AĞ PROGRAMLAMA PROJE1: Yakar Top Oyunu

Bu Projede oyun kurucu ve yöneticisi olacak merkezi sunucu ile, buna bağlı oyuncuların bulunduğu bir sistem tasarlanacaktır.. Sistemde tek bir merkezi sunucu bulunacak, birden fazla sayıda oyun aynı anda eş zamanlı olarak kurulup oynanabilecektir. Tüm oyunların kurulması ve yönetimi tek sunucu üzerinden yapılacaktır. Tüm oyuncular bir takma ad ve şifre ile önce sunucuya bağlanacak sonrasında sunucuda yeni oyun başlatma talebi oluşturacak veya var olan oyunlardan birine dahil olacaklardır. Oyunlar 3, 5, 7 kişilik olarak açılabileceklerdir. Her bir oyun gerekli sayıda oyuncunun dahil olmasının ardından oyun kurucu istemcinin başlama komutu ile başlatılır. Oyun başlayana dek oyun için bekleyen oyuncular kendi aralarında mesajlaşabilir ve aynı oyun için bekleyen diğer oyuncuların adlarını görebilirler. Oyun başladıktan sonra mesajlaşma yasaklanır. Projenin her bir parçasının içermesi gereken özellikler aşağıda maddeler şeklinde verilmiştir.

Projede İstenilen Özellikler:

1. Sunucu ile istemcilerin iletişimi TCP veya UDP protokolü ile yapılabilir. Seçim öğrenciye bırakılmıştır.
2. Sunucu - İstemci haberleşmesi için öğrenciler kendi protokollerini tasarlayacaktır. Bu protokol listesi FTP protokolü gibi mesaj başlığı ve içeriğini birbirinden ayırt etmek için geliştirilecek uygulamada kullanılacaktır. Tasarlanan protokol, sonlu durum makinası (FSM) veya benzer bir sembolize anlatım yöntemi ile proje teslimine sunulacaktır. Protokoldeki tüm mesaj tipleri ve mesajların ilgili taraf tarafından alındığında yapılacak işlem listesi tablo şeklinde verilecektir.
3. Protokol dokümanının amacı sizin uygulamanız ile konuşacak diğer taraf uygulamasını başka bir programcının sizin kodlama bilginize veya anlatımınıza ihtiyaç duymaksızın yapabilmesini sağlamaktır. Bu kriteri yerine getiremeyecek içerikteki protokol dokümanları yetersiz sayılacaktır.
4. Protokol dokümanı docx veya pdf formatında hazırlanıp proje ile birlikte zip dosyası olarak teslim edilecektir.

YAKAR TOP OYUNU Özellikleri (3,5,7 (2n+1) Oyuncu ile oynanır.)

1. Sunucu (Random(5) + 5) sn'de bir yeni top üretir ve bunu elinde top olmayan oyunculardan rasgele seçilen birine atar.
2. Oyun içinde en fazla **n** adet (Top sayısı = n , Oyuncu sayısı: 2n +1 olur) aktif top olabilir. Eğer oyun içinde n adet top varsa sunucu bu oyun grubu için top üretmeyi keser.
3. Her topun 3 sekmelik ömrü vardır. 3 sekme sonunda top kimde kalırsa o yanar ve bir puan eksilir.
4. Bir oyuncuya top geldiğinde elinden çıkarması için 2sn beklemek zorundadır. Bu 2sn içinde topu kime atacağına karar verebilir ve ekrandan seçebilir.
5. Eğer bir oyuncu elindeki topu diğer toplar toplamda 3 hamle yapana kadar atmazsa kendisi yanar ve bir puan kaybeder.
6. Sunucu bu şekilde MAX (Oyun başında atanır.) top üreterek oyuna dahil eder.
7. Oyun sonunda en az yanan oyuncu kazanır.

