    String op;
                    StringTokenizer  st = new StringTokenizer(line, "+-*/",true);
                    a=Integer.parseInt(st.nextToken());
                    op=st.nextToken();
                    b=Integer.parseInt(st.nextToken());

                    switch(op){
                        case "+":{
                            newLine+=a+b;
                            break;
                        }
                        case "-":{
                            newLine+=a-b;
                            break;
                        }
                        case "*":{
                            newLine+=a*b;
                            break;
                        }
                        case "/":{
                            newLine+=a*1.0/b;
                            break;
                        }
                    }

                }
                System.out.println("Server sent: "+newLine);
                out.write(newLine);
                out.newLine();
                out.flush();
            }
            System.out.println("Closing connection");
            in.close();
            out.close();
            socket.close();
            server.close();

        }catch(Exception e){
            System.err.println(e);
        }
    }

    public int kt_snt(long n){
        if(n<2)
            return 0;
        double k=sqrt(n);
        for(int i=2;i<=k;i++)
            if(n%i==0)
                return 0;
            return 1;
    }

    public static void main (String[] args){
        BT5Server server = new BT5Server (6000);
    }

}
```

Bài 6:

Viết chương trình tìm thông tin IP, hoạt động theo mô hình client-server, sử dụng TCP socket.

Tra thông tin server (chỉ dùng khi server hoạt động ở một mạng có NAT): client gửi lệnh hello đến server. Server trả lại client public IP và private IP của server.

Tra cứu IP: client gửi lệnh req x, với x là một địa chỉ IP public. Server trả lại client các thông tin về IP x gồm: thành phố - quốc gia – châu lục mà IP đó thuộc về hoặc trả về thông báo lỗi nếu IP không đúng format/IP private.

Trường hợp client gửi không đúng cú pháp, server trả về hướng dẫn sử dụng chương trình.

Client:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class clientbai2 {
    Socket socket =null;
    BufferedWriter out = null;
    BufferedReader in = null;
    BufferedReader stdIn = null;

    public clientbai2(String host, int port){
        try{
            socket = new Socket(host, port);
            System.out.println("Client connected");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line="";
            while(!line.equals("bye")){
                try{
                    //nhập dữ liệu
                    line = stdIn.readLine();

                    System.out.println("Client sent:" + line);
                    out.write(line);
                    out.newLine();
                    out.flush();// đẩy dữu liệu qua server

                    //nhận kết quả từ server
                    String da= in.readLine();
                    System.out.println(da);
                    System.out.println("------------------------------------------------------------");
                }catch(Exception e){
                    System.err.println(e);
                }
            }
            in.close();
            out.close();
            stdIn.close();
            socket.close();
        }catch(Exception e){
            System.out.println(e);
        }
    }

    public static void main(String[] args){
            clientbai2 client = new clientbai2("localhost",6000);
    }
}
```

Server:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.URL;

public class serverbai2 {
    ServerSocket server = null;
    Socket socket =null;
    BufferedWriter out =null;
    BufferedReader in =null;
    StringBuffer stringBuffer= null;
    BufferedReader in_server_pub_IP = null;
    BufferedReader in_info_IP = null;
    public serverbai2(int port){
        try{
            server = new ServerSocket(port);
            while(true){
                try{
                    System.out.println("Server started. Waiting for client........");
                    socket = server.accept();
                    System.out.println("Client " +socket.getInetAddress().getHostAddress()+" is connected.");
                    out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
                    in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

                    while(true){
                        String line = in.readLine();
                        String newLine="";
                        if(line.equals("bye"))
                            break;
                        System.out.println("Server received: "+line);
                        //client gui "hello"
                        if(line.equals("hello")){
                            URL pubIP = new URL("http://checkip.amazonaws.com");
                            in_server_pub_IP = new BufferedReader(new InputStreamReader(pubIP.openStream()));
                            String ip = in_server_pub_IP.readLine();
                            newLine="Server's public IP: "+ip+" - Server's private IP: "+server.getInetAddress().getHostAddress() ;
                            System.out.println(newLine);
                        }
                        else{
                            String w1 = line.substring(0,3);//lay chu req
                            if(!w1.equals("req")||line.charAt(3)!=' ')
                                newLine="Không đúng cú pháp.....";
                            else{
                                //client gui req x
                                try{
                                    String w2 = line.substring(4,7);
                                    String w3 = line.substring(4,11);
                                    String w4 = line.substring(4); //lay dia chi ip
                                    String pattern="\\d{1,3}.\\d{1,3}.\\d{1,3}.\\d{1,3}$";

