Set2

1. What is the role of the instance variable sideLength?

   Since the boxBug traces out a square box, sideLength is the length of the side of the square, which is also the maximum step that a bug can move on one side.

2. What is the role of the instance variable steps?

   Steps records how many steps the bug has finished on one particular side, if steps is equal to sideLength, the bug will no longer move forward, instead it will begin to turn.

3. Why is the turn method called twice when steps becomes equal to sideLength?

   For each turn, the bug turns right by 45 degree. In order to turn 90 degree, the bug has to turn twice.

4. Why can the move method be called in the BoxBug class when there is no move method in the BoxBug code?

   Because class BoxBug extends class Bug, and in class Bug there is a method called move(), so BoxBug inherits that method.

5. After a BoxBug is constructed, will the size of its square pattern always be the same? Why or why not?

   Yes. There is no method that can change the sideLength, so once a bug is constructed, its walking pattern is also determined.

6. Can the path a BoxBug travels ever change? Why or why not?

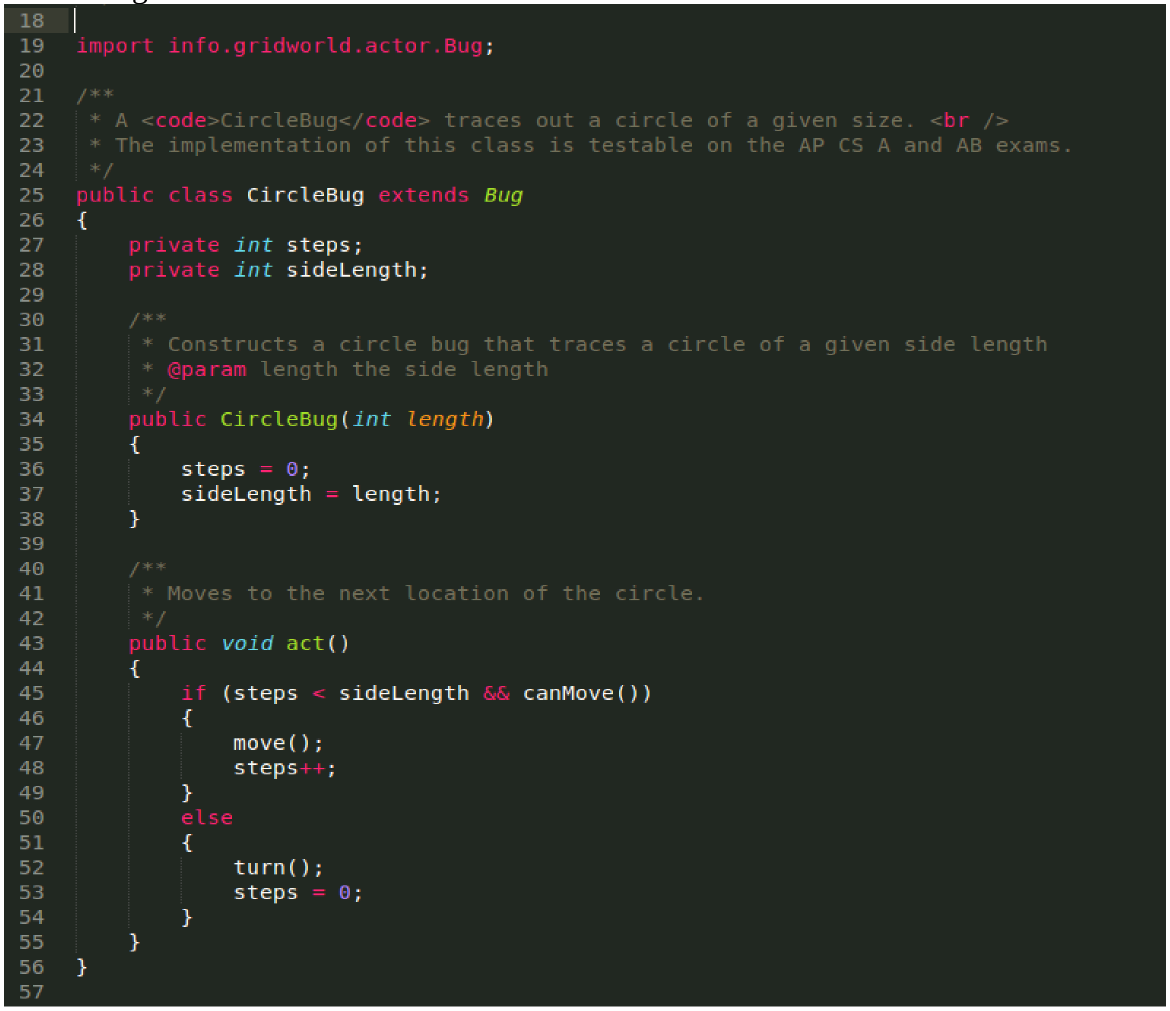
   Yes. If there is a barrier in front of the bug, say a rock, the bug will try to turn instead of moving forward. Under such circumstance, the course of the bug has been changed.

