進階電腦網路

UPD Socket – 猜數字遊戲

（1A2B）

心得報告

資工二

S0954010 謝宥宣

2022/05/19

目錄

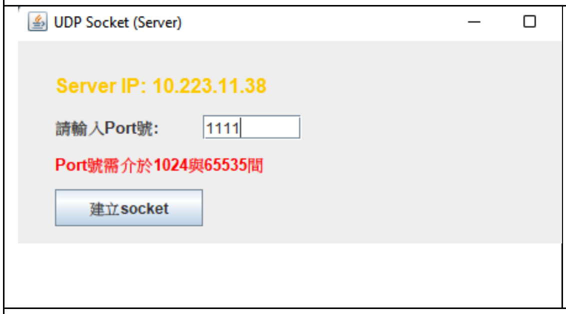
程式功能說明

- 本專題包含兩個程式，一個 Server 程式與一個 Client 程式。
- 本專題程式可透過 UDP 封包傳輸，將兩個不同 IP address 之電腦
- 程式以視窗呈現。
- 操作程式時，需「適時等待」程式運行，為避免 UDP 傳輸封包漏接，傳送與接收間設有 1 秒等待時間。
- 程式設有防呆機制，若任何輸入欄位的輸入值不符要求，將顯示彈出視窗警示。
- 程式輸出結果將存於與 Server 程式同路徑之 record.html 檔案。
- Server 初始化時，會要求使用者輸入 port 號，輸入完成後點擊按鈕設定 socket。
- Client 初始化時，會要求使用者輸入使用者名稱，Server 端 IP address（顯示於 Server 視窗中）和 Server 端 port 號。輸入完成後點擊按鈕設定 socket。
- Server 端初始化後，可設定正確字串之字元數與正確字串。設定完成後，點擊按鈕設定字串與等待接收 Client 端傳送之 Client IP address。
- 若字串欄位輸入＂0＂，則終止程式，關閉 Server 端程式。
- Client 端需等待 Server 端點擊「設定字串」後，方可點擊「傳送本機 IP」，若不依照此順序執行，可能導致程式凍結。
- 前置作業設定完成後，Server 將傳送給 Client 正確字串的字元數。Client 可開始猜測。
- Client 端在按下按鈕前，系統將先檢查字串是否符合字元數與合法字元之規範。
- Client 傳送猜測字串後，Server 將進行運算並回傳 A, B 數值給 Client，Client 接收後將顯示於畫面中。
- 每兩次傳輸終須間格至少 1 秒鐘。
- 若 Client 猜測正確，則 Server 端將顯示 Client 的猜測紀錄與本次猜測資訊（名稱、日期、猜測次數與總費時等），並讀取檔案，運算名次後，回傳名字給 Client。Client 收到後將顯示於畫面上。

- Server 亦會將本次遊玩資訊寫入檔案中，並加粗顯示。
- 遊戲結束後，Server 端將出現接收按鈕，點擊後即可接收來自 Client 之指示（再次遊玩或結束程式）。
- 若 Client 選擇再次遊玩，Server 端將開放設定字串字元數與字串內容，Client 端將開放設定名稱，設定完成後與上述初始化過程相同。
- 若 Client 選擇結束程式，Server 端與 Client 端皆將出現警示畫面後關閉程式。

程式操作流程

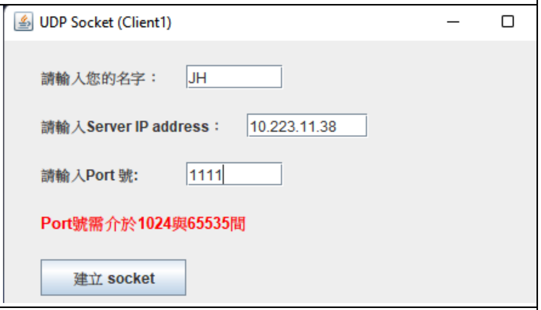
初始化流程：分別設定資訊後，按下「建立 socket」


UDP Socket (Server)

Server IP: 10.223.11.38

請輸入Port號: 1111

Port號需介於1024與65535間


UDP Socket (Client1)

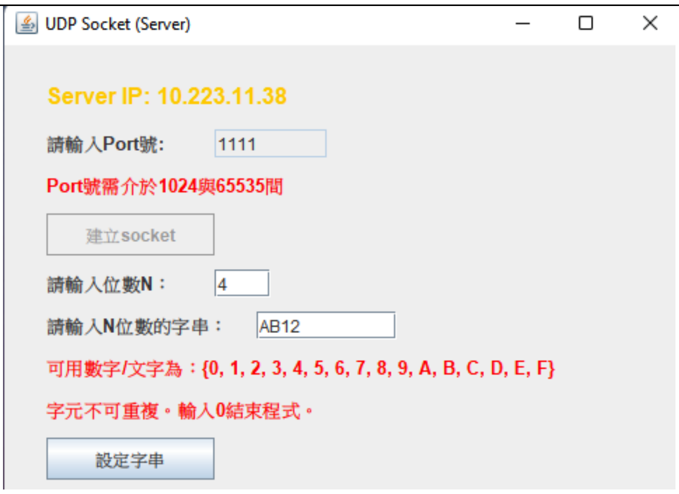
請輸入您的名字： JH

請輸入Server IP address： 10.223.11.38

請輸入Port 號: 1111

Port號需介於1024與65535間

Server 端設定位數 N 與正確字串後點擊「設定字串」。


UDP Socket (Server)