                                        if(w4.matches(pattern)==false){
                                            newLine="IP này không đúng format";
                                        }
                                        else{
                                            if(w2.equals("10.")||w3.equals("192.168")||w3.equals("172.16."))
                                                newLine="Đây là private IP.";
                                            else{
                                                URL infoIP = new URL("http://ip-api.com/line/"+w4+"?fields=status,message,country,city,timezone,query");
                                                in_info_IP = new BufferedReader(new InputStreamReader(infoIP.openStream()));
                                                String status = in_info_IP.readLine();
                                                if(status.equals("success")){
                                                    String country = in_info_IP.readLine();
                                                    String city = in_info_IP.readLine();
                                                    String timezone = in_info_IP.readLine();
                                                    String continent = "";
                                                    for(int i=0;i<timezone.length();i++){
                                                        if(timezone.charAt(i)=='/'){
                                                            continent=timezone.substring(0,i);
                                                            break;
                                                        }
                                                    }
                                                    newLine="Country: "+country+" - City: "+city+" - Continent: "+continent;
                                                }
                                                else System.out.println("IP không tồn tại");
                                            }
                                        }
                                    }catch(Exception e){
                                        newLine="Không đúng cú pháp.";
                                    }
                            }
                        }
                    }
                }
            }
```

```
                    System.out.println("Server sent: "+newLine);
                    out.write(newLine);
                    out.newLine();
                    out.flush();
                }
                System.out.println("Server closed connection");
                in.close();
                in_server_pub_IP.close();
                out.close();
                socket.close();
                //server.close();
            }catch(IOException e){
                System.out.println("Unexpected connection close "+socket.getInetAddress().getHostAddress());
            }
        }
    }catch(Exception e){
        System.err.println(e);
    }
}
public static void main(String[] args){
    serverbai2 server = new serverbai2(6000);
}
}
```

Bài 7:

Viết chương trình tra cứu thông tin cá nhân, hoạt động theo mô hình client-server, sử dụng TCP socket

Client gửi 1 số chứng minh nhân dân/căn cước công dân người VN đến server.

Server tìm kiếm họ tên, quê quán tương ứng với số CMND/CCCD đó và gửi trả ngược lại client hoặc trả thông báo lỗi nếu không tìm thấy thông tin.

Client xuất kết quả ra console.

Server:

```
import java.io.*;
import java.net.*;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.select.Elements;

public class Serverbai4 {
    ServerSocket server = null;
    Socket socket = null;
    BufferedReader in = null;
    BufferedWriter out =null;
    StringBuffer stringBuffer;

    public Serverbai4(int port){
        try{
            server = new ServerSocket(port);
            System.out.println("Server starting.....");
            socket = server.accept();
            System.out.println("Client accepted");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            String line="";
```

```
            while(!line.equals("bye")){
                try{
                    //nhận chuỗi từ Client
                    line = in.readLine();
                    System.out.println("Server received: "+line);
                    String linesent = "";
                    if (line.equals("bye")) {
                        linesent = "stop";
                    } else {
                        String[] result = GetCMND(Long.valueOf(line));
                        System.out.println("");
                        String sb = "";
                        for (int j = 0; j < result.length; j++) {
                            sb+= result[j];
                            sb+="; ";
                        }
                        linesent = sb;
                    }

                    System.out.println("Sever send: " + linesent);
                    out.write(linesent);
                    out.newLine();
                    out.flush();//đẩy dữ liệu qua Client

                }catch(Exception e){
                    System.err.println(e);
                }
            }
            in.close();
            out.close();
            socket.close();
            server.close();
        }catch(Exception e){
            System.err.println(e);
        }
    }
    public String[] GetCMND(long cmnd){
        String[] res = null;
        try {
            // neu tim CMND khong co = loi
            String url = Jsoup.connect("https://masothue.com/Search/?q=" + cmnd + "&type=auto&force-search=1")
                    .followRedirects(true) // cho phep di
                    .execute()
                    .url()
                    .toExternalForm(); // convert sang String
            Document doc = Jsoup.connect(url).get();

