The University of Jordan
School of Engineering
Department of Computer Engineering
First semester (2023-2024).

Network Programming
(Lost Robot)

Part-A

Program-1:

Explanation:

Program 1 (on Dr. Kamal's computer): Ask the Super Robot for its IP address via the UDP protocol.
Send the Super Robot a packet with the word "STOP" as the payload.
On Dr. Kamal's computer, get the reply packet with the payload "OK".
Using the RSSI value found in the reply packet, determine r1, the separation between Dr. Kamal's computer and the Super Robot.
Use the UDP protocol to send r1, x1, and y1 (Dr. Kamal's location) to Husam's computer in a single packet.

Source code:

```
import java.net.*;
import java.io.*;
// Program 1: Dr Kamal's computer
public class Program1 {
    private static final int KAMAL_PORT = 5001;
    private static final int ROBOT_PORT = 6000;
    private static final int HUSAM_PORT = 7000;

    private static final String ROBOT_IP = "193.188.40.1";

    public static void main(String[] args) {
        try {
            DatagramSocket socket = new DatagramSocket(KAMAL_PORT);

            // Ask Super Robot for its IP address
            InetAddress robotAddress = InetAddress.getByName(ROBOT_IP);
            // Send a packet with payload "STOP" to the Super Robot
            String stopCommand = "STOP";
            DatagramPacket stopPacket = new DatagramPacket(stopCommand.getBytes(),
stopCommand.length(), robotAddress, ROBOT_PORT);
            socket.send(stopPacket);

            // Receive the reply packet with payload "OK" from the Super Robot
            byte[] buffer = new byte[256];
            DatagramPacket replyPacket = new DatagramPacket(buffer, buffer.length);
            socket.receive(replyPacket);
            String replyPayload = new String(replyPacket.getData(), 0, replyPacket.getLength());

            // Calculate r1 based on received RSSI
            int receivedRSSI = Integer.parseInt(replyPayload);
            double C = 56.0; // Example value, adjust as needed
            double N = 2.5; // Example value, adjust as needed
            double r1 = Math.pow(10, (C - receivedRSSI) / (10 * N));


            // Send r1, x1, and y1 to Husam's computer
            double x1 = 1; // assumtion
            double y1 = 1; // assumtion
            String dataToSend = r1 + "," + x1 + "," + y1; // Replace x1 and y1 with actual values
            InetAddress husamAddress = InetAddress.getByName("193.188.41.1");
            DatagramPacket sendPacket = new DatagramPacket(dataToSend.getBytes(),
dataToSend.length(), husamAddress, HUSAM_PORT);
            socket.send(sendPacket);

            socket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

}
```

Program-2:

Explanation:

Utilize the UDP protocol to catch a packet with the payload "STOP" in Program 2 (on the Super Robot).
response with the payload "OK" to the packet.

Source code:

```
// Program 2: Super Robot
import java.net.*;
import java.io.*;
 public class Program2 {
    private static final int ROBOT_PORT = 5001;
    private static final int KAMAL_PORT = 5000;

    public static void main(String[] args) {
        try {
            DatagramSocket socket = new DatagramSocket(ROBOT_PORT);

            // Receive a packet with payload "STOP" from Dr Kamal's computer
            byte[] buffer = new byte[256];
            DatagramPacket receivePacket = new DatagramPacket(buffer, buffer.length);
            socket.receive(receivePacket);
            String receivedPayload = new String(receivePacket.getData(), 0,
receivePacket.getLength());

            if (receivedPayload.equals("STOP")) {
                // Reply with a packet containing payload "OK"
                String okMessage = "OK";
                InetAddress kamalAddress = receivePacket.getAddress();
                int kamalPort = receivePacket.getPort();
                DatagramPacket replyPacket = new DatagramPacket(okMessage.getBytes(),
okMessage.length(), kamalAddress, kamalPort);
                socket.send(replyPacket);
            }

            socket.close();
        } catch (Exception e) {
            e.printStackTrace();}}}
```

Program-3:

Explanation:

On Husam's PC, program 3:
Obtain packets containing the following data from Dr. Kamal's and the LAB's computers: r1, r2, r3, x1, x2, x3, y1, y2, and y3.
Determine the spots where the three circles that are produced by the Super Robot's distances from Dr. Kamal's position, Husam's location, and the LAB's location connect.
From the junction points, ascertain the location of the Super Robot.
Using the UDP protocol, send the Super Robot's location to Dr. Kamal's computer and the computer in the LAB.