Server IP: 10.223.11.38

請輸入Port號: 1111

Port號需介於1024與65535間

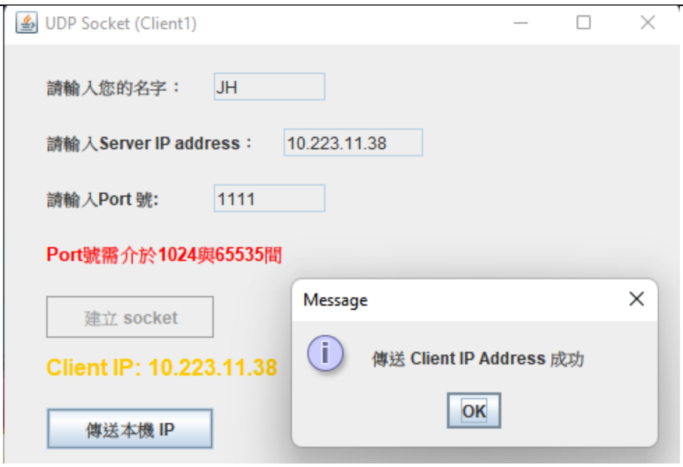
請輸入位數N： 4

請輸入N位數的字串： AB12

可用數字/文字為：{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

字元不可重複。輸入0結束程式。

Server 端點擊「設定字串」後，Client 端可點擊「傳送本地 IP」。


UDP Socket (Client1)

請輸入您的名字： JH

請輸入Server IP address： 10.223.11.38

請輸入Port 號: 1111

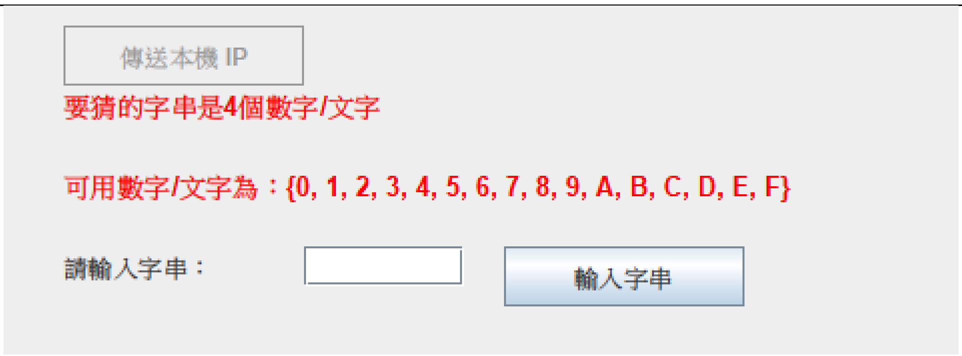
Port號需介於1024與65535間

Client IP: 10.223.11.38

Message

傳送 Client IP Address 成功

若資訊接收正確，則 Client 端將顯示正確答案之字元數，並可開始遊戲。

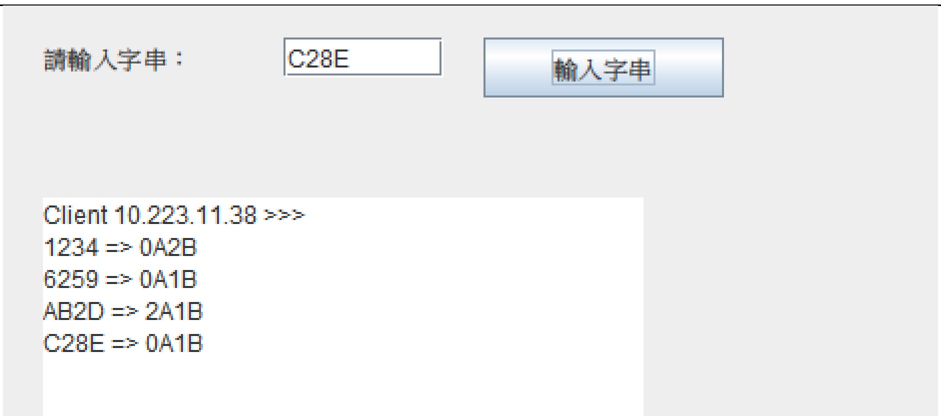


要猜的字串是4個數字/文字

可用數字/文字為：{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

請輸入字串：

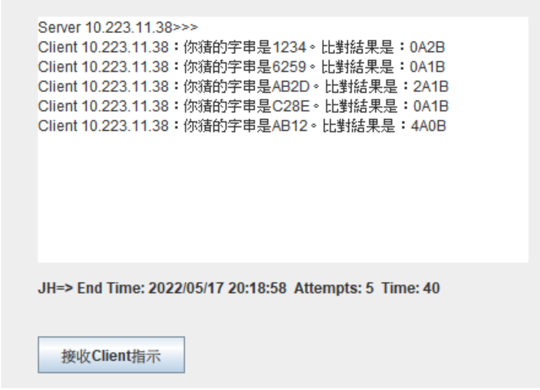
若猜測字串不為正確字串，將顯示 A, B 數值。



請輸入字串：C28E

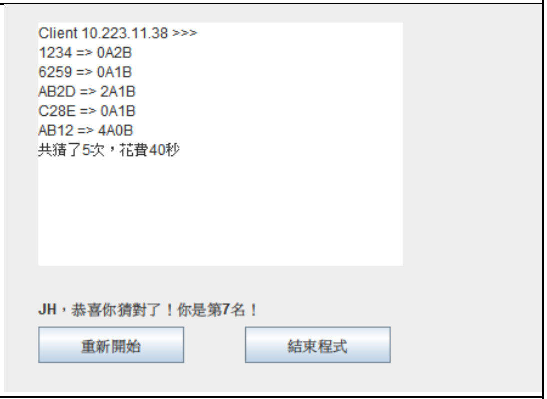
```
Client 10.223.11.38 >>>
1234 => 0A2B
6259 => 0A1B
AB2D => 2A1B
C28E => 0A1B
```