            Elements info = doc.getElementsByClass("copy");
            res = new String[info.size()];
            for(int i = 0 ; i < info.size(); i++){
                res[i] = info.get(i).text();
                System.out.println("" + res[i]);
            }
        } catch (IOException e) {
            System.err.println(e);
        }
        return res;
    }
    public static void main(String[] args){
        Serverbai4 server = new Serverbai4(6000);
    }
}
```

Client:

```
import java.net.*;
import java.io.*;
public class Clientbai4 {
    Socket socket = null;
    BufferedReader in = null;
    BufferedWriter out = null;
    BufferedReader stdIn = null;

    public Clientbai4(String host, int port){
        try{
            socket = new Socket(host,port);
            System.out.println("Client connected.");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line = "";
            while(!line.equals("bye")){
                try{
                    System.out.println("Nhap CNMD: ");
                    line = stdIn.readLine();
                    System.out.println("CMND: "+line);
                    out.write(line);
                    out.newLine();
                    out.flush();//đẩy dữ liệu qua Server

                    //nhận kết quả từ Server
                    String data = in.readLine();
                    System.out.println("Result: "+data);

                }catch(Exception e){
                    System.err.println(e);
                }
            }
            in.close();
            out.close();
            stdIn.close();
            socket.close();
        }catch(Exception e){
            System.err.println(e);
        }
    }

    public static void main(String[] args){
        Clientbai4 client = new Clientbai4("localhost",6000);
    }
}
```

Bài 8:

Viết chương trình tính toán, hoạt động theo mô hình client-server, sử dụng TCP socket

Client gửi 1 chuỗi phép toán gồm nhiều số phân cách nhau bởi 1 trong 4 phép toán (+, -,*, /) đến server, giả sử chuỗi phép toán không chứa các dấu ngoặc. Ví dụ chuỗi phép toán sau: 12+34-56*78/4+14-17

Server phân tích chuỗi, tính kết quả và trả lại client hoặc trả thông báo lỗi nếu chuỗi phép toán không đúng format.

Client xuất kết quả ra console

Client:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class clientbai3 {
    Socket socket = null;
    BufferedWriter out = null;
    BufferedReader in = null;
    BufferedReader stdIn = null;

    public clientbai3(String host, int port){
        try{
            socket = new Socket(host, port);
            System.out.println("Client connected");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line ="";
            String newLine ="";
            while(!line.equals("bye")){
                line = stdIn.readLine();

                System.out.println("Client sent: "+line);
                out.write(line);
                out.newLine();
                out.flush();

                newLine=in.readLine();
                System.out.println(newLine);

            }
            in.close();
            out.close();
            stdIn.close();
            socket.close();

        }catch(Exception e){
            System.err.println(e);
        }
    }

    public static void main (String[] args){
        clientbai3 client = new clientbai3("localhost", 6000);
    }
}
```

Server:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import static java.lang.Math.sqrt;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;
import java.util.Set;
import java.util.StringTokenizer;

public class severbai3 {
    ServerSocket server = null;
    Socket socket = null;
    BufferedWriter out =null;
    BufferedReader in = null;
    private int result;
    private ArrayList<String> ListNumber;
    private ArrayList<String> ListCal;
    private String stCalculation;

    public severbai3(String stCalculation){
        this.stCalculation = stCalculation;
    }
    public int Tinh(String stCalculation){
        ArrayList ListNumber=new ArrayList<>();
        ArrayList ListCal=new ArrayList<>();
        StringTokenizer st = new StringTokenizer(stCalculation, "+-*/",true);
        while(st.hasMoreTokens()){
            ListNumber.add(st.nextToken());
            if(st.hasMoreTokens()){
                ListCal.add(st.nextToken());
            }
        }
        //Tinh phep nhan chia truoc
        for(int i=0; i<ListCal.size();i++){
            if(ListCal.get(i).toString().equals("*")){
                int t = Integer.parseInt((String) ListNumber.get(i))*Integer.parseInt((String) ListNumber.get(i+1));
                ListCal.remove(i);
                ListNumber.set(i, t+"");
                ListNumber.remove(i+1);
                i=0;
            }
            if(ListCal.get(i).toString().equals("/")){
                int t1 = Integer.parseInt((String) ListNumber.get(i))/Integer.parseInt((String) ListNumber.get(i+1));
                ListCal.remove(i);
                ListNumber.set(i, t1+"");
                ListNumber.remove(i+1);
                i=0;
            }
        }
        //Tinh phep cong tru
        for(int i=0; i<ListCal.size();i++){
            if(ListCal.get(i).toString().equals("+")){
                int t = Integer.parseInt((String) ListNumber.get(i))+Integer.parseInt((String) ListNumber.get(i+1));
                ListCal.remove(i);
                ListNumber.set(i, t+"");
                ListNumber.remove(i+1);
                i=0;
            }
            if(ListCal.get(i).toString().equals("-")){
                int t1 = Integer.parseInt((String) ListNumber.get(i))-Integer.parseInt((String) ListNumber.get(i+1));
                ListCal.remove(i);
                ListNumber.set(i, t1+"");
                ListNumber.remove(i+1);
                i=0;
            }
        }
        return result = Integer.parseInt((String) ListNumber.get(0));
    }
```

