Peer to Peer File Transfer

UDP Programming

Student ID	Student Name
Yosef Othman Alshamrani	1935445
Mohammed amin waly	1937293

Contents

Code

Main Method

```java
import java.io.*;
import java.net.*;
import java.util.Scanner;
import java.util.logging.*;

public class CPCS371Project {

    //Binary number to add with the message and get the checksum
    static String s = Integer.toBinaryString(53883);//1101001001111011
    //Acknowledgment
    static boolean Ack;

//---------------------------------- main method
    public static void main(String[] args) throws FileNotFoundException, SocketException, IOException {

        //The file which contains the message
        File fileX = new File("UDP Massage.txt");
        //File to write into
        FileWriter Received = new FileWriter("Received.txt");

        //Check first if file exist or not
        if (!fileX.exists()) {
            System.out.println("the file does not exist");
            System.exit(0);
        }//end if

        //Creating the socket in main method so we can make it as a prameter in sender and receiver method
        DatagramSocket socket = new DatagramSocket(2030);

        Scanner file = new Scanner(fileX);
        //loop to read the file line by line and send the message
        while (file.hasNextLine()) {
            String M = file.nextLine();
            StartingPoint(M,socket);
            if(Ack = true){
                Received.write("+Ack: "+M+"\n");
                Received.flush();
            }else{
            Received.write("-Ack: "+M+"\n");
            Received.flush();
            }
        }
```

Starting Method

```java
//---------------------------------- Start meathod
    public static void StartingPoint(String M, DatagramSocket socket) throws SocketException, IOException{

            byte[] data = M.getBytes();
            String temp = convertByteArraysToBinary(data);
            String x = Checksum(temp);
            Sender(M, socket, data, x);
            Reciever(socket, data, x);
            System.out.println();

    }//end Method
```

Sender Method

```java
//---------------------------------- Sender method
    public static void Sender(String m, DatagramSocket socket, byte[] data, String x) throws UnknownHostException, SocketException, IOException {

        //Addres of the Sender
        InetAddress senderAddress = InetAddress.getLocalHost();

        (new Thread() {
            @Override
            public void run() {

                //Packet containes the array of bytes,the address of the sender,and the socket
                DatagramPacket packet = new DatagramPacket(data, data.length, senderAddress, 2030);

                //------ try and catch here is important to avoid errors
                try {

                    //every address must have broadcast
                    socket.setBroadcast(true);

                } catch (SocketException ex) {
                    Logger.getLogger(CPCS371Project.class.getName()).log(Level.SEVERE, null, ex);
                }

                try {

                    //sending the packet through the socket
                    socket.send(packet);

                } catch (IOException ex) {
                    Logger.getLogger(CPCS371Project.class.getName()).log(Level.SEVERE, null, ex);
                }

                try {
                    //wait
                    Thread.sleep(50);

                } catch (InterruptedException ex) {
                    Logger.getLogger(CPCS371Project.class.getName()).log(Level.SEVERE, null, ex);
                }

            }//close run
        }).start();
    }//close method
```

Receiver Method

```java
//---------------------------------- Receiver method
    public static void Reciever(DatagramSocket socket, byte[] data, String x) throws SocketException, IOException {

        (new Thread() {
            @Override
            public void run() {

                //first we create a packet object
                DatagramPacket packet = new DatagramPacket(new byte[data.length], data.length);

                try {

                    //Receive the packet
                    socket.receive(packet);

                    //store the packet into an array of bytes
                    byte[] data2 = packet.getData();
                    String convertesMessage = convertByteArraysToBinary(data2);

                    String checksum = Checksum(convertesMessage);
                    System.out.println(checksum);
                    //-------------------------------------------------------------------------------------------------
                    if (checksum.equals(x)) {
                        Ack = true;
                        System.out.println("Message Recieved, the checksum is matched");
                        System.out.println("Message recieved from the sender: " + new String(packet.getData()));

                    } else {
                        //if there is an error set the ack to false and send the message again
                        Ack = false;
                        System.out.println("Message is not recieved, There is an error");
                        StartingPoint(packet.toString(), socket);
                    }
                } catch (IOException ex) {
                    Logger.getLogger(CPCS371Project.class.getName()).log(Level.SEVERE, null, ex);
                }
            }//close run
        }).start();
    }//close method
```

Convert To binary Method

```java
//---------------------------------- meathod to Convert the message into binary
    public static String convertByteArraysToBinary(byte[] data) {

        StringBuilder result = new StringBuilder();
        for (byte b : data) {
            int val = b;
            for (int i = 0; i < 1; i++) {
                result.append((val & 192) == 0 ? 0 : 1);
                val <<= 1;
            }//End inner loop
        }//End outer loop

        //this for loop and if condition is to make us take only 16 bit of the message
        for (int i = 0; i < result.length(); i++) {
        if (result.length() < 16)
            result.insert(0, "0");
        else if (result.length() > 16)
            result.deleteCharAt(0);
        else
            break;
        }//end loop
        return result.toString();
    }//end meathod
```

