```
import java.awt.*;
    import java.awt.event.*;
2
    import java.awt.image.*;
3
    import java.io.*;
4
    import java.util.*;
5
    import javax.imageio.*;
6
7
    import javax.swing.*;
8
    public class Background {
9
        private Image background;
10
        private ArrayList<Shrub> shrubs;
11
        private ArrayList<BufferedImage> images;
12
13
         public Background() {
14
             try {
15
                 background = ImageIO.read(new File("background.png"));
16
                 shrubs = new ArrayList<>();
17
                 images = new ArrayList<>();
18
19
                 images.add(ImageIO.read(new File("tree.png")));
                 images.add(ImageIO.read(new File("shrub.png")));
20
                 shrubs.add(new Shrub(1000, 400 - images.get(0).getHeight(), images.get(0)));
21
                 //images.add(ImageIO.read(new File("")));
22
             }
23
             catch(IOException e) {
24
25
                 System.out.println("io");
             }
26
        }
27
28
         public void scroll(double v) {
29
             for(Shrub s : shrubs) {
30
                 s.scroll();
31
                 s.updateV(v);
32
             }
33
             if(shrubs.get(0).getXCoord() <= -70) {</pre>
34
                 BufferedImage img = images.get((int)(Math.random() * images.size()));
35
                 shrubs.add(new Shrub(1000, 400 - img.getHeight(), img));
36
                 shrubs.remove(0);
37
             }
38
        }
39
40
         public void draw(Graphics g) {
41
             g.drawImage(background, 0, 0, null);
42
             for(Shrub s : shrubs) {
43
                 s.draw(g);
44
             }
45
        }
46
47
    }
48
```

```
import java.awt.*;
    import java.awt.event.*;
2
3
    import java.awt.image.*;
4
    import java.io.*;
    import java.util.*;
5
6
    import javax.swing.*;
7
    public class Floor {
8
9
        private int xCoord;
10
        private int yCoord;
11
12
        public Floor(int x, int y) {
13
            xCoord = x;
14
            yCoord = y;
15
        }
16
        public void draw(Graphics g) {
17
            g.setColor(Color.black);
18
            g.fillRect(xCoord, yCoord, 1000, 500);
19
        }
20
    }
21
```

```
import java.awt.*;
    import java.awt.event.*;
2
    import java.awt.image.*;
3
    import java.io.*;
4
    import java.util.*;
5
    import javax.imageio.*;
6
    import javax.swing.*;
7
8
    public class Highscore extends Score {
9
        private BufferedImage img;
10
        private Image imageNumber;
11
        private int highscore;
12
13
        public Highscore(int x) {
14
             highscore = x;
15
16
             try {
                 img = ImageIO.read(new File("highscore.png"));
17
18
             catch(IOException e) {}
19
        }
20
21
        public void draw(Graphics g) {
22
             Integer inte = new Integer(highscore);
23
             String num = inte.toString();
24
25
             char c;
             int x = 0;
26
             try {
27
                 for(int i = 0; i < num.length(); i++) {</pre>
28
                     c = num.charAt(i);
29
                     imageNumber = changeNumber(c);
30
                     x = 1000 - ((num.length() - i) * 50);
31
                     g.drawImage(imageNumber, x, 0, null);
32
                 }
33
             }
34
             catch(IOException e) {}
35
             g.drawImage(img, x - img.getWidth() - (50 * (num.length() - 1)), 0, null);
36
37
38
39
    }
```

```
import java.awt.*;
    import java.awt.event.*;
2
    import java.awt.image.*;
3
    import java.io.*;
4
    import java.util.*;
5
    import javax.imageio.*;
6
    import javax.swing.*;
7
8
    public class LoseScreen {
9
        private Image youLose;
10
        private Image playAgain;
11
        private Image screwThisGame;
12
        private Image dot;
13
        private int doty = 200;
14
15
        public LoseScreen() {
16
             try {
17
                 youLose = ImageIO.read(new File("you lose.png"));
18
                 playAgain = ImageIO.read(new File("play again.png"));
19
                 screwThisGame = ImageIO.read(new File("screw this game.png"));
20
                 dot = ImageIO.read(new File("dot.png"));
21
             }
22
             catch(IOException e) {
23
24
25
             }
        }
26
27
         public void dotDown() {
28
             doty = 250;
29
30
31
        public void dotUp() {
32
             doty = 200;
33
        }
34
35
        public int getDot() {
36
             return doty;
37
        }
38
39
         public void draw(Graphics g) {
40
             g.drawImage(dot, 435, doty, null);
41
             g.drawImage(youLose, 400, 100, null);
42
             g.drawImage(playAgain, 450, 200, null);
43
             g.drawImage(screwThisGame, 450, 250, null);
44
         }
45
    }
```

```
import java.awt.*;
    import java.awt.event.*;
2
    import java.awt.image.*;
3
    import java.io.*;
4
    import java.util.*;
5
     import javax.imageio.*;
6
     import javax.swing.*;
7
8
    public class Obstacle {
9
         private static int enemySprite;
10
11
         private int xCoord;
12
         private int yCoord;
13
         private double velocity;
14
         private int width;
15
         private int height;
16
