```java
package GreenbergSnake1;
import edu.du.dudraw.*;

public class Driver
{
   public static void main(String[] args)
   {
       System.out.println("Use the arrows on your keyboard or the letters WASD to
control your snake. Do not intersect the walls or the body of your snake.");
       Game g1= new Game();
   }
}
```

```java
package GreenbergSnake1;
import java.util.Random;

import edu.du.dudraw.Draw;

public class Food
{
   private int xF;
   private int yF;
   private boolean spawn;
   private boolean eat;

   public Food()
   {
       Random random = new Random();
       xF= (int)(Math.random() * 45);
       yF= (int)(Math.random() * 45);
   }

   public void spawn()
   {
       Random random = new Random();
       xF= (int)(Math.random() * 45);
       yF= (int)(Math.random() * 45);
   }

   public int getxF()
   {
```

```
        return this.xF;
    }

    public int getyF()
    {
        return this.yF;
    }

    public void draw(Draw d)
    {
        d.setPenColor(d.ORANGE);
        d.filledRectangle(this.getxF(), this.getyF(), 0.5, 0.5);
    }
}
```

```
package GreenbergSnake1;
import javax.xml.parsers.SAXParser;

import edu.du.dudraw.*;

public class Game implements DrawListener
{
    private Snake s;
    private boolean gameOver;
    private int score;
    private Food f;
    private Draw canvas;

    public Game()
    {
        this.canvas= new Draw();
        this.s= new Snake();
        this.f= new Food();
        this.score= 0;
        canvas.startUpdate();
        canvas.setCanvasSize(500, 500);
        canvas.setXscale(0, 50);
        canvas.setYscale(0, 50);
        canvas.enableDoubleBuffering();
        canvas.addListener(this);
```

```
    }

    public void drawStart()
    {
        canvas.text(10, 48, "Score: " + score);

    }

    public void update()
    {
        canvas.clear();
        drawStart();
        if(gameOver==false)
        {
            s.draw(canvas);
            f.draw(canvas);
            s.move();
            if(s.eat(f)==true)
            {
                score++;
                s.add();
                f.spawn();
                update();
            }
            if(s.intersectSnake() || s.intersectWall())
            {
                gameOver= true;
                System.out.println("Game over. Press tab to restart (sometimes game
requires tab to be pressed 2x).");
                canvas.clear();
            }
        }
        canvas.show();
    }




    @Override
    public void keyPressed(int arg0)
    {
        //System.out.println(arg0);
```

```
        if(arg0 == 87 || arg0== 38)
        {
            s.setDirection(0);
        }

        if (arg0 == 65 || arg0== 37)
        {
            s.setDirection(3);
        }

        if(arg0 == 83 || arg0== 40)
        {
            s.setDirection(2);
        }

        if (arg0 == 68 || arg0== 39)
        {
            s.setDirection(1);
        }

        if(arg0 == 9)
        {
            reset();
        }
    }

    public void reset()
    {
        this.s= new Snake();
        this.f= new Food();
        this.score= 0;
        this.gameOver= false;
    }

    @Override
    public void keyReleased(int arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void keyTyped(char arg0) {
```

```
        // TODO Auto-generated method stub

    }

    @Override
    public void mouseClicked(double arg0, double arg1) {
        // TODO Auto-generated method stub

    }

    @Override
    public void mouseDragged(double arg0, double arg1) {
        // TODO Auto-generated method stub

    }

    @Override
    public void mousePressed(double arg0, double arg1) {
        // TODO Auto-generated method stub

    }

    @Override
    public void mouseReleased(double arg0, double arg1) {
        // TODO Auto-generated method stub

    }



}
```

```
package GreenbergSnake1;

public class Segment
{
    private double x;
    private double y;

    public Segment()
```

```
    {
        x= 0;
        y= 0;
    }

    public Segment(double xVal, double yVal)
    {
        x= xVal;
        y= yVal;
    }

    public double getX()
    {
        return this.x;
    }

    public double getY()
    {
        return this.y;
    }

    public void setX(double xVal)
    {
        x= xVal;
    }

    public void setY(double yVal)
    {
        y= yVal;
    }

    public Segment prev() {
        // TODO Auto-generated method stub
        return null;
    }

    public Segment next() {
        // TODO Auto-generated method stub
        return null;
    }
}
```

```java
package GreenbergSnake1;
import java.util.LinkedList;

import edu.du.dudraw.Draw;

public class Snake
{
   LinkedList<Segment> body;

   //Data members
   private int size;
   private Segment head;
   private int direction; //0= north, 1= east, 2= south, 3= west

   //Constructor creates an empty list
   public Snake()
   {
       this.body= new LinkedList<Segment>();
       this.head= new Segment(25, 25);
       size= body.size();
       this.direction= 0;
   }
   public boolean intersectWall()
   {
       if((getHead().getX()<0 || getHead().getX()>50) || (getHead().getY()<0 ||
getHead().getY()>50))
       {
           return true;
       }
       return false;
   }

   public boolean intersectSnake()
   {
       for(int i=0; i<body.size(); i++)
       {
           if(head.getX()==body.get(i).getX() && head.getY()==body.get(i).getY())
           {
               return true;
           }
       }
```

```
    return false;
}

public boolean eat(Food f)
{
   if((this.head.getX()== f.getxF() && this.head.getY()== f.getyF()))
   {
       return true;
   }
   return false;
}

public int getSize()
{
    return size;
}

public Segment getHead()
{
    return this.head;
}

public int getDirection()
{
    return this.direction;
}

public void setDirection(int i)
{
    direction= i;
}

public void add()
{
    Segment newseg= new Segment();
    body.addLast(newseg);
}

public void draw(Draw d)
{
    d.setPenColor(d.GREEN);
    for (int i = 0; i < body.size(); i++)
```

```
        {
            d.filledRectangle(this.body.get(i).getX(), this.body.get(i).getY(), 0.5,
0.5);
        }
        d.setPenColor(d.RED);
        d.filledRectangle(head.getX(), head.getY(), 0.5, 0.5);
    }

    public void move()
    {
        if(body.size()>0)
        {
            for(int i=body.size()-1; i>0; i--)
            {
                Segment current= body.get(i);
                Segment previous= body.get(i-1);
                current.setY(previous.getY());
                current.setX(previous.getX());
            }
            body.get(0).setX(head.getX());
            body.get(0).setY(head.getY());
        }

            if(direction==0)
            {
                head.setY(head.getY()+1);
            }
            if(direction==2)
            {
                head.setY(head.getY()-1);
            }

            if(direction==1)
            {
                head.setX(head.getX()+1);
            }
            if(direction==3)
            {
                head.setX(head.getX()-1);
            }
    }
}
```