Merkezi Sunucu Özellikleri (SERVER):

1. Sunucu sistemde tek bir uygulama olacaktır. Bir port üzerinden bağlanacak istemcileri kabul edecektir.
2. Sunucu sisteme bağlı oyuncuların listesini tutma ve aktif tüm oyunların durumlarını takip etmeden sorumludur.
3. Sunucu sisteme bağlanacak oyuncu (istemci) uygulamalarında kullanıcı adı şifre kontrolü yapar ve oturumları takip eder.
4. Sunucu oyunculardan gelen oyun kurma emirlerini alır, buna uygun yeni bir oyunu sisteme dahil eder ve oyuncu bekleme modunda bunu tüm oyuna dahil olmamış oyunculara iletir. Yani henüz bir oyuna dahil

olmamış tüm oyuncular sistemdeki eleman eksikliği olan ve başlamamış oyunların bir listesini sunucudan sürekli olarak (değişikliklerde) alırlar.

5. Sunucu oyunculardan yeni oyun kurma veya eleman ihtiyacı olan oyunlara dahil olma isteklerini alır ve buna uygun reaksiyon göstererek oyuncuyu ilgili oyun grubuna dahil eder.

Proje ile ilgili Notlar:

1. Geliştirilen proje iletişim kurallarının FTP benzeri protokol olarak tasarımı, sembolize gösterimi ve kısa anlatımı, protokol içerisindeki mesajların listesi ve her mesajın iletiminde yapılacak işlemler listesi doc, docx , pdf formatlarından birinde olacak şekilde projenin yanında teslim edilecektir..
2. Uygulamanız, çalışır uygulama ve kodları ile birlikte. (Kâğıt ortamında çıktı istenmemektedir. Bölüm sayfasındaki sisteme yükleme yapılacaktır. Max 8MB'lık Proje şeklinde NetBeans veya VisualStudio ortamına uygun proje dosyaları ile birlikte verilecek. Derlenmiş Projeler bazen bu boyutu aşabilmektedir. Projenizin sadece çalışır jar dosyası ve gereken kodların bulunması yeterlidir.)
3. Projenizin belirlenen bir günde (mail grubu üzerinden paylaşılacaktır) sunumunu yapmanız gerekmektedir. **Sunulmayan projeler değerlendirme dışı bırakılır.** Projeniz için hazırladığınız Protokol Anlatım dosyası ve Sunumunuz proje notunuzun %50'si uygulama özellikleri ve başarılı çalışması %50'ni oluşturacaktır.
4. Java veya harici bir dil kullanılabilir. Programlama dili seçimi öğrenciye bırakılmıştır.

Ödev Teslimi Tek bir zip dosyası şeklinde bölüm sayfasındaki sistemden yapılacaktır. Toplam dosya boyutunuz 8Mb'ı geçemez. Bu nedenle Proje içerisindeki kullanıcı istatistiği tutan dosyaları temizleyerek dosyanızı sisteme yükleyiniz.

GAMER.JAVA:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package yakartop;

import java.net.Socket;

/**
 *
 * @author student
 */
public class Gamer {
    public  String TakmaAd;
    private Socket client;
    private MainFrame myFrame;
    public  MessageList myInputMessages;
    public  MessageList myOutputMessages;
    private HandlerClient handleInput;
    private HandlerSender handleOutput;
    public Gamer(Socket socket ,  MainFrame frm)
    {
        client = socket;
        myFrame = frm;
        myInputMessages = new MessageList();
        myOutputMessages = new MessageList();
        if(client != null){
            handleInput = new HandlerClient(socket, myInputMessages, frm,this);
            handleOutput = new HandlerSender(socket, myOutputMessages);
            handleInput.start();
            handleOutput.start();
        }
    }
}
```

MESSAGELIST.JAVA

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package yakartop;

import java.util.concurrent.ArrayBlockingQueue;

/**
 *
 * @author student
 */
public class MessageList {
        private ArrayBlockingQueue<String> myQueue;
    public MessageList(){
      myQueue = new ArrayBlockingQueue<String>(100);
    }

    public synchronized void add(String newMessage)
    {
        myQueue.add(newMessage);
       notifyAll();
    }

    public synchronized String take(){
        try {
            while(myQueue.size() == 0)
            {
                wait(); //gönderilecek bir mesaj yoksa socket threadini beklet.
            }
            return myQueue.poll();

        } catch (Exception e) {
            return e.getMessage();
        }
    }

}
```