7. When will the value of steps be zero?

   Firstly, when the bug is constructed, its steps is zero; secondly, when the bug faces a wall and it cannot move, its steps is zero too; thirdly, when the number of steps the bug has walked equals sideLength, its steps is also reset to zero.

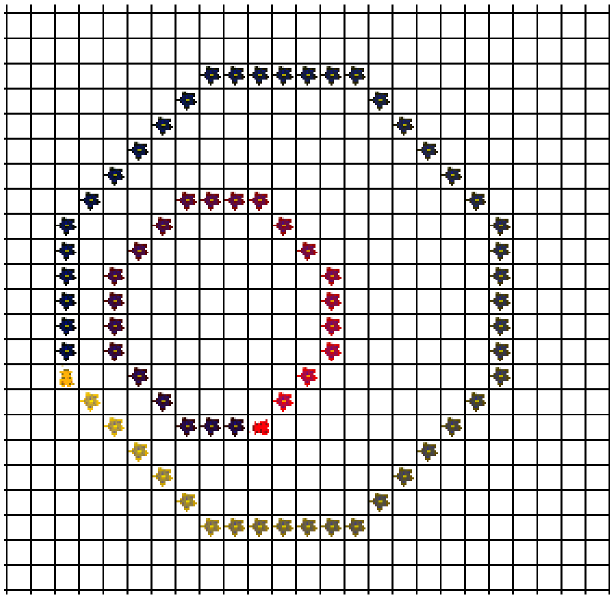
Exercises

1. CircleBug

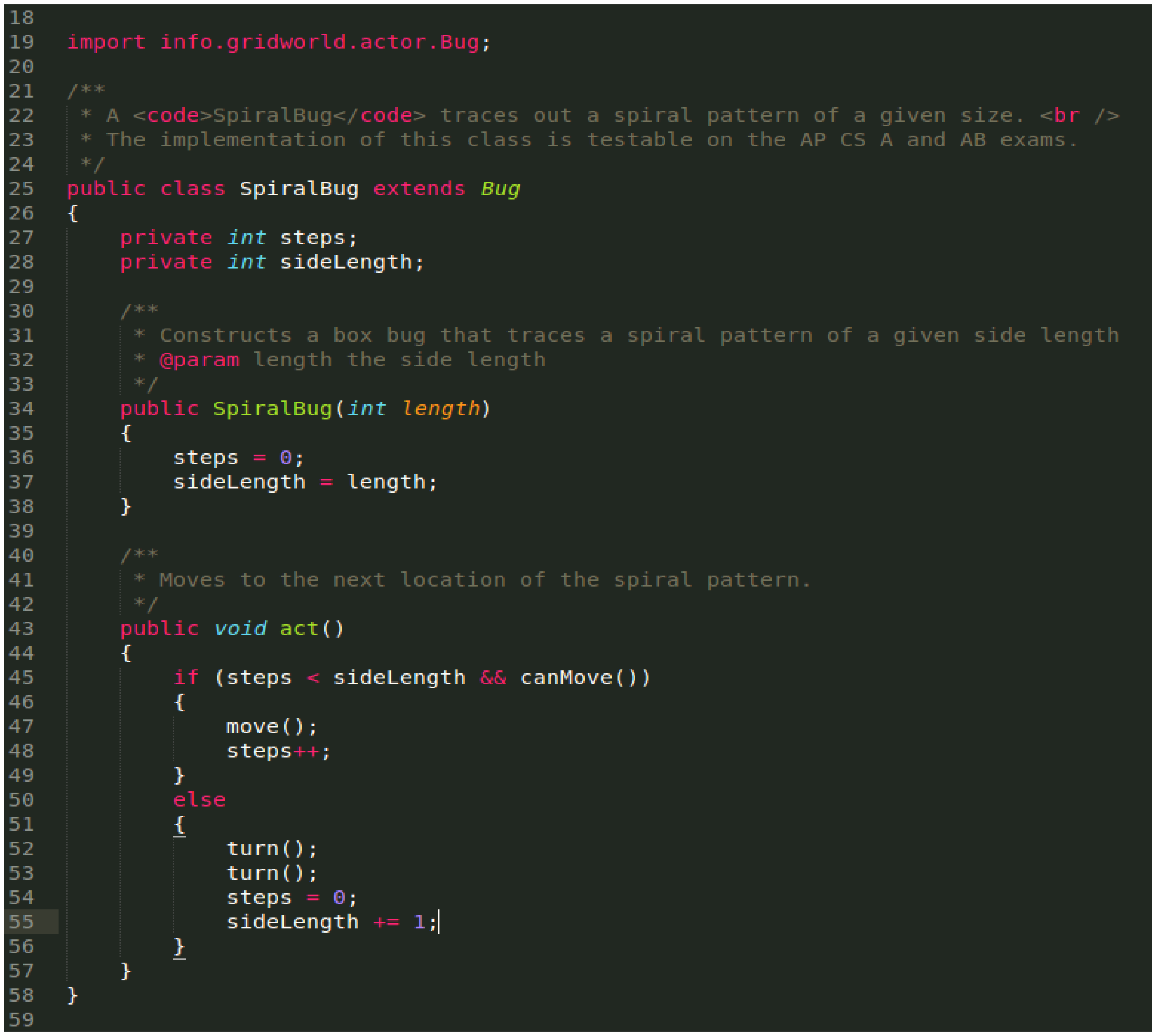


```java

import info.gridworld.actor.Bug;

/**
 * A <code>CircleBug</code> traces out a circle of a given size. <br />
 * The implementation of this class is testable on the AP CS A and AB exams.
 */
public class CircleBug extends Bug
{
    private int steps;
    private int sideLength;

    /**
     * Constructs a circle bug that traces a circle of a given side length
     * @param length the side length
     */
    public CircleBug(int length)
    {
        steps = 0;
        sideLength = length;
    }

    /**
     * Moves to the next location of the circle.
     */
    public void act()
    {
        if (steps < sideLength && canMove())
        {
            move();
            steps++;
        }
        else
        {
            turn();
            steps = 0;
        }
    }
}

