```java
import greenfoot.*;  // (World, Actor, GreenfootImage, and Greenfoot)

/**
 * A Greep is an alien creature that likes to collect tomatoes.
 *
 * @author (your name here)
 * @version 0.1
 */
public class Greep extends Creature
{
    // Remember: you cannot extend the Greep's memory. So:
    // no additional fields (other than final fields) allowed in this class!

    /**
     * Default constructor for testing purposes.
     */
    public Greep()
    {
        this(null);
    }


    /**
     * Create a Greep with its home space ship.
     */
    public Greep(Ship ship)
    {
        super(ship);
    }


    /**
     * Do what a greep's gotta do.
     */
    public void act()
    {
        super.act();   // do not delete! leave as first statement in act().

        if( getFlag(1) && getFlag(2) ) { // DELIVER ACTION
            this.deliver();
        } else if( getFlag(1) ) { // WAIT ACTION
            this.waiter();
        } else if( getFlag(2) ) { // SEARCH ACTION
            if( this.search() ) {
                if( !super.seePaint("red") ) {
                    super.spit("red");
                    this.waiter();
                } else {
                    super.loadTomato();
                }
            }
        } else { // ONLY WORKS FIRST TIME AROUND
            if( !super.carryingTomato() ) {
                if( this.search() ) {
                    if( !super.seePaint("red") ) {
                        super.spit("red");
                        this.waiter();
                    } else {
                        super.loadTomato();
                    }
                }
            }
        }
    }

    private boolean search() {
```

```
67            super.setFlag(1, false);
68            super.setFlag(2, true);
69
70            this.checkEdge();
71            this.checkWater();
72            //this.checkShip(0);
73            if( this.checkBreadcrumbs() )
74                this.followCrumbs();
75            super.move();
76            return (TomatoPile) getOneIntersectingObject(TomatoPile.class) != null;
77        }
78
79        private boolean checkWater() {
80            if( super.atWater() ) { // turn if at waters edge
81                super.turn(180);
82                super.move();
83                return true;
84            }
85            return false;
86        }
87
88        private boolean checkEdge() {
89            if( super.atWorldEdge() ) { // turn if at edge of map
90                super.turn(90);
91                super.move();
92                return true;
93            }
94            return false;
95        }
96
97        private boolean checkShip(int i) {
98            if( super.atShip() ) { // if its at the ship and carrying a tomato then drop it
              and turn 180 else just turn 180 then move
99                if(i == 1)
100                   super.dropTomato();
101               super.turn(180);
102               super.move();
103               return true;
104           }
105           return false;
106       }
107
108       private boolean checkBreadcrumbs() {
109           return true;
110       }
111
112       private void followCrumbs() {
113
114       }
115
116       private void waiter() {
117           super.setFlag(1, true);
118           super.setFlag(2, false);
119
120           if( super.carryingTomato() )
121               this.deliver();
122       }
123
124       private void deliver() {
125           super.setFlag(1, true);
126           super.setFlag(2, true);
127
128           super.spit("orange");
129           super.turnHome();
130           super.move();
131
```

```
132             if(super.atShip()) {
133                 this.checkShip(1);
134                 this.reset();
135             }
136         }
137 
138         private void reset() {
139             super.setMemory(0);
140             super.setFlag(1, false);
141             super.setFlag(2, false);
142         }
143 
144         /**
145          * Is there any food here where we are? If so, try to load some!
146          */
147         public void checkFood()
148         {
149             // check whether there's a tomato pile here
150             TomatoPile tomatoes = (TomatoPile) getOneIntersectingObject(TomatoPile.class);
151             if(tomatoes != null) {
152                 loadTomato();
153                 // Note: this attempts to load a tomato onto *another* Greep. It won't
154                 // do anything if we are alone here.
155             }
156         }
157 
158 
159         /**
160          * This method specifies the name of the author (for display on the result board).
161          */
162         public static String getAuthorName()
163         {
164             return "Aryan Gupta";  // write your name here!
165         }
166 
167 
168         /**
169          * This method specifies the image we want displayed at any time. (No need
170          * to change this for the competition.)
171          */
172         public String getCurrentImage()
173         {
174             if(carryingTomato())
175                 return "greep-with-food.png";
176             else
177                 return "greep.png";
178         }
179     }
```