HANDLERCLIENT:JAVA

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package yakartop;

import java.io.IOException;
import java.io.PrintWriter;
import java.net.Socket;
import java.util.Scanner;

/**
 *
 * @author student
 */
public class HandlerClient extends Thread{
     private Socket client;
    private Scanner input;
    private PrintWriter output;
    private MessageList myMessages;
    private MainFrame myFrame;
    private Gamer me;
    public HandlerClient(Socket socket , MessageList msg , MainFrame frm , Gamer g)
    {
        //Set up reference to associated socket...
        client = socket;
        myMessages = msg;
        myFrame = frm;
        me = g;
        try
        {
            input = new Scanner(client.getInputStream());
        }
        catch(IOException ioEx)
        {
        }
```

```
    }

    public void run()
    {
        String received;
        do
        {
            //Accept message from client on
            //the socket's input stream...
            received = input.nextLine();
            //GELEN MESAJ TÜRÜNE VE OYUNUN DURUMUNA GÖRE YAPILACAK
İŞLEMLER:
           //PROTOKOL ve GAME LOGIC BURADA YER ALACAK.
           if(received != null && received != ""){
            myFrame.makeAction(received, me);
           }
           received = "";

        //Repeat above until 'QUIT' sent by client...
        }while (!received.equals("QUIT"));

        try
        {
            if (client!=null)
            {
                System.out.println("Closing down connection...");
                client.close();
            }
        }
        catch(IOException ioEx)
        {
                System.out.println("Unable to disconnect!");
        }
    }
}
```

HANDLERSENDER.JAVA

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package yakartop;

import java.io.IOException;
import java.io.PrintWriter;
import java.net.Socket;

/**
 *
 * @author student
 */
public class HandlerSender extends Thread{
    private Socket client;

    private PrintWriter output;
    private MessageList myMessages;
    public HandlerSender(Socket socket , MessageList msg){
        client = socket;
        myMessages = msg;
        try
        {
            output = new PrintWriter(client.getOutputStream(),true);
        }
        catch(IOException ioEx)
        {
        }
    }

    public void run(){
        String message;
        do{
          message =  myMessages.take();
            output.println(message);
        }
        while(message !=  "QUIT");

        try
        {
            if (client!=null)
            {
                System.out.println("Closing down connection...");
                client.close();
```

```
            }
        }
        catch(IOException ioEx)
        {
                System.out.println("Unable to disconnect!");
        }
    }
}
```

SERVER LISTENER

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package yakartop;

import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author student
 */
public class ServerListener extends Thread {
    private int portNumber;
    private int clientCount;
    private MainFrame myFrm;
    private ServerSocket serverSocket;

    public ServerListener(String port, MainFrame mFrm)
    {
        portNumber = Integer.parseInt( port);
        serverSocket = null;
        clientCount = 1;
        Gamer g1 = new Gamer(null, mFrm);
        mFrm.gamers[0] = g1;
        myFrm = mFrm;
```

```
    }


    public void run(){
        try
        {
            serverSocket = new ServerSocket(portNumber);
        }
        catch (IOException ioEx)
        {
                System.exit(1);
        }

        do
        {
            try {
                Socket client = serverSocket.accept();
                Gamer newGamer = new Gamer(client, myFrm);
                myFrm.gamers[clientCount] = newGamer;
                clientCount++;
            } catch (IOException ex) {
                Logger.getLogger(ServerListener.class.getName()).log(Level.SEVERE, null, ex);
            }