```
public severbai3(int port){
    try{
        server = new ServerSocket(port);
        System.out.println("Server started. Waiting for client.....");
        socket = server.accept();
        System.out.println("Client " +socket.getInetAddress()+" is connected.");
        out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
        in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
        String line="";
        while(!line.equals("bye")){
            line = in.readLine();
            String newLine="";
            System.out.println("Server received: "+line);
            int kq=Tinh(line);
            //String pattern="\\d{0,6}[*/+-]\\d{0,6}";
            //if(line.matches(pattern)==false){
            //    newLine="Khong dung format";
            //}
            //else{
                /*int a,b;
                double result = 0;
                String op*/
                //StringTokenizer  st = new StringTokenizer(line, "+-*/",true);
                /*a=Integer.parseInt(st.nextToken());
                op=st.nextToken();
                b=Integer.parseInt(st.nextToken());

                switch(op){
                    case "+":{
                        newLine+=a+b;
                        break;
                    }
                    case "-":{
                        newLine+=a-b;
                        break;
                    }
                    case "*":{
                        newLine+=a*b;
                        break;
                    }
                    case "/":{
                        newLine+=a*1.0/b;
                        break;
                    }
                }*/

            //}
            //System.out.println("Server sent: "+newLine);
            out.write("Kết quả:" +kq);
            out.newLine();
            out.flush();
        }
        System.out.println("Closing connection");
        in.close();
        out.close();
        socket.close();
        server.close();

    }catch(Exception e){
        System.err.println(e);
    }
}
```

```
    /*public int kt_snt(long n){
        if(n<2)
            return 0;
        double k=sqrt(n);
        for(int i=2;i<=k;i++)
            if(n%i==0)
                return 0;
            return 1;
    }*/

    public static void main (String[] args){
        severbai3 server = new severbai3 (6000);
    }

}
```

Bài 9:

Viết chương trình tra thông tin và điểm sinh viên trên cổng thông tin đào tạo, sử dụng mô hình client-server, dựa trên TCPSocket.

Client gửi 1 mã số sinh viên đến server và chờ nhận kết quả phản hồi.

Server thực hiện 2 chức năng dưới đây và gửi dữ liệu về lại client:

o Chức năng 1: tra cứu họ tên, nơi sinh, ngành, khóa đào tạo.

o Chức năng 2: điểm chi tiết học kỳ 1 năm học 2019-2020 (gồm: mã học phần, tên học phần, điểm kết thúc học phần – hệ 10)

client:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;

public class clientbai1 {
    Socket socket =null;
    BufferedWriter out = null;
    BufferedReader in = null;
    BufferedReader stdIn = null;

    public clientbai1(String host, int port){
        try{
            socket = new Socket(host, port);
            System.out.println("Client connected ");
            System.out.println("Nhập MSSV:  ");
            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            stdIn = new BufferedReader(new InputStreamReader(System.in));
            String line="";
            while(!line.equals("bye")){
                try{
                    //nhập mssv cần tra cứu
                    line = stdIn.readLine();
                    System.out.println("Client sent");
                    System.out.println("MSSV: " + line);
                    out.write(line);
                    out.newLine();
                    out.flush();// đẩy dữu liệu qua server

                    //nhận kết quả từ server
                    String da= in.readLine();
                    System.out.println("Kết quả: " +da);
                    System.out.println("------------------------------------------------------------");
                }catch(Exception e){
                    System.err.println(e);
                }
            }
            in.close();
            out.close();
            stdIn.close();
            socket.close();
        }catch(Exception e){
            System.out.println(e);
        }
    }

    public static void main(String[] args){
            clientbai1 client = new clientbai1("localhost",6000);
    }
}
```

