Conecta 4

PROP 2021-22 Q2

Mario Martin – Ignasi Gómez Sebastià – Bernat Orellana

Tècniques d'intel·ligència Artificial

- Algoritmes per a jocs:
  - MIN-MAX
  - MIN-MAX amb poda alpha-beta

Objectiu

- Guanyar una partida de conecta 4
- Adversaris:
  - Aficionat (Aleatori)
    - Guanyar
  - Professional (MIN-MAX)
    - Guanyar o empatar

Programació bàsica

- Fitxer d'exemple *Juga2.java*
  - *Constructor*

```
 */
public Juga2(Jugador p1, Jugador p2, boolean useAutoMode) {
    initComponents();

    jTextField1.setEnabled(false);
    jTextField2.setEnabled(false);
    jTextField3.setEnabled(false);
    player1 = p1;
    player2 = p2;
    this.autoMode = useAutoMode;

    init();

}

private void init() {
    t = new Tauler(8);

    currentPlayer = player1;
    otherPlayer = player2;
    currentColor = 1;
    otherColor = -1;

    jTextField1.setText(player1.nom());
    jTextField3.setText(player2.nom());


    Dimension mides = jLayeredPane1.getSize();
    Ymax = mides.getHeight();
    Xmax = mides.getWidth();
    Step = (int) Xmax / 8;
}
```

Creació del tauler

Programació bàsica

- Fitxer d'exemple *Juga2.java*
  - *Programa Principal*

```
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(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>


    // Definiu al vostre gust els jugadors a enfrontar.
    Jugador p1 = new Manual();
    //Jugador p1 = new Aleatori();

    Jugador p2 = new Profe(8,false);
    //Jugador p2 = new Manual();

    boolean autoMode = true;
    final Juga2 j = new Juga2(p1,p2, autoMode);

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

Programació bàsica

- Fitxer d'exemple *Juga2.java*
  - *Click i fil de background*

```
private void jLayeredPane1MouseClicked(java.awt.event.MouseEvent evt) {//GEN-FIRST:event_jLayeredPaneMouseClicked
      // TODO add your handling code here:

    if(estaPensant) return;

    int X = evt.getX();
    int Y = evt.getY();


    int col = (int) X / Step;

    if ("Manual".equals(currentPlayer.nom())) {
        mouCurrentPlayer(col);
    } else {
        runAuto();
    }
    //mostraTornActual();

} //GEN-LAST:event_jLayeredPaneMouseClicked


private void mouCurrentPlayer(int colu) {

    t.afegeix(colu, currentColor);
    repaint();
    verificaSiHaAcabat(colu, currentColor);

}

private void runAuto() {
    estaPensant = true;
    // Executem l'automàtic en background
    String t =  (currentPlayer==player1?"P1":"P2")+ " ESTIC PENSANT.....";
    jTextField2.setText(t);
    jLayeredPane1.setBackground(new java.awt.Color(255, 255, 0));
    jLayeredPane1.setEnabled(false);
    (new Mover(currentColor, currentPlayer)).execute();
}
```

Programació bàsica

- Fil per "pensar" el moviment.

```
class Mover extends SwingWorker<Integer, Object> {

    int color;
    Jugador jugador;

    Mover(int color, Jugador jugador) {
        this.color = color;
        this.jugador = jugador;
    }

    @Override
    public Integer doInBackground() {
        return jugador.moviment(t, color);
    }

    @Override
    protected void done() {
        try {

            jLayeredPanel.setBackground(new java.awt.Color(255, 255, 255));
            jLayeredPanel.setEnabled(true);

            mouCurrentPlayer(get());
            repaint();
            estaPensant = false;
        } catch (Exception ignore) {
        }
    }
}
```

Programació bàsica

- Fitxer d'exemple *Juga.*
  - *Final del joc*