若猜測為正確字串，Server 端將顯示猜測紀錄與本次遊玩資訊。Client 端將於最下方顯示本次遊玩時間的名次。



```
Server 10.223.11.38>>>
Client 10.223.11.38：你猜的字串是1234。比對結果是：0A2B
Client 10.223.11.38：你猜的字串是6259。比對結果是：0A1B
Client 10.223.11.38：你猜的字串是AB2D。比對結果是：2A1B
Client 10.223.11.38：你猜的字串是C28E。比對結果是：0A1B
Client 10.223.11.38：你猜的字串是AB12。比對結果是：4A0B
```

JH=> End Time: 2022/05/17 20:18:58 Attempts: 5 Time: 40



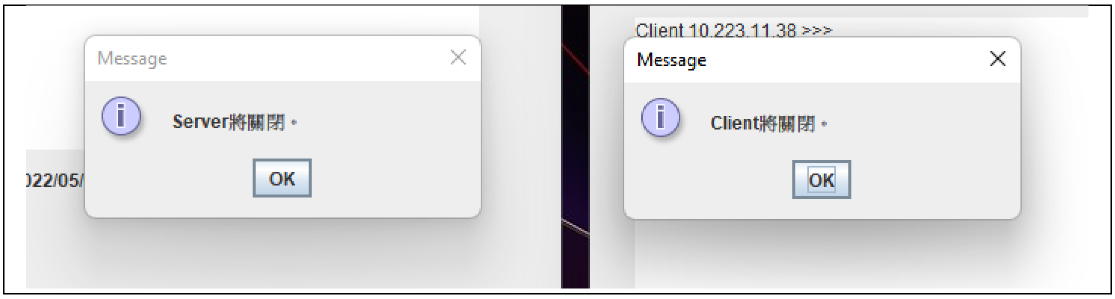
```
Client 10.223.11.38 >>>
1234 => 0A2B
6259 => 0A1B
AB2D => 2A1B
C28E => 0A1B
AB12 => 4A0B
共猜了5次，花費40秒
```