Server

```
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathFactory;
import java.io.*;
import java.lang.System.*;
import java.net.*;
import java.util.*;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.xml.xpath.XPath;
import org.jsoup.Connection;
import org.jsoup.Jsoup;
import org.jsoup.nodes.*;
import org.jsoup.select.Elements;
public class serverbai1{
    String res ;
    private Socket socket = null;
    private ServerSocket serverSocket = null;
    BufferedReader in = null;
    BufferedWriter out = null;

    public serverbai1(int port) {
        try {
            System.out.println("Server is ready ");
            serverSocket = new ServerSocket(port);
            System.out.println("Server started !");
            System.out.println("Waiting for a client ...");
            socket = serverSocket.accept();
            System.out.println("Client accepted !");

            in = new BufferedReader(new InputStreamReader(
                    socket.getInputStream()));
            out = new BufferedWriter(new OutputStreamWriter(
                    socket.getOutputStream()));

            String linerecived = "";
            while (!linerecived.equals("bye")) {
                linerecived = in.readLine();
                System.out.println("Received from a client: " + linerecived);
                String linesent = "";
                if (linerecived.equals("bye")) {
                    linesent = "Dừng";
                } else {
                    final long startTime =System.currentTimeMillis();
                    String url= "http://thongtindaotao.sgu.edu.vn/Default.aspx?page=xemdiemthi&id="+linerecived;
                    String domain ="http://thongtindaotao.sgu.edu.vn/";
                    try{
                        String urlLog =domain +"default.aspx?page=nhapmasv&flag=XemDiemThi";

                    String userAgent="Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Safari/537.36";
                    Connection.Response response = Jsoup.connect(urlLog)
                              .method(Connection.Method.GET)
                              .execute();
                    Document login = response.parse();
```

```
                //Lấy thông tin sinh viên và điểm học kỳ gần nhất
                response =Jsoup.connect(urlLog)
                                .data("__EVENTTAGET","")
                                .data("__EVENTARGUMENT", "")
                                .data("__VIEWSTATE", login.getElementById("__VIEWSTATE").val())
                                .data("__VIEWSTATEGENERATOR", login.getElementById("__VIEWSTATEGENERATOR").val())
                                .data("ctl00$ContentPlaceHolder1$ctl00$txtMaSV",linerecived)
                                .data("ctl00$ContentPlaceHolder1$ctl00$btnOK","OK")
                                .userAgent(userAgent)
                                .timeout(0)
                                .followRedirects(true)
                                .cookies(response.cookies())
                                .method(Connection.Method.POST)
                                .execute();
                login=response.parse();
                String maSV= login.getElementById("ctl00_ContentPlaceHolder1_ctl00_ucThongTinSV_lblMaSinhVien").text();
                String hoTen= login.getElementById("ctl00_ContentPlaceHolder1_ctl00_ucThongTinSV_lblTenSinhVien").text();
                String noiSinh= login.getElementById("ctl00_ContentPlaceHolder1_ctl00_ucThongTinSV_lblNoiSinh").text();
                String nganh= login.getElementById("ctl00_ContentPlaceHolder1_ctl00_ucThongTinSV_lbNganh").text();
                String khoa= login.getElementById("ctl00_ContentPlaceHolder1_ctl00_ucThongTinSV_lblKhoa").text();
                System.out.println(maSV+ " - "+hoTen+ " - "+noiSinh+ " - "+nganh+ " - "+khoa);
                linesent = maSV+ " - "+hoTen+ " - "+noiSinh+ " - "+nganh+ " - "+khoa;
                //Lấy điểm học kỳ bất kỳ
                XPath xpath = XPathFactory.newInstance().newXPath();
                String url1= domain +"Default.aspx?page=xemdiemthi&id="+linerecived;
                response = Jsoup.connect(url1)
                                .method(Connection.Method.GET)
                                .execute();
                response = Jsoup.connect(url1)
                                .data("__EVENTTARGET","")
                                .data("__EVENTARGUMENT","")
                                .data("__VIEWSTATE",login.getElementById("__VIEWSTATE").val())
                                .data("__VIEWSTATEGENERATOR",login.getElementById("__VIEWSTATEGENERATOR").val())
                                .data("ctl00$ContentPlaceHolder1$ctl00$txtChonHK", "20191")
                                .data("ctl00$ContentPlaceHolder1$ctl00$btnChonHK","Xem")
                                .userAgent(userAgent)
                                .timeout(0)
                                .followRedirects(true)
                                .cookies(response.cookies())
                                .method(Connection.Method.POST)
                                .execute();
                Document doc = response.parse();
                Elements diemTK = doc.selectXpath("//*[@id=\"ctl00_ContentPlaceHolder1_ctl00_div1\"]/table/tbody/tr[9]/td/span[2]");
                res = diemTK.text();
                /*for(Element ex : diemTK)
                    System.out.println(ex.text());
                   res = new String[diemTK.size()];
                    for(int i = 0 ; i < diemTK.size(); i++){
                        res[i] = diemTK.get(i).text();
                        System.out.println("" + res[i]);
                }*/
            } catch (IOException e) {
                       System.out.println(e);
```

```
                    }
                }
                System.out.println("Send to a client: " + linesent+ "; Điểm: "+res);
                out.write(linesent+"; Điểm: "+ res);
                out.newLine();
                out.flush();
                System.out.println("----------------------------------------------------------");
            }
            in.close();
            out.close();
            socket.close();
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
    public static void main(String[] args) {
        serverbai1 server= new serverbai1(6000);
    }

}
```