```
private void verificaSiHaAcabat(int colu, int color) {

    String text1 = "", text2 = "", text3 = "", dTitle = "";

    if (t.solucio(colu, color) || !t.espotmoure()) {

        if (t.solucio(colu, color)) {
            if (currentPlayer == player1) {
                text1 = "WINNER";
                text3 = "LOSER";
                text2 = "VERMELL AMB MOVIMENT A COLUMNA " + (colu + 1);
                dTitle = "GUANYA P1(" + currentPlayer.nom() + ")";
            } else {
                text3 = "WINNER";
                text1 = "LOSER";
                text2 = "BLAU AMB MOVIMENT A COLUMNA " + (colu + 1);
                dTitle = "GUANYA P2(" + currentPlayer.nom() + ")";
            }
        } else {
            // no es pot moure
            text1 = "NO PUC MOURE";
            text3 = "NO PUC MOURE";
            text2 = "TAULES";
            dTitle = "TAULES";
        }

        jTextField1.setText(text1);
        jTextField2.setText(text2);
        jTextField3.setText(text3);

        int n = JOptionPane.showConfirmDialog(
                this, dTitle,
                "Tornar a jugar",
                JOptionPane.YES_NO_OPTION);
        if (n == JOptionPane.YES_OPTION) {
            init();
        } else if (n == JOptionPane.NO_OPTION) {
            System.exit(0);
        }
    } else {
        canviTorn();
    }
    mostraTornActual();
}
```

Programació bàsica

- Interfície del jugador

```java
/**
 * Interfície que han d'implementar tots els Jugadors (automàtics i manuals)
 * de la nostra aplicació.
 * @author Profe
 */
public interface Jugador {
    /**
     * Decideix el moviment del jugador donat un tauler i un color de peça que ha de posar.
     * @param t  Tauler actual de joc
     * @param color  Color de la peça que possarà
     * @return Columna on fer el moviment
     */
    public int moviment(Tauler t, int color);

    /**
     * Retorna el nom del jugador que s'utlilitza per visualització a la UI
     * @return Nom del jugador
     */
    public String nom();
}
```

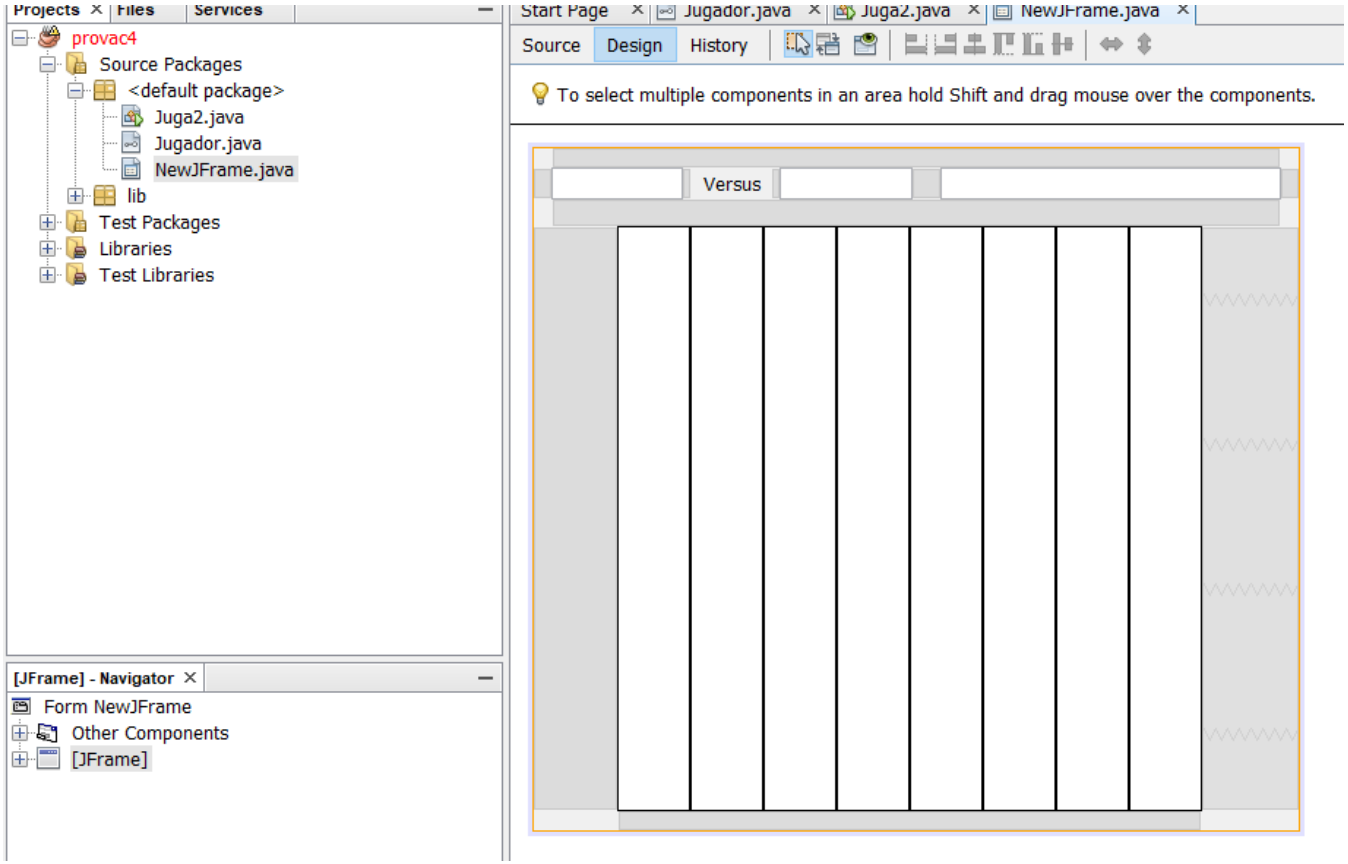
Programació bàsica

- Ex: Jugador aleatori