JH，恭喜你猜對了！你是第7名！

Server 端按下「接收 Client 指示」後，Client 端可選擇「重新開始」或「結束程式」。

若選擇結束程式，則在下方訊息出現後關閉程式。

若選擇重新開始，則將會開放重設字串資訊與使用者名稱。



)22/05/

Message

Server將關閉。

Client 10.223.11.38 >>>

Message

Client將關閉。

程式碼註解

Server 程式碼

```
import javax.swing.JFrame;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.net.*;
import java.util.concurrent.TimeUnit;
import java.io.*;

public class socket_Server extends JFrame {

    static JButton create = new JButton("建立 socket"); //global elements
    static JTextField myportnum = new JTextField();
    static JLabel length = new JLabel("請輸入位數 N：");
    static JTextField mylength = new JTextField();
    static JLabel str = new JLabel("請輸入 N 位數的字串：");
    static JTextField mystr = new JTextField();
    static JLabel hint2 = new JLabel("可用數字/文字為：{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}");
    static JLabel hint21 = new JLabel("字元不可重複。輸入 0 結束程式。");
    static JButton set = new JButton("設定字串");
    static InetAddress addr;
    static String client;
    static JTextArea record = new JTextArea();
    static int portnum;
    static String ans = "";
    static String toprint = "";
    static JLabel result = new JLabel();
    static JLabel test1 = new JLabel();
static JLabel test2 = new JLabel();
static JButton wait = new JButton("接收 Client 指示");

public static void main(String[] args) throws Exception {
    JFrame frm = new JFrame("UDP Socket (Server)");
    frm.setLayout(null);
    Container ctp = frm.getContentPane();
    ctp.setLayout(null);

    JLabel myIP = new JLabel("Server IP: " + getaddress());
    myIP.setBounds(30, 20, 200, 30);
    myIP.setFont((new Font("Serief", Font.BOLD, 16)));
    myIP.setForeground(Color.ORANGE);
    ctp.add(myIP);

    JLabel portnum = new JLabel("請輸入 Port 號:");
    portnum.setFont((new Font("Serief", Font.BOLD, 13)));
    JLabel hint = new JLabel("Port 號需介於 1024 與 65535 間");
    hint.setFont((new Font("Serief", Font.BOLD, 13)));
    hint.setForeground(Color.RED);
    ctp.add(portnum);
    ctp.add(myportnum);
    ctp.add(hint);
    portnum.setBounds(30, 60, 100, 20);
    myportnum.setBounds(150, 60, 80, 20);
    hint.setBounds(30, 90, 200, 20);
    create.addActionListener(new Actlis());

    ctp.add(create);
    create.setBounds(30, 120, 120, 30);

    length.setFont(((new Font("Serief", Font.BOLD, 13))));
    str.setFont(((new Font("Serief", Font.BOLD, 13))));
```

```
length.setVisible(false);
mylength.setVisible(false);
str.setVisible(false);
mystr.setVisible(false);
hint2.setVisible(false);
hint21.setVisible(false);

ctp.add(length);
ctp.add(mylength);
ctp.add(str);
ctp.add(mystr);
ctp.add(hint2);
ctp.add(hint21);
length.setBounds(30, 160, 100, 20);
mylength.setBounds(150, 160, 40, 20);
str.setBounds(30, 190, 140, 20);
mystr.setBounds(180, 190, 100, 20);
hint2.setBounds(30, 220, 400, 20);
hint21.setBounds(30, 250, 400, 20);
hint2.setFont((new Font("Serief", Font.BOLD, 13)));
hint2.setForeground(Color.RED);
hint21.setFont((new Font("Serief", Font.BOLD, 13)));
hint21.setForeground(Color.RED);

ctp.add(set);
set.setBounds(30, 280, 120, 30);
set.setVisible(false);
set.addActionListener(new Actlis());
record.setBounds(30, 340, 400, 200);
record.setEditable(false);
ctp.add(record);
result.setBounds(30, 550, 400, 20);
ctp.add(result);
```

```
        ctp.add(test1);

        ctp.add(test2);

        test1.setBounds(300, 60, 100, 20);

        test2.setBounds(150, 100, 400, 20);

        ctp.add(wait);

        wait.setBounds(30, 600, 120, 30);

        wait.setVisible(false);

        wait.addActionListener(new Actlis());

        frm.setSize(500, 800);


        // frm.setLocation(800, 150);

        frm.setVisible(true);

        frm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    }


    public static String getaddress() throws Exception { // function to get Server IPv4.

        InetAddress myIP = InetAddress.getLocalHost();

        String myIPv4 = myIP.getHostAddress(); // Delete hostname, keeping only IPv4.

        return myIPv4;

    }

    static public class Actlis extends WindowAdapter implements ActionListener // .addActionListener(new
Actlis());

    {

        public void windowClosing(WindowEvent e) {

            System.exit(0);

        }

        public void actionPerformed(ActionEvent e) {

            if (e.getSource() == create) {

                Boolean fail = false;

                JFrame msg = new JFrame();

                if (myportnum.getText().equals("")) { //port num not input

                    JOptionPane.showMessageDialog(msg, "請輸入 Port 號。");

                    fail = true;
                } else {
                    portnum = Integer.parseInt(myportnum.getText());
                    if (portnum >= 1024 && portnum <= 65535) { //check if port num is between 1024 and 65535
                        myportnum.setEditable(false);
                    } else {    //if not, show error message
                        JOptionPane.showMessageDialog(msg, "非法 Port 號，請重新輸入。");
                        fail = true;
                    }
                }
                if (fail == false)
                    try {
                        DatagramSocket server = new DatagramSocket(portnum); //try create socket
                        server.close();
                        JOptionPane.showMessageDialog(msg, "Socket 設定成功！");
                        length.setVisible(true); //disable current step column and buttons, also enable next step elements
                        mylength.setVisible(true);
                        str.setVisible(true);
                        mystr.setVisible(true);
                        hint2.setVisible(true);
                        hint21.setVisible(true);
                        set.setVisible(true);
                        create.setEnabled(false);
                    } catch (Exception ex) {
                    }
            }
            if (e.getSource() == set) {
                int n = 0;
                Boolean fail = false;
                JFrame msg = new JFrame();
                if (mystr.getText().equals("0")) { //if mystr column inputs 0, terminate server
                    JOptionPane.showMessageDialog(msg, "Server 端將關閉");
                    System.exit(0);
```

```
}
if (mylength.getText().equals("")) { //if mylength column not entered,
    JOptionPane.showMessageDialog(msg, "請輸入字串長度 N。");
    fail = true;
} else {
    try {
        n = Integer.parseInt(mylength.getText());

    } catch (Exception ex) { //if legal number not entered
        JOptionPane.showMessageDialog(msg, "請輸入合法數字。");
        fail = true;
    }
}
if (fail == false) { // if above conditions meet
    String mystr1 = "";
    if (mystr.getText().equals("")) {
        JOptionPane.showMessageDialog(msg, "請輸入字串。");
        fail = true;
    } else {
        mystr1 = mystr.getText();
        if (mystr.getText().length() != n) { //if length of string not equal to n
            JOptionPane.showMessageDialog(msg, "請輸入長度" + n + "的字串。");
            fail = true;
        } else {
            for (int i = 0; i < mystr1.length(); i++) {
                if (mystr1.charAt(i) < '0' || mystr1.charAt(i) > '0') { //check if inputs are all legal chars
                    if (mystr1.charAt(i) > 'F') {
                        JOptionPane.showMessageDialog(msg, "請輸入合法字串。");
                        fail = true;
                    }
                }
            }
        }
    }
    if (fail == false) // check duplicated char
    {
        for (int i = 0; i < mystr1.length(); i++) {
            for (int j = i + 1; j < mystr1.length(); j++) {
                if (mystr1.charAt(i) == mystr1.charAt(j)) {
                    JOptionPane.showMessageDialog(msg, "請輸入不重複的字串。");
                    fail = true;
                }
            }
        }
    }
    if (fail == false) { //if conditions above meet
        JOptionPane.showMessageDialog(msg, "字串設定成功！");
        ans = mystr1;