Source code:

```
// Program 3: Husam's computer
import java.net.*;
 import java.io.*;
 public class Program3 {
    private static final int HUSAM_PORT = 7001;
    private static final int KAMAL_PORT = 5000;
    private static final int LAB_PORT = 8000;
    public static void main(String[] args) {
        try {
            DatagramSocket socket = new DatagramSocket(HUSAM_PORT);
            // Receive r1, x1, and y1 from Dr Kamal's computer
            byte[] buffer = new byte[1024];
            DatagramPacket receivePacket = new DatagramPacket(buffer, buffer.length);
            socket.receive(receivePacket);
            String receivedData = new String(receivePacket.getData(), 0, receivePacket.getLength());
            String[] dataArr = receivedData.split(",");
             double r1 = Double.parseDouble(dataArr[0]);
             double x1 = Double.parseDouble(dataArr[1]);
            double y1 = Double.parseDouble(dataArr[2]);
            // Receive r2, x2, and y2 from Husam's computer
            byte[] buffer2 = new byte[1024];
            DatagramPacket receivePacket2 = new DatagramPacket(buffer2, buffer2.length);
            socket.receive(receivePacket2);
            String receivedData2 = new String(receivePacket2.getData(), 0,
receivePacket2.getLength());
            String[] dataArr2 = receivedData2.split(",");
            double r2 = Double.parseDouble(dataArr2[0]);
            double x2 = Double.parseDouble(dataArr2[1]);
            double y2 = Double.parseDouble(dataArr2[2]);

            // Receive r3, x3, and y3 from LAB's computer
            byte[] buffer3 = new byte[1024];
            DatagramPacket receivePacket3 = new DatagramPacket(buffer3, buffer3.length);
            socket.receive(receivePacket3);
            String receivedData3 = new String(receivePacket3.getData(), 0,
receivePacket3.getLength());
            String[] dataArr3 = receivedData3.split(",");
            double r3 = Double.parseDouble(dataArr3[0]);
            double x3 = Double.parseDouble(dataArr3[1]);
            double y3 = Double.parseDouble(dataArr3[2]);

            // Solve location equations to determine the position of the Super Robot
            double A = 2 * (x1 - x3);
            double B = 2 * (y1 - y3);
            double C = Math.pow(r3, 2) - Math.pow(r1, 2) - Math.pow(x3, 2) + Math.pow(x1, 2) -
Math.pow(y3, 2) + Math.pow(y1, 2);

            double D = 2 * (x2 - x3);
            double E = 2 * (y2 - y3);
            double F = Math.pow(r3, 2) - Math.pow(r2, 2) - Math.pow(x3, 2) + Math.pow(x2, 2) -
Math.pow(y3, 2) + Math.pow(y2, 2);
            double x = (C * E - F * B) / (A * E - D * B);
            double y = (C * D - A * F) / (B * D - E * A);

            // Send Super Robot position to Dr Kamal's computer
            String robotPosition = x + "," + y;
            InetAddress kamalAddress = InetAddress.getByName("193.188.43.1");
            DatagramPacket kamalPacket = new DatagramPacket(robotPosition.getBytes(),
robotPosition.length(), kamalAddress, KAMAL_PORT);
            socket.send(kamalPacket);

            // Send Super Robot position to LAB's computer
            InetAddress labAddress = InetAddress.getByName("193.188.42.1");
            DatagramPacket labPacket = new DatagramPacket(robotPosition.getBytes(),
robotPosition.length(), labAddress, LAB_PORT);
            socket.send(labPacket);
            socket.close();
        } catch (Exception e) {
            e.printStackTrace(); }}}
```

Program-4:

Explanation:

Program 4 (on the computers at the LAB and owned by Dr. Kamal):
Obtain data packets detailing the location of the Super Robot from Husam's computer.
On the computer console, print the location of the Super Robot.