17
         private BufferedImage img;
18
19
         public Obstacle(int x, int y, double v) {
20
             velocity = v;
21
             xCoord = x;
22
             yCoord = y;
23
             try {
24
                 enemySprite = (int)(Math.random() * 5);
25
                 switch(enemySprite) {
26
                     case 0:
27
                          img = ImageIO.read(new File("angryface.png"));
28
                         break;
29
                     case 1:
30
                         img = ImageIO.read(new File("apple.png"));
31
                         break;
32
                     case 2:
33
                         img = ImageIO.read(new File("toxic.png"));
34
                         break;
35
                     case 3:
36
                         img = ImageIO.read(new File("star.png"));
37
38
                     case 4:
39
                         img = ImageIO.read(new File("recyclebin.png"));
40
                         break;
41
                 }
42
             }
43
             catch(IOException e) {}
44
             width = img.getWidth();
45
             height = img.getHeight();
46
         }
47
48
         public int getXCoord() {
49
             return xCoord;
50
         }
51
52
         public int getYCoord() {
53
```

```
54
              return yCoord;
 55
          }
 56
 57
          public int getWidth() {
              return width;
 58
 59
 60
 61
          public int getHeight() {
              return height;
 62
 63
 64
          public void scroll() {
 65
 66
              xCoord -= velocity;
 67
          }
 68
          public double getV() {
 69
 70
              return velocity;
 71
          }
 72
 73
          public void updateV(double x) {
 74
              velocity = x;
 75
          }
 76
 77
          public void draw(Graphics g) {
 78
              g.drawImage(img, xCoord, yCoord, null);
 79
 80
 81
          public void drawDeath(Graphics g) {
 82
              try {
 83
                  switch(enemySprite) {
 84
                      case 0:
 85
                          img = ImageIO.read(new File("angryangryface.png"));
                          break;
 86
 87
                      case 1:
 88
                          img = ImageIO.read(new File("angryapple.png"));
                          break;
 89
 90
                      case 2:
 91
                          img = ImageIO.read(new File("angrytoxic.png"));
 92
                          break;
 93
                      case 3:
 94
                           img = ImageIO.read(new File("angrystar.png"));
 95
                          break;
 96
                      case 4:
                          img = ImageIO.read(new File("angryrecyclebin.png"));
 97
 98
                          break;
 99
                  }
100
                  draw(g);
101
              }
102
              catch(IOException e) {}
103
              width = img.getWidth();
              height = img.getHeight();
104
105
106
107
          public String toString() {
108
              return "Obstacle-> X: " + xCoord + "\tY: " + yCoord;
109
```

110 | }

```
import javax.swing.*;
    import java.awt.image.*;
2
    import java.awt.*;
3
    import java.io.*;
4
    import javax.imageio.*;
5
     public class Player {
 6
         private BufferedImage img;
7
         private int xCoord;
8
         private int yCoord;
9
         private int width;
10
         private int height;
11
         private int sprite;
12
         private double velocity;
13
14
         public Player(int x, int y) {
15
             xCoord = x;
16
             yCoord = y;
17
             try {
18
                 img = ImageIO.read(new File("cat.png"));
19
             }
20
             catch(IOException e) {
21
                 System.out.println("Failed to load image");
22
             }
23
             width = img.getWidth();
24
25
             height = img.getHeight();
26
         public void jump() {
27
             if(yCoord == 300)
28
                 velocity = -10;
29
30
         public void gravity() {
31
             sprite++;
32
             velocity += 0.378;
33
             yCoord += (velocity + 0.0072);
34
             if(yCoord > 300) {
35
                 yCoord = 300;
36
             }
37
         }
38
         public boolean collision(int x, int y, int w, int h) {
39
             boolean left = x > xCoord;
40
             boolean right = x < xCoord + width;</pre>
41
             boolean up = y < yCoord + height;</pre>
42
             boolean down = y + h > yCoord;
43
             return left && right && down && up;
44
         public int getXCoord() {
46
             return xCoord;
47
48
         public int getYCoord() {
49
             return yCoord;
50
51
         public void draw(Graphics g) {
52
             try {
53
```

```
if(Stage.lose)
54
55
                     img = ImageIO.read(new File("deadcat.png"));
56
                else if(sprite % 3 == 0)
                     img = ImageIO.read(new File("cat.png"));
57
58
                 else
                     img = ImageIO.read(new File("cat2.png"));
59
60
            }
            catch(IOException e) {}
61
            g.drawImage(img, xCoord, yCoord, null);
62
63
64
        public String toString() {
            return "Player-> X: " + xCoord + "\tY: " + yCoord;