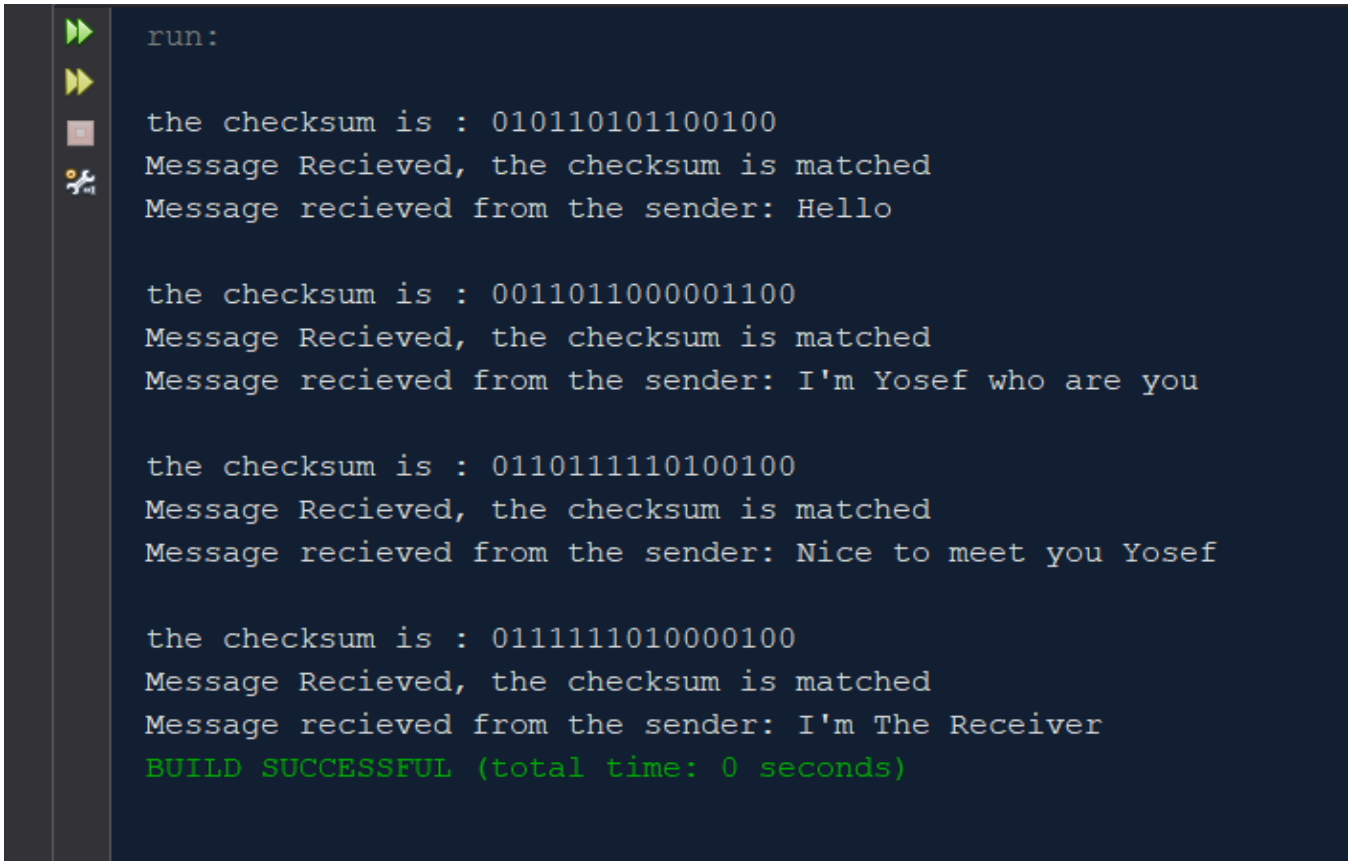
Checksum Method

```
//---------------------------------- meathod to Calculate the cheacksum
    public static String Checksum(String temp) {

        //The two input Strings, containing the binary representation of the Systemvalue and the message value
        //Use as radix 2 because it's binary
        int Systemvalue = Integer.parseInt(s, 2);
        int Takenvalue = Integer.parseInt(temp, 2);

        //adding the two binary numbers as integers
        int sum = Systemvalue+Takenvalue;

        //taking the length of the sum value so if it is exceeded 16 we conclude that there is a carry
        //and we have to do another work which is warp the carry and add it to the right side of the binary number
        int length = String.valueOf(Integer.toBinaryString(sum)).length();

        //after seeing the length we conclude that if it is 6 the number is 17 bits so there is a caary
        if(length > 5){

            //so we add one first to the binary number
            sum = sum + 1;

            //then delete the first bit which we add it to the right side
            String sumAfterDelete = Integer.toBinaryString(sum).substring(1);
            return "the checksum is : " + Onescomplement(sumAfterDelete); //returns the answer as a binary value;
        }

       return "the checksum is : " + Onescomplement(Integer.toBinaryString(sum)); //returns the answer as a binary value;
    }//end meathod
```

Ones complement Method

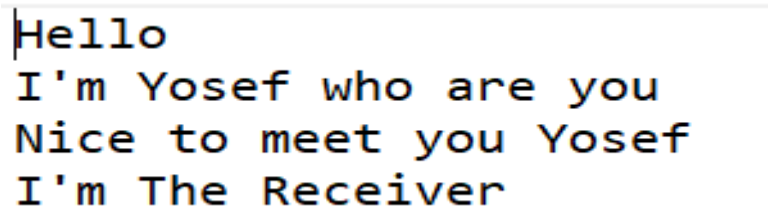
```
//---------------------------------- One's complement Method
    public static StringBuilder Onescomplement(String check){

        StringBuilder newstr = new StringBuilder();
        for (int i = 0; i < check.length(); i++) {
            if(check.charAt(i) == '0')
                newstr.insert(i, "1");
            else
                newstr.insert(i, "0");
        }//end loop
        return newstr;
    }//end method

}// end class
```

Sample Output



Input Output Files

Input File



Output File

```
+Ack: Hello
+Ack: I'm Yosef who are you
+Ack: Nice to meet you Yosef
+Ack: I'm The Receiver
```