        mylength.setEditable(false); //disable previous step elements
        mystr.setEditable(false);
        set.setEnabled(false);
        try {
            client = rcvmsg(); //receive string of client's IP address from client.
            //test1.setText(client);
            addr = InetAddress.getByName(client); //set client's IP address as the destination when 
server sends.
            // record.setText("Client IP =" + client);
            TimeUnit.SECONDS.sleep(1); //wait for 1 sec, prevent client not receiving packet
            sendmsg(mylength.getText()); //send to client length
            toprint += "Server " + getaddress() + ">>>\n"; //initialize string to print on server

            while (true) {
                String guess = rcvmsg(); //receive cluent's guess and save it to guess
                // record.setText(guess);
                int a = 0;
                int b = 0;
                for (int i = 0; i < guess.length(); i++) { //loop to calculate a,c values
                    for (int j = 0; j < guess.length(); j++) {
                        if (guess.charAt(i) == ans.charAt(j)) {
                            if (i == j) {
                                a++;
                            } else {
                                b++;
                            }
                        }
                    }
                }
                toprint += "Client " + client + "：你猜的字串是" + guess + "。比對結果是：" + a + "A" + b + "B\n"; //add result to string to print
                TimeUnit.SECONDS.sleep(1); //pause 1 sec before sending AB result to client.
                String tosend = a + "A" + b + "B";
                sendmsg(tosend); //send AB results to client.
                if (a == guess.length()) { //if client's guess is correct
                    record.setText(toprint); //print the records
                    break;
                }
            }
            ///// files
            String arr[] = rcvmsg().split(","); //receive result messages and split it with "," (client send information with , between)
            String name = arr[0];
            String endtime = arr[1];
            String count = arr[2];
            String totaltime = arr[3];
            String toprint1 = name + "=> End Time: " + endtime + "  Attempts: " + count +
                    "  Time: "
                    + totaltime; //set format to print
            toprint += toprint1;
            result.setText(toprint1);
            result.setVisible(true);
            //// Write into file
            int score = 0;
            File file = new File("record.html");
            file.createNewFile(); //if "record.html" not found, create new file; do nothing otherwise

            FileReader fr = new FileReader("record.html");
            BufferedReader bfr = new BufferedReader(fr);
            String str = "";
            String rec[] = new String[100];
            int cnt = 0;
            // test1.setText("Test1");
            while ((str = bfr.readLine()) != null) { //remove tags from old format and replace "Time: " with "," for further use.
                str = str.replaceAll("<b>", "");
                str = str.replaceAll("</b>", "");
                str = str.replaceAll("<br>", "");
                str = str.replaceAll("Time: ", ",");
                // test2.setText(str);
                // System.out.println(String.valueOf(str));
                rec[cnt] = str;
                cnt++; //count the total amount of records
            }

            fr.close();
            FileWriter fw = new FileWriter("record.html");
            BufferedWriter bfw = new BufferedWriter(fw);
            Boolean write = false;
            // test2.setText(str);
            try {
                for (int i = 0; i < cnt; i++) { //start from shortest time to longest time, if current time < record time, write in current, else write original record
                    String resulttime[] = rec[i].split(","); //split the results with ",", get the 3 (resulttime[2]) to get time from record.
```

```
                  // 1result.setText(String.valueOf(resulttime[2]));


                  if ((write == false)
                      && (Integer.valueOf(resulttime[2]) >= Integer.valueOf(totaltime))) { //current
time < record time
                    // test2.setText("test2");
                    score = i + 1;
                    bfw.write("<b>" + toprint1 + "</b><br>"); //bold current record
                    bfw.newLine();
                    write = true;
                    // test2.setText("true");
                  }
                  rec[i] = rec[i].replaceAll(",", "Time: "); //restore record
                  rec[i] = rec[i] + "<br>";
                  bfw.write(rec[i]);
                  bfw.newLine();
                }
              } catch (Exception except) {
                JOptionPane.showMessageDialog(msg, "處理資料錯誤!!");
              }


              if (write == false) // if current is last or first
              {
                if (cnt != 0)
                  score = cnt + 1;
                else
                  score = 1;
                bfw.write("<b>" + toprint1 + "</b><br>");
                bfw.newLine();
              }
              bfw.flush();
              fw.close();
              TimeUnit.SECONDS.sleep(1); //pause 1 sec before sending
```

```
                        sendmsg(String.valueOf(score));

                        // TimeUnit.SECONDS.sleep(1);
                        wait.setVisible(true); // enable next stage element

                    } catch (Exception ee) {
                        JOptionPane.showMessageDialog(msg, "Server端操作失敗，請稍後再試。");
                    }
                }
            }
        }
        if (e.getSource() == wait) { //set a button to prevent time collapse
            JFrame msg = new JFrame();
            String rst = rcvmsg(); //receive decision made from client
            wait.setVisible(false);
            if (rst.equals("0")) { //terminate program
                JOptionPane.showMessageDialog(msg, "Server將關閉。");
                System.exit(0);
            } else if (rst.equals("1")) { //restart program
                mylength.setEditable(true);
                mystr.setEditable(true);
                set.setEnabled(true);
                toprint = "";
                record.setText("");
                result.setText("");
            }
        }
    }
}

public static String rcvmsg() { //function to receive message
    JFrame msg = new JFrame();
    String clientmsg = "";
```

```
        try {
            byte buffer[] = new byte[100];
            DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
            DatagramSocket socket = new DatagramSocket(portnum);
            socket.setSoTimeout(30000); // timeout 30s
            socket.receive(packet);
            clientmsg = new String(buffer, 0, packet.getLength());
            // System.out.println(clientmsg + "test");
            socket.close();

        } catch (SocketTimeoutException timeout) {
            JOptionPane.showMessageDialog(msg, "Timeout！");
            System.exit(0);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(msg, "Server 發生接收錯誤，請稍後再試！");
        }
        return clientmsg;
    }
    // 192.168.0.83
    public static void sendmsg(String str) { //function to send message
        JFrame msg = new JFrame();
        int oLength = str.length(); // 將訊息長度大小放到 oLength
        byte buffer[] = new byte[oLength];
        buffer = str.getBytes();
        DatagramPacket packet = new DatagramPacket(buffer, oLength, addr, portnum);
        try {
            for (int i = 0; i < 1500; i++) {
                DatagramSocket socket = new DatagramSocket(); // 建立 socket
                socket.send(packet); // 送出 packet 封包訊息
                socket.close();
            }
        } catch (Exception e) {
            JOptionPane.showMessageDialog(msg, "Server 發生傳送錯誤，請稍後再試！");
            System.exit(0);
        }
    }
}
```