65
66
        }
67
   }
```

```
import java.awt.*;
1
    import java.awt.event.*;
2
    import java.awt.image.*;
3
    import java.io.*;
4
    import java.util.*;
5
    import javax.imageio.*;
6
    import javax.swing.*;
7
8
    public class Score {
9
        private Image score;
10
        private Image imageNumber;
11
12
         public Score() {
13
             try {
14
                 score = ImageIO.read(new File("score.png"));
15
16
             catch(IOException e) {}
17
        }
18
19
         public Image changeNumber(char c) throws IOException {
20
             Image number = null;
21
             switch(c) {
22
                 case '0':
23
                     number = ImageIO.read(new File("zero.png"));
24
25
                     break;
                 case '1':
26
                     number = ImageIO.read(new File("one.png"));
27
                     break;
28
                 case '2':
29
                     number = ImageIO.read(new File("two.png"));
30
                     break;
31
                 case '3':
32
                     number = ImageIO.read(new File("three.png"));
33
                     break;
34
                 case '4':
35
                     number = ImageIO.read(new File("four.png"));
36
                     break;
37
                 case '5':
38
                     number = ImageIO.read(new File("five.png"));
39
                     break;
40
                 case '6':
41
                     number = ImageIO.read(new File("six.png"));
42
                     break;
43
                 case '7':
44
                     number = ImageIO.read(new File("seven.png"));
45
                     break;
46
                 case '8':
47
                     number = ImageIO.read(new File("eight.png"));
48
                     break;
49
                 case '9':
50
                     number = ImageIO.read(new File("nine.png"));
51
                     break;
52
             }
53
```

```
54
             return number;
55
         }
56
57
         public void draw(Graphics g, int s) {
58
             g.drawImage(score, 0, 0, null);
59
             Integer inte = new Integer(s);
             String num = inte.toString();
60
61
             char c;
62
             try {
                 for(int i = 0; i < num.length(); i++) {</pre>
63
64
                     c = num.charAt(i);
                     imageNumber = changeNumber(c);
65
66
                     g.drawImage(imageNumber, i * 50 + 75, 0, null);
67
68
             }
69
             catch(IOException e) {}
70
        }
71
    }
```

```
import java.awt.*;
    import java.awt.event.*;
 2
    import java.awt.image.*;
 3
    import java.io.*;
 4
    import java.util.*;
 5
     import javax.imageio.*;
 6
     import javax.swing.*;
 7
 8
    public class Shrub {
 9
10
         private BufferedImage img;
11
         private int xCoord;
12
         private int yCoord;
13
         private int width;
14
         private int height;
15
         private double velocity;
16
17
         public Shrub(int x, int y, BufferedImage i) {
18
19
             img = i;
             xCoord = x;
20
             yCoord = y;
21
             width = i.getWidth();
22
             height = i.getHeight();
23
         }
24
25
         public int getXCoord() {
26
             return xCoord;
27
         }
28
29
         public int getYCoord() {
30
             return yCoord;
31
32
33
         public int getWidth() {
34
             return width;
35
         }
36
37
         public int getHeight() {
38
             return height;
39
40
41
         public void scroll() {
42
             xCoord -= velocity;
43
         }
44
45
         public void updateV(double v) {
46
             velocity = v;
47
         }
48
49
         public void draw(Graphics g) {
50
             g.drawImage(img, xCoord, yCoord, null);
51
         }
52
    }
53
```

```
//imports
    import java.awt.*;
2
    import java.awt.event.*;
3
    import java.awt.image.*;
4
    import java.io.*;
5
    import java.util.*;
6
    import javax.swing.*;
7
8
    public class Stage extends JPanel implements KeyListener, ActionListener {
9
10
        //stage height and width
11
         private static final int STAGE_HEIGHT = 500;
12
         private static final int STAGE_WIDTH = 1000;
13
14
         //private
15
        private File dat;
16
         private javax.swing.Timer timer;
17
         private Player player;
18
19
        private Floor floor;
         private ArrayList<Obstacle> obstacles;
20
        private LoseScreen losescreen;
21
        private Score scoreboard;
22
        private Highscore highscore;
23
        private int score;
24
25
        private int high_score;
         private Background background;
26
         private double velocity;
27
28
         //lose varriable
29
         public static boolean lose;
30
31
32
         public Stage(int width, int height) {
33
             setPreferredSize(new Dimension(width, height));
34
             player = new Player(50, 310);
35
            floor = new Floor(0, STAGE_HEIGHT - 100);
36
            timer = new javax.swing.Timer(20, this);
37
38
            losescreen = new LoseScreen();
            scoreboard = new Score();
39
            initializeDataFile();
40
            highscore = new Highscore(high_score);
