```java
public class Turtle
{
    private Point position;
    private String name;
    private Color color;

    /**
     * orientation:
     * 0 = haut
     * 1 = bas
     * 2 = gauche
     * 3 = droite
     */
    private int orientation = 0; // regarder vers le haut au départ

    public Turtle(Point position, String name, Color color)
    {
        this.position = position;
        this.name = name;
        this.color = color;
    }

    public Point getPosition()
    {
        return position;
    }

    public String getName()
    {
        return name;
    }

    public String toString()
    {
        return name;
    }

    public void goRight(int pDist)
    {
        position.x += pDist;
        orientation = 3;
        // ou:
        //   position.setLocation(position.x+pDist, position.y);
    }

    public void goLeft(int pDist)
    {
        position.x -= pDist;
        orientation = 2;
        // ou:
        //   position.setLocation(position.x-pDist, position.y);
    }

    public void goUp(int pDist)
    {
        position.y -= pDist;
        orientation = 0;
        // ou:
        //   position.setLocation(position.x, position.y-pDist);
    }

    public void goDown(int pDist)
    {
        position.y += pDist;
        orientation = 1;
        // ou:
        //   position.setLocation(position.x, position.y+pDist);
    }

    public void draw(Graphics g)
    {
        // Dessiner la tortue avec la couleur donnée
        int x = getPosition().x;
        int y = getPosition().y;

        g.setColor(color);

        if (orientation == 0) // haut
        {
            g.fillOval(x - 8, y - 6, 16, 16);
            g.fillOval(x - 3, y - 12, 6, 8);
            g.drawLine(x - 10, y - 5, x + 10, y + 12);
```

```
            g.drawLine(x + 10, y - 5, x - 10, y + 12);
        }
        else if (orientation == 1) // bas
        {
            g.fillOval(x - 8, y - 10, 16, 16);
            g.fillOval(x - 3, y + 4, 6, 8);
            g.drawLine(x - 10, y + 5, x + 10, y - 12);
            g.drawLine(x + 10, y + 5, x - 10, y - 12);
        }
        else if (orientation == 2) // gauche
        {
            g.fillOval(x - 6, y - 8, 16, 16);
            g.fillOval(x - 12, y - 3, 8, 7);
            g.drawLine(x - 5, y - 10, x + 12, y + 10);
            g.drawLine(x - 5, y + 10, x + 12, y - 10);
        }
        else // droite
        {
            g.fillOval(x - 10, y - 8, 16, 16);
            g.fillOval(x + 4, y - 3, 8, 7);
            g.drawLine(x + 5, y - 10, x - 12, y + 10);
            g.drawLine(x + 5, y + 10, x - 12, y - 10);
        }

        // Dessiner le nom à côté en bleu
        g.drawString(name, x - 10, y + 24);
    }
}
```

```java
public class Turtles
{
    private ArrayList<Turtle> alTurtles = new ArrayList<>();

    public Turtle remove(int pIndex)
    {
        return alTurtles.remove(pIndex);
    }

    public Turtle get(int pIndex)
    {
        return alTurtles.get(pIndex);
    }

    public void add(Turtle pTurtle)
    {
        alTurtles.add(pTurtle);
    }

    public Object[] toArray()
    {
        return alTurtles.toArray();
    }

    public void draw(Graphics g)
    {
        for (int i = 0; i < alTurtles.size(); i++)
            alTurtles.get(i).draw(g);
    }

    public int findByName(String pName)
    {
        // Version plus correcte algorithmiquement,
        // mais moins élégante et moins facile à comprendre

        boolean found = false;
        int i = 0;
        while ((!found) && (i<alTurtles.size()))
        {
            if (alTurtles.get(i).getName().equals(pName))
                found = true;
            else
                i++;
        }
        if (found)
            return i;
        else
            return -1;
    }
}
```

```java
public class DrawPanel extends javax.swing.JPanel
{
    private Turtles turtles = null;

    public void setTurtle(Turtles turtles)
    {
        this.turtles = turtles;
    }

    public DrawPanel()
    {
        initComponents();
    }

    public void paintComponent(Graphics g)
    {
        int w = getWidth();
        int h = getHeight();

        g.setColor(Color.WHITE);
        g.fillRect(0, 0, w, h);

        if (turtles != null)
            turtles.draw(g);
    }
// Skipped: ... initComponents { ... }
// Skipped: ... graphic attributes
}
```

```
public class MainFrame extends javax.swing.JFrame
{
     private Turtles turtles = new Turtles();

     public MainFrame()
     {
          initComponents();
          drawPanel.setTurtle(turtles);
          turtleList.setListData(turtles.toArray());
          updateView();
     }

     public void updateView()
     {
          turtleList.setListData(turtles.toArray());
          repaint();
     }
// Skipped: ... initComponents { ... }
    private void rightButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_rightButtonActionPerformed
    {//GEN-HEADEREND:event_rightButtonActionPerformed
          int i = turtleList.getSelectedIndex();
          if (i >= 0)
               turtles.get(i).goRight(10);
          repaint();
    }//GEN-LAST:event_rightButtonActionPerformed

    private void leftButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_leftButtonActionPerformed
    {//GEN-HEADEREND:event_leftButtonActionPerformed
          int i = turtleList.getSelectedIndex();
          if (i >= 0)
               turtles.get(i).goLeft(10);
          repaint();
    }//GEN-LAST:event_leftButtonActionPerformed

    private void upButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_upButtonActionPerformed
    {//GEN-HEADEREND:event_upButtonActionPerformed
          int i = turtleList.getSelectedIndex();
          if (i >= 0)
               turtles.get(i).goUp(10);
          repaint();
    }//GEN-LAST:event_upButtonActionPerformed

    private void downButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_downButtonActionPerformed
    {//GEN-HEADEREND:event_downButtonActionPerformed
          int i = turtleList.getSelectedIndex();
          if (i >= 0)
               turtles.get(i).goDown(10);
          repaint();
    }//GEN-LAST:event_downButtonActionPerformed

    private void addButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_addButtonActionPerformed
    {//GEN-HEADEREND:event_addButtonActionPerformed
          Point position = new Point((int) (Math.random() * drawPanel.getWidth()),
                  (int) (Math.random() * drawPanel.getHeight()));
          Turtle turtle = new Turtle(position, nameTextField.getText(), colorChooser.getColor());
          turtles.add(turtle);
          updateView();
    }//GEN-LAST:event_addButtonActionPerformed

    private void removeButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_removeButtonActionPerformed
    {//GEN-HEADEREND:event_removeButtonActionPerformed
          int i = turtleList.getSelectedIndex();
          if (i >= 0)
               turtles.remove(i);
          updateView();
    }//GEN-LAST:event_removeButtonActionPerformed

    private void newButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_newButtonActionPerformed
    {//GEN-HEADEREND:event_newButtonActionPerformed
          turtles = new Turtles();
          drawPanel.setTurtle(turtles);
          updateView();
    }//GEN-LAST:event_newButtonActionPerformed

    private void findButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_findButtonActionPerformed
    {//GEN-HEADEREND:event_findButtonActionPerformed
          int i = turtles.findByName(nameTextField.getText());
          if (i >= 0)
               turtleList.setSelectedIndex(i);
    }//GEN-LAST:event_findButtonActionPerformed
// Skipped: ... Look & Feel
// Skipped: ... graphic attributes
```

```
}
```