        }while(clientCount <= 5);




    }
}
```

MAINFRAME.JAVA

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
```

```
 * and open the template in the editor.
 */
package yakartop;

import java.io.IOException;
import java.net.InetAddress;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author student
 */
public class MainFrame extends javax.swing.JFrame {
    public Gamer[] gamers = new Gamer[5];
    public Gamer server = null;
    private static InetAddress host;

    public MainFrame() {
        initComponents();
    }
    public void printResult(String msg){
        txtaMesajlar.setText(txtaMesajlar.getText() + "\n"  + msg);
    }

    public void makeAction(String msg)
    {
        printResult(msg);
        repaint();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        txtIp = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        txtPort = new javax.swing.JTextField();
```

```
btnHostGame = new javax.swing.JButton();
btnJoinGame = new javax.swing.JButton();
jScrollPane1 = new javax.swing.JScrollPane();
txtaMesajlar = new javax.swing.JTextArea();
txtTahmin = new javax.swing.JTextField();
btnSend = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

jLabel1.setText("İP:");

txtIp.setText("127.0.0.1");

jLabel2.setText("Port");

txtPort.setText("5005");

btnHostGame.setText("Host Game");
btnHostGame.setName("btnHostGame"); // NOI18N
btnHostGame.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnHostGameActionPerformed(evt);
    }
});

btnJoinGame.setText("Join Game");
btnJoinGame.setName("btnJoinGame"); // NOI18N
btnJoinGame.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnJoinGameActionPerformed(evt);
    }
});

txtaMesajlar.setColumns(20);
txtaMesajlar.setRows(5);
jScrollPane1.setViewportView(txtaMesajlar);

btnSend.setText("Gönder");
btnSend.setName("btnSend"); // NOI18N
btnSend.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSendActionPerformed(evt);
    }
});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
```

```
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(txtIp, javax.swing.GroupLayout.PREFERRED_SIZE, 106,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(txtPort, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(52, 52, 52)
                        .addComponent(btnHostGame)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(btnJoinGame)
                        .addContainerGap())
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addComponent(txtTahmin, javax.swing.GroupLayout.PREFERRED_SIZE,
61, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(btnSend)
                        .addGap(172, 172, 172))))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel1)
                    .addComponent(txtIp, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel2)
```

```
                .addComponent(txtPort, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(btnHostGame)
                .addComponent(btnJoinGame))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 188,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(btnSend)
                .addComponent(txtTahmin, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(18, 18, 18)
            .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 237,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void btnHostGameActionPerformed(java.awt.event.ActionEvent evt) {
        ServerListener newHosting = new ServerListener(txtPort.getText(), this);
        newHosting.start();
    }

    private void btnJoinGameActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            host = InetAddress.getByName(txtIp.getText());
            Socket client = null;
            client = new Socket(host,Integer.parseInt(txtPort.getText()));
            server = new Gamer(client,this);
            printResult("Bağlanıldı Hoşgeldin mesajı bekleniyor.");
            btnHostGame.setEnabled(false);
            btnJoinGame.setEnabled(false);
        } catch (UnknownHostException ex) {
            Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
        } catch (IOException ex) {
            Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    private void btnSendActionPerformed(java.awt.event.ActionEvent evt) {

        server.myOutputMessages.add("Mesaj " + txtTahmin.getText());

    }
```

```
    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and
feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MainFrame.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MainFrame.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MainFrame.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MainFrame.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MainFrame().setVisible(true);
            }
        });
    }
```

```
    // Variables declaration - do not modify
    private javax.swing.JButton btnHostGame;
    private javax.swing.JButton btnJoinGame;
    private javax.swing.JButton btnSend;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextField txtIp;
    private javax.swing.JTextField txtPort;
    private javax.swing.JTextField txtTahmin;
    private javax.swing.JTextArea txtaMesajlar;
    // End of variables declaration
}
```