Client 程式碼

```
import javax.swing.JFrame;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.net.*;
import java.net.DatagramPacket;
import java.util.concurrent.TimeUnit;
import java.time.format.DateTimeFormatter;
import java.time.LocalDateTime;

public class socket_Client extends JFrame {

    static JButton create = new JButton("建立 socket"); //global elements
    static JTextField myname = new JTextField();
    static JTextField myip = new JTextField();
    static JTextField myport = new JTextField();
    static InetAddress addr;
    static JButton sendip = new JButton("傳送本機 IP");
    static JTextArea play = new JTextArea();
    static JLabel clientip = new JLabel();
    static JLabel rulefromserver = new JLabel();
    static JLabel rule1 = new JLabel("可用數字/文字為：{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}");
    static String serverip = "";
    static int portnumber;
    static JTextField myguess = new JTextField();
    static JLabel guess = new JLabel("請輸入字串：");
    static JButton sendguess = new JButton("輸入字串");
    static int length; // n
    static String toprint = "";
    static int count = 0;
    static long time = 0;
    static long endtime = 0;
    static JLabel finals = new JLabel();
    static JButton restart = new JButton("重新開始");
    static JButton endgame = new JButton("結束程式");

    public static void main(String[] args) throws Exception {
        JFrame frm = new JFrame("UDP Socket (Client1)");
        frm.setLayout(null);
        Container ctp = frm.getContentPane();
        ctp.setLayout(null);

        JLabel name = new JLabel("請輸入您的名字：");
        JLabel ip = new JLabel("請輸入Server IP address：");
        JLabel port = new JLabel("請輸入Port 號: ");
        JLabel hint = new JLabel("Port 號需介於1024 與65535 間");
        hint.setFont((new Font("Serief", Font.BOLD, 13)));
        hint.setForeground(Color.RED);
        name.setBounds(30, 20, 100, 20);
        myname.setBounds(150, 20, 80, 20);
        ip.setBounds(30, 60, 200, 20);
        myip.setBounds(200, 60, 100, 20);
        port.setBounds(30, 100, 100, 20);
        myport.setBounds(150, 100, 80, 20);
        hint.setBounds(30, 140, 200, 20);

        create.setBounds(30, 180, 120, 30);
        create.addActionListener(new Actlis());
        clientip.setBounds(30, 220, 200, 20);
        clientip.setFont((new Font("Serief", Font.BOLD, 16)));
        clientip.setForeground(Color.ORANGE);
        clientip.setText("Client IP: " + getaddress());
```

```
sendip.setEnabled(false);

sendip.setBounds(30, 260, 120, 30);

sendip.addActionListener(new Actlis());

clientip.setVisible(false);

sendip.setVisible(false);

ctp.add(name);

ctp.add(myname);

ctp.add(ip);

ctp.add(myip);

ctp.add(port);

ctp.add(myport);

ctp.add(hint);

ctp.add(create);

ctp.add(clientip);

ctp.add(sendip);

ctp.add(play);

ctp.add(rulefromserver);

ctp.add(rule1);

rule1.setVisible(false);

play.setEditable(false);

play.setBounds(30, 450, 300, 200);

rulefromserver.setBounds(30, 290, 400, 20);

rule1.setBounds(30, 330, 400, 20);

rulefromserver.setFont((new Font("Serief", Font.BOLD, 13)));

rulefromserver.setForeground(Color.RED);

rule1.setFont((new Font("Serief", Font.BOLD, 13)));

rule1.setForeground(Color.RED);

ctp.add(guess);

guess.setBounds(30, 370, 80, 20);

guess.setVisible(false);

ctp.add(myguess);

myguess.setBounds(150, 370, 80, 20);

myguess.setVisible(false);
```

```
        ctp.add(sendguess);
        sendguess.setBounds(250, 370, 120, 30);
        sendguess.setVisible(false);
        sendguess.addActionListener(new Actlis());
        ctp.add(finals);
        finals.setBounds(30, 670, 300, 30);
        ctp.add(restart);
        ctp.add(endgame);
        restart.setBounds(30, 700, 120, 30);
        endgame.setBounds(200, 700, 120, 30);
        restart.setVisible(false);
        restart.addActionListener(new Actlis());
        endgame.setVisible(false);
        endgame.addActionListener(new Actlis());
        frm.setSize(500, 800);
        // frm.setLocation(800, 150);
        frm.setVisible(true);
        frm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    static public class Actlis extends WindowAdapter implements ActionListener // .addActionListener(new
Actlis());
    {
        public void windowClosing(WindowEvent e) {
            System.exit(0);
        }

        public void actionPerformed(ActionEvent e) {
            if (e.getSource() == create) {
                JFrame msg = new JFrame();
                Boolean fail = false;
                if (myname.getText().equals("")) { // name not input
                    fail = true;
                    JOptionPane.showMessageDialog(msg, "請輸入名字。");
```

```
}
if (myip.getText().equals("")) { //IP address not input
    fail = true;
    JOptionPane.showMessageDialog(msg, "請輸入 IP address。");
}
if (myport.getText().equals("")) { //port num not input
    fail = true;
    JOptionPane.showMessageDialog(msg, "請輸入 Port 號。");
} else {
    int portnum = Integer.parseInt(myport.getText());

    if (portnum >= 1024 && portnum <= 65535) { //check if port num is between 1024 and 65535
        JOptionPane.showMessageDialog(msg, "Socket 設定成功！");
        myname.setEditable(false);
        myip.setEditable(false);
        myport.setEditable(false);
        sendip.setEnabled(true);
        sendip.setVisible(true);
        clientip.setVisible(true);
        serverip = myip.getText();
        portnumber = Integer.parseInt(myport.getText());

        try { //try to create socket
            addr = InetAddress.getByName(serverip);
            DatagramSocket clientSocket = new DatagramSocket();
            clientSocket.close();
        } catch (Exception ee) {
            JOptionPane.showMessageDialog(msg, "Fail to create socket.");
        }

    } else { //illegal port num input
        JOptionPane.showMessageDialog(msg, "非法 Port 號，請重新輸入。");
        fail = true;
```

```
            }
        }
        if (fail == false)
            try {

                DatagramSocket client = new DatagramSocket(); // try to create socket
                client.close();
                create.setEnabled(false); //disable current step column and buttons
                myname.setEditable(false);
                myip.setEditable(false);
                myport.setEditable(false);
            } catch (Exception ex) {
                JOptionPane.showMessageDialog(msg, "Fail to create socket.");
            }

    }
    if (e.getSource() == sendip) {
        JFrame msg = new JFrame();

        try {
            String clientIP = getaddress(); //get client IP address
            sendmsg(clientIP); //send client IP address to server
            JOptionPane.showMessageDialog(msg, "傳送 Client IP Address 成功");
            sendip.setEnabled(false);