41
            background = new Background();
42
            obstacles = new ArrayList<>();
43
            obstacles.add(new Obstacle(800, 290, 7));
44
            this.setFocusable(true);
45
             addKeyListener(this);
46
            lose = false;
47
            timer.start();
48
        }
49
50
         public void initializeDataFile() {
51
            try {
52
                 dat = new File("info.dat");
53
```

```
Scanner scan = new Scanner(dat);
 54
 55
                  high_score = Integer.valueOf(scan.nextLine());
 56
              }
 57
              catch(IOException e) {}
          }
 58
 59
 60
          public void closeDataFile() {
 61
              try {
                  PrintWriter pw = new PrintWriter(dat);
 62
 63
                  pw.println(score);
 64
                  pw.close();
              }
 65
 66
              catch(IOException e) {}
 67
          }
 68
 69
          public void actionPerformed(ActionEvent e) {
 70
              player.gravity();
 71
              status();
 72
              if(!lose) {
 73
                  for(Obstacle o : obstacles) {
 74
                      o.scroll();
 75
                      velocity = o.getV() + (score * 0.01);
76
                      o.updateV(velocity);
 77
 78
                  background.scroll(velocity);
 79
 80
              else {
 81
                   timer.stop();
 82
              }
              repaint();
 84
          }
 85
 86
          public void keyPressed(KeyEvent e) {
 87
              if(lose) {
 88
                  switch(e.getKeyCode()) {
 89
                       case KeyEvent.VK_UP:
 90
                           losescreen.dotUp();
 91
                           repaint();
 92
                           break;
                       case KeyEvent.VK_DOWN:
 93
 94
                           losescreen.dotDown();
                           repaint();
 96
                           break;
                       case KeyEvent.VK_ENTER:
 97
 98
                           if(losescreen.getDot() - 200 == 0) {
 99
                               start();
100
                               lose = false;
101
                           }
102
                           else {
103
                               System.exit(0);
104
                           }
105
                           break;
106
                  }
107
              }
108
              else if(e.getKeyCode() == KeyEvent.VK_SPACE) {
109
                  player.jump();
```

```
110
              }
111
          }
112
113
          public void keyTyped(KeyEvent e) {}
114
          public void keyReleased(KeyEvent e) {}
115
116
          public void paintComponent(Graphics g) {
117
              super.paintComponent(g);
118
119
              background.draw(g);
120
              scoreboard.draw(g, score);
121
              highscore.draw(g);
122
              player.draw(g);
123
              floor.draw(g);
              for(Obstacle o : obstacles) {
124
125
                  o.draw(g);
126
              }
127
              if(lose) {
128
                  for(Obstacle o : obstacles) {
129
                      o.drawDeath(g);
130
131
                  losescreen.draw(g);
              }
132
133
          }
134
          public void status() {
135
              for(Obstacle o : obstacles) {
136
137
                  //System.out.print(player + "\t" + o + "\r");
138
                  if(player.collision(o.getXCoord(), o.getYCoord(), o.getWidth(), o.getHeight())) {
139
                       lose = true;
140
                       if(score > high_score) {
141
                           closeDataFile();
142
                      }
143
                       repaint();
                  }
144
              }
145
146
147
              if(obstacles.get(0).getXCoord() <= -70) {</pre>
148
                  int y = ((int)(Math.random() * 10) % 2) * 100 + 200;
149
                  obstacles.add(new Obstacle(1000, y, 7));
150
                  obstacles.remove(0);
151
                  score++;
152
              }
153
          }
154
155
          public void start() {
156
              score = 0;
              player = new Player(50, 310);
157
              floor = new Floor(0, STAGE_HEIGHT - 100);
158
              timer = new javax.swing.Timer(20, this);
159
160
              losescreen = new LoseScreen();
161
              scoreboard = new Score();
              initializeDataFile();
162
163
              highscore = new Highscore(high_score);
164
              obstacles = new ArrayList<>();
165
              obstacles.add(new Obstacle(800, 290, 7));
```

```
166
             this.setFocusable(true);
167
             addKeyListener(this);
             lose = false;
168
169
             timer.start();
170
         }
171
         public static void main(String args[]) {
172
             JFrame frame = new JFrame();
173
             frame.setTitle("Cat run");
174
             frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
175
176
             JPanel panel = new Stage(STAGE_WIDTH, STAGE_HEIGHT);
             Container c = frame.getContentPane();
177
178
             c.add(panel);
179
             frame.pack();
180
             frame.setVisible(true);
181
             frame.setResizable(false);
182
         }
183
     }
```