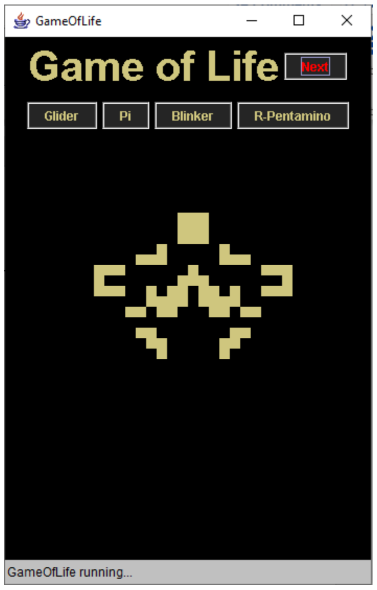
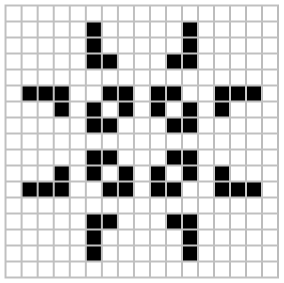
Conway’s Game of Life

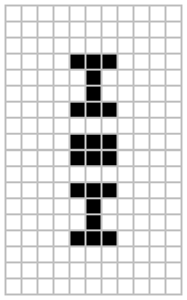
The Game of Life is zero-player game. Each square on a grid is in one of two possible states: live or dead. The next generation of each cell is determined by its interactions with its eight neighbours:

- Any live cell with fewer than two live neighbours dies, as if by loneliness.
- Any live cell with two or three live neighbours lives on to the next generation.
- Any live cell with more than three live neighbours dies, as if by overpopulation.
- Any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.

By clicking the next button, you can see the next generation.







Your Assignment:

- Code the NextGeneration method.
- Specifically, the part where you need to count your neighbours is incorrect – it is missing edge guards.
- It is highlighted in the code below.

```
import javax.swing.*;
import java.applet.*;
import java.awt.event.*;
import java.awt.*;
public class GameOfLife extends Applet implements ActionListener
{
    int row = 36;
    int col = 30;
    int tracker[] [] = new int [row] [col];
    JButton pics[] = new JButton [row * col];

    public void init ()
    {

        //set up R-Pentamino
        tracker [16] [15] = 1;
        tracker [15] [16] = 1;
        tracker [16] [16] = 1;
        tracker [16] [17] = 1;
        tracker [17] [15] = 1;

        resize (350, 500);
        setBackground (Color.black);
        JLabel title = new JLabel ("Game of Life");
        title.setFont (new Font ("Arial", Font.BOLD, 40));
        title.setForeground (new Color (207, 198, 126));
        add (title);
        Panel p = new Panel ();
        JButton next = new JButton ("Next");
        next.setBackground (new Color (37, 37, 37));
        next.setForeground (Color.red);
        next.addActionListener (this);
        next.setActionCommand ("next");
        add (next);
        JButton glider = new JButton ("Glider");
        glider.setBackground (new Color (37, 37, 37));
        glider.setForeground (new Color (207, 198, 126));
        glider.addActionListener (this);
        glider.setActionCommand ("Glider");
        p.add (glider);
        JButton pi = new JButton ("Pi");
        pi.setBackground (new Color (37, 37, 37));
        pi.setForeground (new Color (207, 198, 126));
        pi.addActionListener (this);
        pi.setActionCommand ("Pi");
        p.add (pi);
        JButton blinker = new JButton ("Blinker");
        blinker.setBackground (new Color (37, 37, 37));
        blinker.setForeground (new Color (207, 198, 126));
        blinker.addActionListener (this);
        blinker.setActionCommand ("Blinker");
        p.add (blinker);
          JButton r = new JButton ("R-Pentamino");
        r.setBackground (new Color (37, 37, 37));
        r.setForeground (new Color (207, 198, 126));
        r.addActionListener (this);
        r.setActionCommand ("R-Pentamino");
        p.add (r);
        add (p);
        Panel grid = new Panel (new GridLayout (row, col));
        int m = 0;
        for (int i = 0 ; i < row ; i++)
        {
            for (int j = 0 ; j < col ; j++)
            {

                pics [m] = new JButton ();
                pics [m].setPreferredSize (new Dimension (10, 10));
                pics [m].setBorderPainted (false);
                pics [m].setBackground (Color.green);
                pics [m].setActionCommand ("" + m);
                pics [m].addActionListener (this);
                grid.add (pics [m]);
                m++;
            }
        }
        add (grid);
        redraw ();
    }


    public void reset ()
    {
        for (int i = 0 ; i < row ; i++)
        {
            for (int j = 0 ; j < col ; j++)
            {
                tracker [i] [j] = 0;

            }
        }
    }


    public void actionPerformed (ActionEvent e)
    {
        if (e.getActionCommand ().equals ("Blinker"))
        {
            reset ();
            tracker [8] [8] = 1;
            tracker [9] [8] = 1;
            tracker [10] [8] = 1;
            redraw ();
        }
        else if (e.getActionCommand ().equals ("Pi"))
        {
            reset ();
            tracker [15] [14] = 1;
            tracker [16] [14] = 1;
            tracker [17] [14] = 1;
            tracker [15] [15] = 1;
            tracker [15] [16] = 1;
            tracker [16] [16] = 1;
            tracker [17] [16] = 1;
            redraw ();
        }
        else if (e.getActionCommand ().equals ("Glider"))
        {
            reset ();
            tracker [1] [3] = 1;
            tracker [2] [1] = 1;
            tracker [2] [3] = 1;
            tracker [3] [2] = 1;
            tracker [3] [3] = 1;

            tracker [7] [7] = 1;
            tracker [8] [5] = 1;
            tracker [8] [7] = 1;
            tracker [9] [6] = 1;
            tracker [9] [7] = 1;
            redraw ();
        }
        else if (e.getActionCommand ().equals ("R-Pentamino"))
        {
            reset ();
            tracker [16] [15] = 1;
            tracker [15] [16] = 1;
            tracker [16] [16] = 1;
            tracker [16] [17] = 1;
            tracker [17] [15] = 1;
            redraw ();
        }
        else if (e.getActionCommand ().equals ("next"))
            nextGeneration ();
    }


    public void redraw ()
    {
        int m = 0;
        for (int i = 0 ; i < row ; i++)
        {
            for (int j = 0 ; j < col ; j++)
            {
                if (tracker [i] [j] == 1)
                    pics [m].setBackground (new Color (207, 198, 126));
                else
                    pics [m].setBackground (Color.black);
                m++;
            }
        }

    }


    public void nextGeneration ()
    {
        int next[] [] = new int [row] [col];

        int count = 0;
        for (int i = 0 ; i < row ; i++)
        {
            for (int j = 0 ; j < col ; j++)
            {
                count = 0;
                if (tracker [i - 1] [j - 1] == 1)
                    count++;
                if (tracker [i - 1] [j] == 1)
                    count++;
                if (tracker [i - 1] [j + 1] == 1)
                    count++;

                if (tracker [i] [j - 1] == 1)
                    count++;
                if (tracker [i] [j + 1] == 1)
                    count++;

                if (tracker [i + 1] [j - 1] == 1)
                    count++;
                if (tracker [i + 1] [j] == 1)
                    count++;
                if (tracker [i + 1] [j + 1] == 1)
                    count++;

                if (tracker [i] [j] == 1 && (count == 2 || count == 3))
                    next [i] [j] = 1;
                else if (tracker [i] [j] == 0 && (count == 3))
                    next [i] [j] = 1;
            }
        }
        for (int i = 0 ; i < row ; i++)
        {
            for (int j = 0 ; j < col ; j++)
            {
                tracker [i] [j] = next [i] [j];

            }
        }
        redraw ();
    }
}
```