            // play.setText(rcvmsg());
            //TimeUnit.SECONDS.sleep(1); //
            length = Integer.valueOf(rcvmsg()); //receive string length from server
            String txt = "要猜的字串是" + length + "個數字/文字";
            rulefromserver.setText(txt);
            rulefromserver.setVisible(true);
            rule1.setVisible(true);
            guess.setText("請輸入字串：");
            guess.setVisible(true);
            myguess.setVisible(true);
            sendguess.setVisible(true);
            toprint += "Client " + clientIP + " >>>\n";
            ///////////////////////////////////////////////////////
        } catch (Exception ex) {
            JOptionPane.showMessageDialog(msg, "操作失敗，請稍後再試。");
        }
    }

    if (e.getSource() == sendguess) { //if sendguess button clicked
        if (count == 0) //initialize, first send starts timer
            time = System.currentTimeMillis();
        Boolean fail = false;
        JFrame msg = new JFrame();
        String str = myguess.getText();
        if (str.length() != length) { //check if string length = n
            JOptionPane.showMessageDialog(msg, "請輸入長度" + length + "的字串。");
            fail = true;
        } else { //check for illegal char
            for (int i = 0; i < str.length(); i++) {
                if (str.charAt(i) < '0' || str.charAt(i) > '0') {
                    if (str.charAt(i) > 'F') {
                        JOptionPane.showMessageDialog(msg, "請輸入合法字串。");
                        fail = true;
                    }
                }
            }
        }
        if (fail == false) // check duplicated char
        {
            for (int i = 0; i < str.length(); i++) {
                for (int j = i + 1; j < str.length(); j++) {
                    if (str.charAt(i) == str.charAt(j)) {
```

```
                JOptionPane.showMessageDialog(msg, "請輸入不重複的字串。");
                fail = true;
            }
        }
    }
}
if (fail == false) { //if above conditions meet
    try {
        sendmsg(str); // send guess string
        // JOptionPane.showMessageDialog(msg, "字串傳送成功！");
        TimeUnit.SECONDS.sleep(1); //pause for 1 sec
        count++; // count +1 for every guess
        String result = rcvmsg(); //receive result of guess from server
        toprint += str + " => " + result + "\n";
        play.setText(toprint);
        if (Character.getNumericValue(result.charAt(0)) == str.length()) { //if guess right
            sendguess.setEnabled(false);
            endtime = System.currentTimeMillis(); //get current time to calculate time spent
            long totaltime = endtime - time;
            totaltime /= 1000; //convert from millisec to sec
            toprint += "共猜了" + count + "次，花費" + totaltime + "秒";
            play.setText(toprint);
            DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/dd HH:mm:ss");
            LocalDateTime now = LocalDateTime.now(); // get local date and time
            String nowtime = dtf.format(now);
            String tosend = myname.getText() + "," + nowtime + "," + count + "," + totaltime; //send
information, using "," to split
            TimeUnit.SECONDS.sleep(1); //pause 1 sec
            sendmsg(tosend); //send information to server
            //TimeUnit.SECONDS.sleep(1);
            String score = rcvmsg(); //receive ranking from server
            String finalscore = myname.getText() + "，恭喜你猜對了！你是第" + score + "名！";
            finals.setText(finalscore);
```

```
              restart.setVisible(true); //enable next stage buttons
              endgame.setVisible(true);
            }
          } catch (Exception ee) {
            JOptionPane.showMessageDialog(msg, "字串傳送失敗，請稍後再試。");
          }
        }
      }
      if(e.getSource() == restart) //if restart is clicked
      {
        toprint = ""; //set step 1 elements to correct status
        guess.setText("");
        myname.setEditable(true);
        create.setEnabled(true);
        myip.setVisible(false);
        sendip.setVisible(false);
        rule1.setVisible(false);
        rulefromserver.setVisible(false);
        play.setText("");
        guess.setVisible(false);
        myguess.setEditable(true);
        sendguess.setEnabled(true);
        myguess.setVisible(false);
        sendguess.setVisible(false);
        finals.setText("");
        restart.setVisible(false);
        endgame.setVisible(false);
        count = 0;
        try {
          // TimeUnit.SECONDS.sleep(1);
           sendmsg("1"); //send decision to server
        } catch (Exception eq) {
           //TODO: handle exception
```

```
            }
        }
        if(e.getSource() == endgame) //if endgame is clicked
        {
            JFrame msg = new JFrame();
            try {
                //TimeUnit.SECONDS.sleep(1);
                sendmsg("0"); //send decision to server
                JOptionPane.showMessageDialog(msg, "Client 將關閉。");
                System.exit(0); //terminate client
            } catch (Exception eq) {
                //TODO: handle exception
            }
        }
    }
}

public static String getaddress() throws Exception { // function to get Client IPv4.
    InetAddress myIP = InetAddress.getLocalHost();
    String myIPv4 = myIP.getHostAddress(); // Delete hostname, keeping only IPv4.
    return myIPv4;
}

public static String rcvmsg() { //function to receive message from server
    JFrame msg = new JFrame();
    String servermsg = "";
    try {
        byte buffer[] = new byte[100];
        DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
        DatagramSocket socket = new DatagramSocket(portnumber);
        socket.receive(packet);
        servermsg = new String(buffer, 0, packet.getLength());
        socket.close();
        } catch (SocketTimeoutException timeout) {

            JOptionPane.showMessageDialog(msg, "Timeout！");

            System.exit(0);


        } catch (Exception e) {

            JOptionPane.showMessageDialog(msg, "Client 發生接收錯誤，請稍後再試！");

        }

        return servermsg;

    }


    public static void sendmsg(String str) { //function to send message

        JFrame msg = new JFrame();

        int oLength = str.length(); // 將訊息長度大小放到 oLength

        byte buffer[] = new byte[oLength];

        buffer = str.getBytes();

        DatagramPacket packet = new DatagramPacket(buffer, oLength, addr,
Integer.parseInt(myport.getText()));

        try {

            for (int i = 0; i < 1500; i++) {

                DatagramSocket socket = new DatagramSocket(); // 建立 socket

                socket.send(packet); // 送出 packet 封包訊息

                socket.close();

            }

        } catch (Exception e) {

            JOptionPane.showMessageDialog(msg, "Client 發生傳送錯誤，請稍後再試！");

        }

    }

}
```