```
/**
 * Jugador aleatori
 * "Alea jacta est"
 * @author Profe
 */
public class Aleatori
  implements Jugador, IAuto
{
  private String nom;

  public Aleatori()
  {
    nom = "RandomBanzai";
  }

  public int moviment(Tauler t, int color)
  {
    int col = (int)(8.0D * Math.random());
    while (!t.movpossible(col)) {
      col = (int)(8.0D * Math.random());
    }
    return col;
  }

  public String nom()
  {
    return nom;
  }
}
```

Programació bàsica

- Interfície gràfica (Jframe)



Programació bàsica

- Documentació
  - *Fitxer : javadoc_libc4/index.html*

Instal.lació

- Obrir projecte
  - Obrir directori ***c4_the_game*** com a projecte Netbeans
  - Executar ***Juga2*** per provar
    - Es poden canviar paràmetres dels participants a **main**.
    - Podeu activar o desactivar que els jugadors ***IAuto*** juguin automàticament (autoMode=true) o esperin al clic (autoMode=false)

```
/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)


    // Definiu al vostre gust els jugadors a enfrontar.
    Jugador p1 = new Manual();
    //Jugador p1 = new Aleatori();

    Jugador p2 = new Profe(8,false);
    //Jugador p2 = new Manual();

    boolean autoMode = true;
    final Juga2 j = new Juga2(p1,p2, autoMode);
```

Activitat

- Grups de 2
- Zip amb
  - Codi (Projecte NetBeans sense *build* ni *dist*)
  - Document PDF amb documentació
  - Fitxer de text amb noms i DNIs dels alumnes
- El nom del zip son els dnis dels alumnes del grup
- Només lliura una persona de la parella.

Avaluació

- 10% Nota
- Nota
  - 70% Codi i funcionalitat
    - 20% El jugador funciona i:
      - No triga més de un minut en decidir cada moviment (a profunditat 8).
      - S'ha parametritzat el codi amb profunditat màxima
      - Es mostra per consola el nombre de jugades finals explorades (amb heurística calculada)
    - 30% Guanya ***Aleatori***
    - 20% Guanya o empata amb ***Profe*** **(**l'automàtic ☺ !**)**
      - Profunditat 8. Assegureu que la profunditat és la que dieu que és !

Avaluació

- 30% Documentació
  - 5%: Documentació del codi (JavaDoc) generada en una carpeta **./Javadoc** dins del mateix projecte.
  - 15%: Explicació de l'Heurística dissenyada pel jugador i detalls d'implementació que considereu rellevants.
  - 10% Estudi de la incidència de la poda alfa-beta en el número de nodes explorats. Possible incidència de l'ordenació dels nodes en la poda.

Avaluació

- Condicions
  - Lliurament: dijous 25 de novembre, 23:59 (improrrogable)
  - El jugador ha de ser un treball original vostre
  - El codi del jugador ha de ser correcte
    - Netbeans no reporta errors
    - El projecte compila
    - Es pot carregar el jugador al main del Juga2