Source code:

```
// Program 4: Dr Kamal's computer and LAB's computer
import java.net.*;
 import java.io.*;
public class Program4 {
   private static final int KAMAL_PORT = 5001;
   private static final int LAB_PORT = 8000;

   public static void main(String[] args) {
      try {
         // Dr Kamal's computer
         DatagramSocket kamalSocket = new DatagramSocket(KAMAL_PORT);
         byte[] kamalBuffer = new byte[1024];
         DatagramPacket kamalPacket = new DatagramPacket(kamalBuffer, kamalBuffer.length);
         kamalSocket.receive(kamalPacket);
         String kamalData = new String(kamalPacket.getData(), 0, kamalPacket.getLength());
         System.out.println("Dr Kamal's computer received position: " + kamalData);

         // LAB's computer
         DatagramSocket labSocket = new DatagramSocket(LAB_PORT);
         byte[] labBuffer = new byte[1024];
         DatagramPacket labPacket = new DatagramPacket(labBuffer, labBuffer.length);
         labSocket.receive(labPacket);
         String labData = new String(labPacket.getData(), 0, labPacket.getLength());
         System.out.println("LAB's computer received position: " + labData);
         kamalSocket.close();
         labSocket.close();
      } catch (Exception e) {
         e.printStackTrace();
      }}}
```

Part-B

To approximate the location of the Super Robot when the circles do not intersect, Husam can use a technique called trilateration. Trilateration involves finding the point of intersection of the perpendicular bisectors of the line segments connecting the known locations to the Super Robot's estimated location.

Here are the changes to be made to Husam's Java program (Program-3) to implement trilateration:

- After receiving r1, x1, y1, r2, x2, y2, r3, x3, and y3, calculate the estimated position of the Super Robot using trilateration:

```
// Solve location equations to estimate the position of the Super Robot
double A = 2 * (x1 - x3);
double B = 2 * (y1 - y3);
double C = Math.pow(r3, 2) - Math.pow(r1, 2) - Math.pow(x3, 2) +
Math.pow(x1, 2) - Math.pow(y3, 2) + Math.pow(y1, 2);

double D = 2 * (x2 - x3);
double E = 2 * (y2 - y3);
double F = Math.pow(r3, 2) - Math.pow(r2, 2) - Math.pow(x3, 2) +
Math.pow(x2, 2) - Math.pow(y3, 2) + Math.pow(y2, 2);

double x = (C * E - F * B) / (A * E - D * B);
double y = (C * D - A * F) / (B * D - E * A);

// Check if the estimated position is valid
boolean isValid = !Double.isNaN(x) && !Double.isNaN(y);

// Send Super Robot position or an error message to Dr Kamal's computer
String robotPosition;
if (isValid) {
    robotPosition = x + "," + y;
} else {
   robotPosition = "ERROR: Circles do not intersect";
}

InetAddress kamalAddress = InetAddress.getByName("193.188.43.1");
DatagramPacket kamalPacket = new
DatagramPacket(robotPosition.getBytes(), robotPosition.length(),
kamalAddress, KAMAL_PORT);
socket.send(kamalPacket);

// Send Super Robot position or an error message to the LAB's computer
InetAddress labAddress = InetAddress.getByName("193.188.42.1");
DatagramPacket labPacket = new DatagramPacket(robotPosition.getBytes(),
robotPosition.length(), labAddress, LAB_PORT);
socket.send(labPacket);
```

By implementing trilateration, Husam can estimate the position of the Super Robot even when the circles formed by the RSSI measurements do not intersect. The estimated position will be sent to both Dr Kamal's computer and the LAB's computer for further analysis or action.