問題、討論與心得

問題與討論

1. 問題：測試同裝置間的 datagram 傳輸時，發現可能會發生兩者程式不同步的情況，導致一個程式開始接收時，另一程式尚未開始傳的問題。

   解決辦法：為避免程式因為收到 datagram 而凍結，在 Client 端加上 socket.setSoTimeout(30000); ，使其在 30 秒內沒收到 datagram 自動關閉。另一解決方法為：適度以使用者點擊 button 才觸發的方式來同步，可避免程式因執行速度不同而產生時差。

2. 問題：輸出檔案無法顯示字體效果？

   解決辦法：數次嘗試以<b></b>框出期望有粗體字效果的文字，但皆無法在.txt 檔案中實現。經詢問學長，學長提示可用.html 檔案代替，即可產出期望中的文字效果。

3. 問題：UDP 傳輸有一定機率收不到，常常導致程式出狀況。

   解決辦法：可使 UDP 傳輸時嘗試一次性傳送多個重複訊息給對方，可提升訊息傳遞正確性。因此，在本程式中，每次執行 sendmsg() function 時皆會重複傳送 1500 次。

4. 問題：在同裝置（同 IP address）內可正常運作傳輸，但若在兩部裝置間傳輸，卻會收不到對方的 datagram？

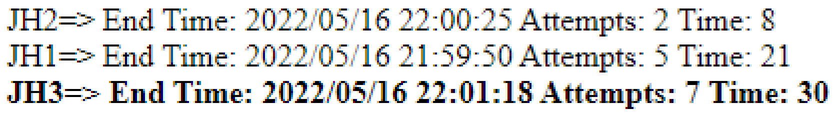
   解決辦法：經過請教學長與查詢網路資訊，最可能的原因為：電腦的防火牆或路由器擋住 UDP 的 datagram，因此，關閉兩台電腦的防火牆後，我改為都連接手機分享的 Wi-Fi 後，即可正常收發與顯示訊息。

心得

這次的作業可能是從進到資工系以來，單一程式寫過最多行的作業。雖然乍看之下作業要求簡單，但因為需顧慮的點過多，如時間順序，接收成功與否，是否有 blocking call 等問題，真正要實現起來十分困難。另外，當初我會決定要用視窗化來呈現的原因是因為教授上課時提到，希望繳交作業時能以執行檔方式繳交，但我不太清楚若以 jar 檔繳交，要如何顯示於 console 中，因此我為了能有一個簡易的文字框，才下定決心要以視窗呈現。雖然視窗化呈現要寫的程式比一般程式要多許多，要調整的參數也較多，但我認為視窗化呈現反倒能讓我清楚知道目前程式執行的進度，能自訂的元件也較多。另外，此次作業的繳交時機剛好也是另一堂課的專題繳交周以及再另一堂課的小考周，所有事情同時一窩蜂出現，真的令我有些措手不及。幸好學長有提供非常有幫助的協助與建議，才讓我得以在期限內完成。若不是學長的幫助，我的程式可能還卡在路上。真的非常謝謝學長！這次的作業，讓我得以第一次將從課堂上學習到的知識實現，寫完後真的非常有成就感！

附錄

一、 遊玩紀錄輸出檔案內容截圖（record.html）



```
JH2=> End Time: 2022/05/16 22:00:25 Attempts: 2 Time: 8
JH1=> End Time: 2022/05/16 21:59:50 Attempts: 5 Time: 21
JH3=> End Time: 2022/05/16 22:01:18 Attempts: 7 Time: 30
```

二、 程式執行過程 demo 影片

網址：https://drive.google.com/file/d/1IcLj793ZnnhI-TFhkUizVRGqDzYjL1Iw/view?usp=sharing