Bài 10:

Viết chương trình gửi tin nhắn hai chiều giữa client-server sử dụng UDP socket.

Client gửi 1 chuỗi ký tự bất kỳ đến server

Server nhận và gửi chuỗi đảo ngược từng từ về client.

Client xuất kết quả ra console, chương trình kết thúc khi client gửi chuỗi bye.

VD: client gửi chuỗi hello world, server trả lại chuỗi olleh dlrow

Client::

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
import java.net.UnknownHostException;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;

public class client {

    public static int destPort =1234;
    public static String hostname=  "localhost";
    public static void main(String[] args) throws SocketException, IOException {
        try {
            DatagramSocket socket;
            DatagramPacket send, receive;
            InetAddress add;
            Scanner stdIn;
            add= InetAddress.getByName(hostname);
            socket =new DatagramSocket();
            stdIn=new Scanner(System.in);
            while(true){
                System.out.print("Client input: ");
                String tmp =stdIn.nextLine();
                byte[] data=tmp.getBytes();
                send= new DatagramPacket(data,data.length,add,destPort);
                System.out.println("Client sent "+tmp+" to "+add.getHostAddress()+" form port "+socket.getLocalPort());
                socket.send(send);
                if(tmp.equals("bye")){
                    System.out.println("Client socket closed");
                    stdIn.close();
                    socket.close();
                    break;

                }

                receive =new DatagramPacket(new byte[9999],9999);
                socket.receive(receive);
                tmp=new String (receive.getData(),0,receive.getLength());
                System.out.println("Client get: "+tmp+" from server");
            }
        } catch (UnknownHostException ex) {
            Logger.getLogger(client.class.getName()).log(Level.SEVERE, null, ex);
        }

    }

}
```

Server:

```
import java.util.logging.Level;
import java.util.logging.Logger;

import java.io.File;
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;
import java.net.UnknownHostException;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;
import java.util.StringTokenizer;
import java.util.logging.Level;
import java.util.logging.Logger;

public class server {
    public static int buffsize =9999;
    public static int port =1234;

    public static void main(String[] args) throws SocketException, IOException {

            DatagramSocket socket;
            DatagramPacket send, receive;
            String s = null;

            try{
            socket=new DatagramSocket(port);
            receive =new DatagramPacket(new byte[buffsize],buffsize);

            while(true){
                socket.receive(receive);
                String tmp=new String (receive.getData(),0,receive.getLength());
                System.out.println("Server receive: "+tmp+" from "+receive.getAddress().getHostAddress()+" at port "+socket.getLocalPort());
                if(tmp.equals("bye")){
                    System.out.println("Server socket closed");
                    socket.close();
                    break;
                }
                StringBuilder newline = new StringBuilder();
                newline.append(tmp);
                s = newline.reverse().toString();
                send=new DatagramPacket(s.getBytes(),s.getBytes().length,receive.getAddress(),receive.getPort());
                System.out.println("Server sent back "+s+" to client");
                socket.send(send);
            }
        } catch (UnknownHostException ex) {
            Logger.getLogger(client.class.getName()).log(Level.SEVERE, null, ex);
        }

    }
}
```