

```
public class MovingObject
{
    private int x;
    private int y;

    private int stepX = 0;
    private int stepY = 0;

    public MovingObject(int x, int y)
    {
        this.x = x;
        this.y = y;
    }

    public int getX()
    {
        return x;
    }

    public void setX(int x)
    {
        this.x = x;
    }

    public int getY()
    {
        return y;
    }

    public void setY(int y)
    {
        this.y = y;
    }

    public int getStepX()
    {
        return stepX;
    }

    public void setStepX(int stepX)
    {
        this.stepX = stepX;
    }

    public int getStepY()
    {
        return stepY;
    }

    public void setStepY(int stepY)
    {
        this.stepY = stepY;
    }

    public void move()
    {
        x += stepX;
        y += stepY;
    }
}
```

```
public class Ball extends MovingObject
{
    private int radius;

    public Ball(int x, int y, int radius)
    {
        super(x, y);
        this.radius = radius;

        setStepY(5);
    }

    public int getRadius()
    {
        return radius;
    }

    public void move(int width)
    {
        super.move();

        if (getX() < radius)
            setX(radius);
        else if (getX() + radius > width)
            setX(width - radius);
    }

    public void draw(Graphics g)
    {
        g.setColor(Color.RED);
        g.fillOval(getX() - radius, getY() - radius, 2 * radius, 2 * radius);
        g.setColor(Color.BLACK);
        g.drawOval(getX() - radius, getY() - radius, 2 * radius, 2 * radius);
    }
}
```

```
public class Block extends MovingObject
{
    private int width;
    private int height;

    public Block(int x, int y, int width, int height)
    {
        super(x, y);
        this.width = width;
        this.height = height;

        setStepY(-3);
    }

    public int getWidth()
    {
        return width;
    }

    public void setWidth(int width)
    {
        this.width = width;
    }

    public int getHeight()
    {
        return height;
    }

    public boolean isTouching(Ball ball)
    {
        int bx = ball.getX();
        int by = ball.getY() + ball.getRadius();

        return (bx >= getX() && (bx <= getX() + width)
            && (by >= getY() && (by <= getY() + height));
    }

    public void draw(Graphics g)
    {
        g.setColor(Color.GREEN);
        g.fillRect(getX(), getY(), width, height);
        g.setColor(Color.BLACK);
        g.drawRect(getX(), getY(), width, height);
    }
}
```

```

public class Game
{
    private ArrayList<Block> alBlocks = new ArrayList<>();
    private Ball ball = null;

    private int points = 0;
    private int lives = 3;

    public Game(int width, int height)
    {
        int x, w;

        for (int i = 0; i < 10; i++)
        {
            x = (int) (Math.random() * width / 2);
            w = (int) (Math.random() * (width - x) * 2 / 3);

            Block block = new Block(x, i * height / 10, w, height / 30);
            alBlocks.add(block);
        }

        ball = new Ball(width / 2, height / 2, height / 40);
    }

    public int getPoints()
    {
        return points;
    }

    public int getLives()
    {
        return lives;
    }

    public void ballLeft()
    {
        ball.setStepX(-10);
    }

    public void ballRight()
    {
        ball.setStepX(10);
    }

    public void move(int width, int height)
    {
        Block b;
        boolean touching = false;
        for (int i = 0; i < alBlocks.size(); i++)
        {
            b = alBlocks.get(i);
            b.move();
            if (b.getY() < 0)
            {
                b.setY(height);
                b.setX((int) (Math.random() * width / 2));
                b.setWidth((int) (Math.random() * (width - b.getX()) * 2 / 3));

                points++;
                if (points % 100 == 0)
                    lives++;
            }
            if (b.isTouching(ball))
            {
                touching = true;
                ball.setStepY(b.getStepY());
            }
        }
        if (!touching)
        {
            ball.setStepY(5);
            ball.setStepX(0);
        }
        ball.move(width);
        if ((ball.getY() < 0) || (ball.getY() > height))
        {
            lives--;
            ball.setY(height / 2);
            ball.setX(width / 2);
        }
    }

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

        if (ball != null)
            ball.draw(g);
    }
}

```

```
public class DrawPanel extends javax.swing.JPanel
{
    private Game game = null;

    public DrawPanel()
    {
        initComponents();
    }

    public void setGame(Game game)
    {
        this.game = game;
    }

    public void paintComponent(Graphics g)
    {
        g.setColor(Color.WHITE);
        g.fillRect(0, 0, getWidth(), getHeight());

        if (game != null)
            game.draw(g);
    }
}
// Skipped: ... initComponents { ... }
// Variables declaration - do not modify//GEN-BEGIN:variables
// End of variables declaration//GEN-END:variables
```

```

public class MainFrame extends javax.swing.JFrame
{
    private Game game = null;
    private Timer timer = null;

    public MainFrame()
    {
        initComponents();

        timer = new Timer(20, stepButton.getActionListeners()[0]);
        stepButton.setVisible(false);
    }
    // Skipped: ... initComponents { ... }
    private void startButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_startButtonActionPerformed
    { //GEN-HEADEREND:event_startButtonActionPerformed
        game = new Game(drawPanel.getWidth(), drawPanel.getHeight());
        drawPanel.setGame(game);

        timer.start();
        startButton.setVisible(false);
    } //GEN-LAST:event_startButtonActionPerformed

    private void stepButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_stepButtonActionPerformed
    { //GEN-HEADEREND:event_stepButtonActionPerformed
        game.move(drawPanel.getWidth(), drawPanel.getHeight());

        pointsLabel.setText("Points = " + game.getPoints());
        livesLabel.setText("Lives = " + game.getLives());

        if (game.getLives() == 0)
        {
            timer.stop();
            drawPanel.setGame(null);
            startButton.setVisible(true);
            livesLabel.setText("GAME OVER!!");
        }
        repaint();
    } //GEN-LAST:event_stepButtonActionPerformed

    private void drawPanelMousePressed(java.awt.event.MouseEvent evt)//GEN-FIRST:event_drawPanelMousePressed
    { //GEN-HEADEREND:event_drawPanelMousePressed
        if (game == null)
            return;

        if (evt.getButton() == MouseEvent.BUTTON1)
            game.ballLeft();
        else if (evt.getButton() == MouseEvent.BUTTON3)
            game.ballRight();
    } //GEN-LAST:event_drawPanelMousePressed

    private void drawPanelMouseReleased(java.awt.event.MouseEvent evt)//GEN-FIRST:event_drawPanelMouseReleased
    { //GEN-HEADEREND:event_drawPanelMouseReleased
        // Ne rien faire...
    } //GEN-LAST:event_drawPanelMouseReleased
    // Skipped: ... Look & Feel
    // Variables declaration - do not modify//GEN-BEGIN:variables
    private DrawPanel drawPanel;
    private javax.swing.JLabel livesLabel;
    private javax.swing.JLabel pointsLabel;
    private javax.swing.JButton startButton;
    private javax.swing.JButton stepButton;
    // End of variables declaration//GEN-END:variables
}

```