```

CircleBug covers the route of an octagon instead of a square if there is no barrier in front of it.

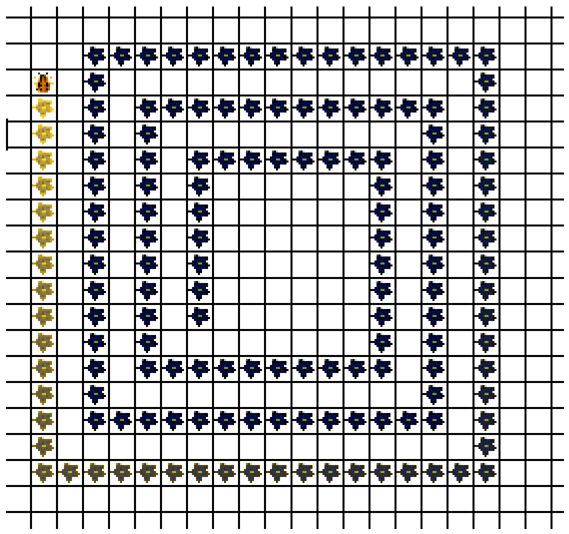


2. SpiralBug

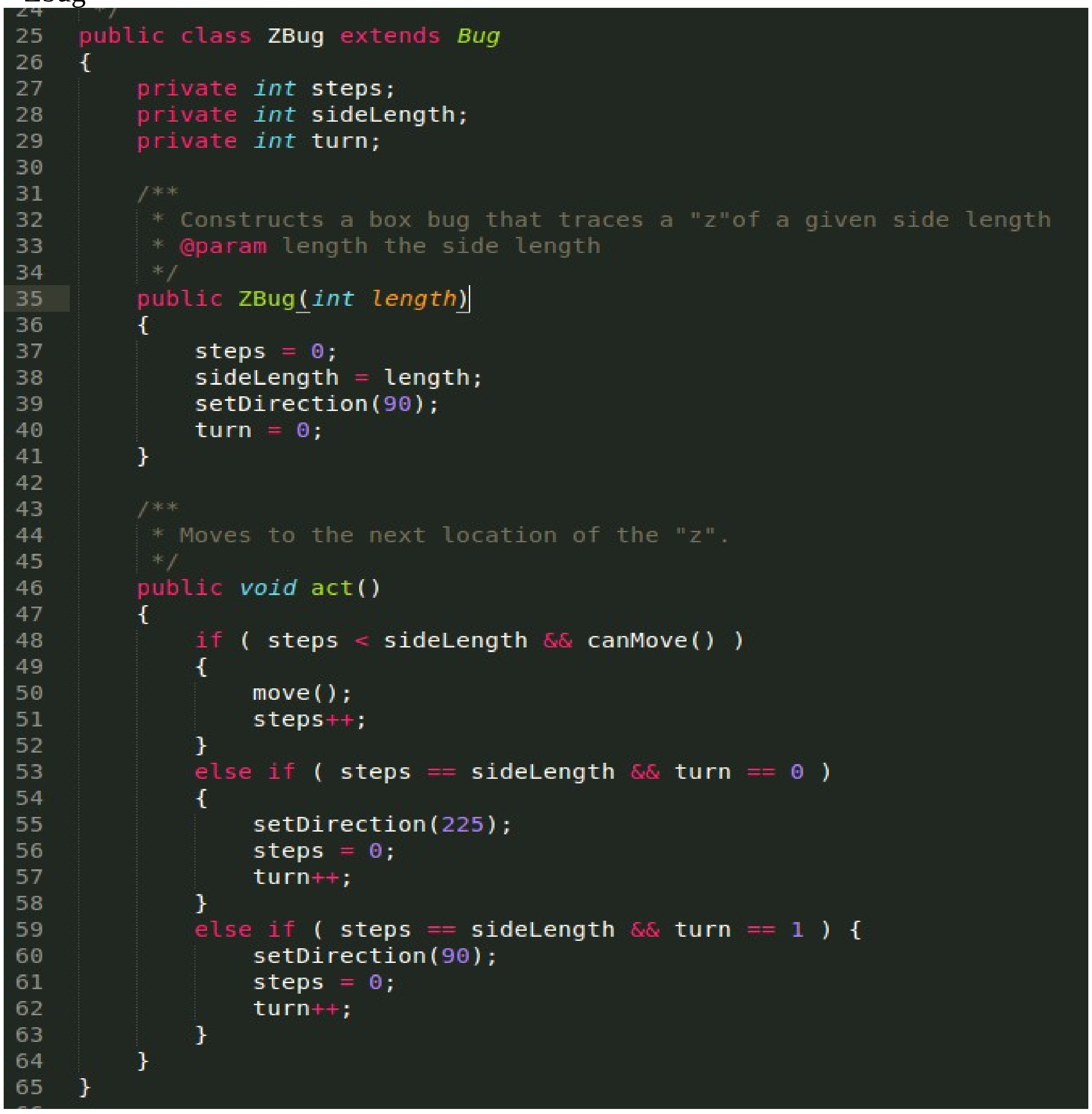


```
import info.gridworld.actor.Bug;

/**
 * A <code>SpiralBug</code> traces out a spiral pattern of a given size. <br />
 * The implementation of this class is testable on the AP CS A and AB exams.
 */
public class SpiralBug extends Bug
{
    private int steps;
    private int sideLength;

    /**
     * Constructs a box bug that traces a spiral pattern of a given side length
     * @param length the side length
     */
    public SpiralBug(int length)
    {
        steps = 0;
        sideLength = length;
    }

    /**
     * Moves to the next location of the spiral pattern.
     */
    public void act()
    {
        if (steps < sideLength && canMove())
        {
            move();
            steps++;
        }
        else
        {
            turn();
            turn();
            steps = 0;
            sideLength += 1;
        }
    }
}
```

Pattern:

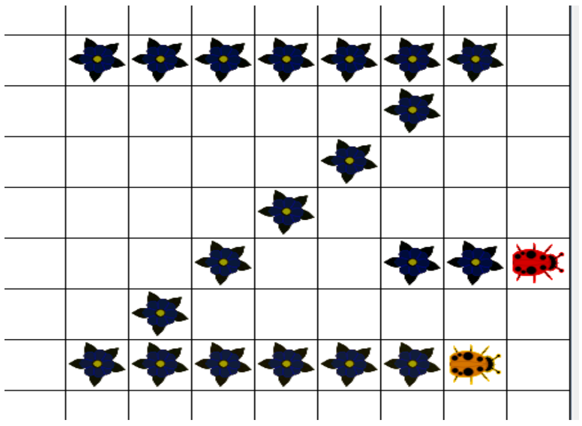


3. Zbug

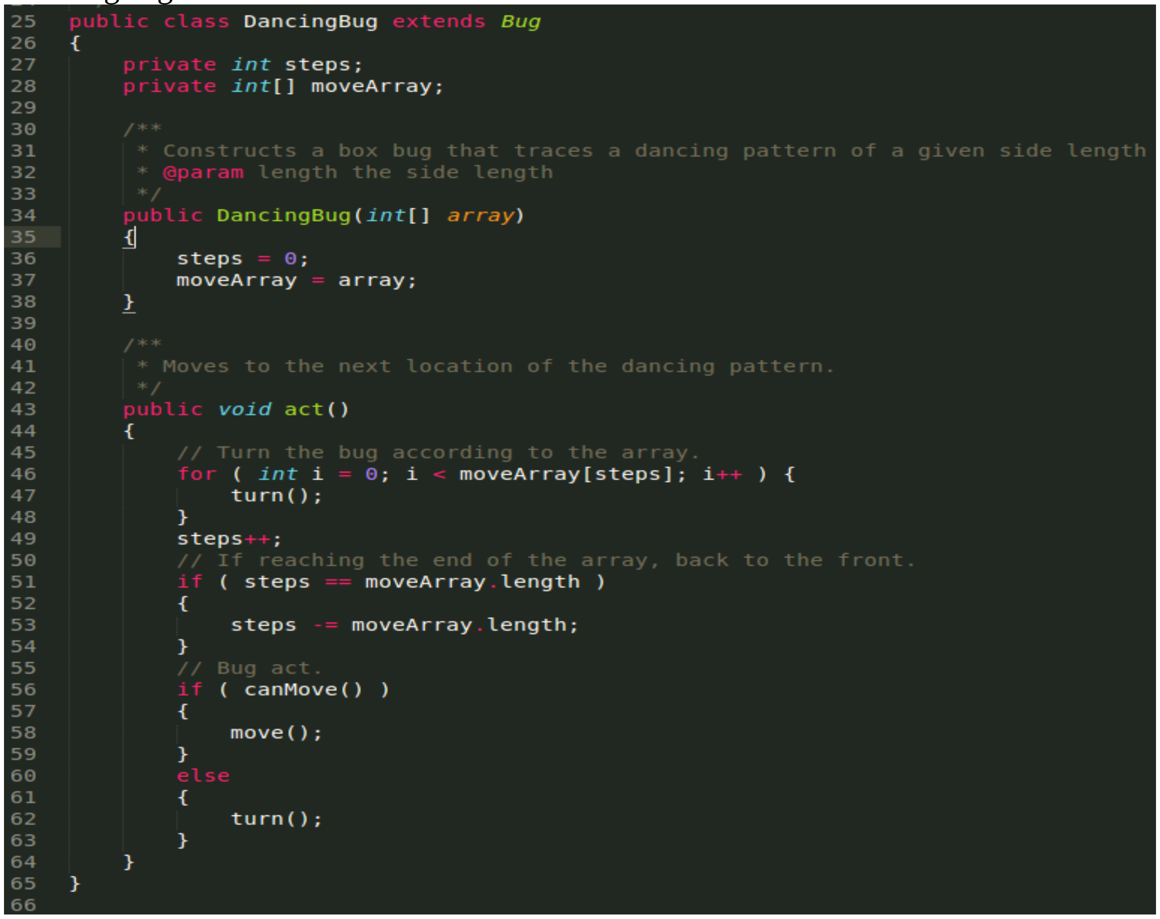


```java
     */
public class ZBug extends Bug
{
    private int steps;
    private int sideLength;
    private int turn;

    /**
     * Constructs a box bug that traces a "z"of a given side length
     * @param length the side length
     */
    public ZBug(int length)
    {
        steps = 0;
        sideLength = length;
        setDirection(90);
        turn = 0;
    }

    /**
     * Moves to the next location of the "z".
     */
    public void act()
    {
        if ( steps < sideLength && canMove() )
        {
            move();
            steps++;
        }
        else if ( steps == sideLength && turn == 0 )
        {
            setDirection(225);
            steps = 0;
            turn++;
        }
        else if ( steps == sideLength && turn == 1 ) {
            setDirection(90);
            steps = 0;
            turn++;
        }
    }
}
```

Pattern:

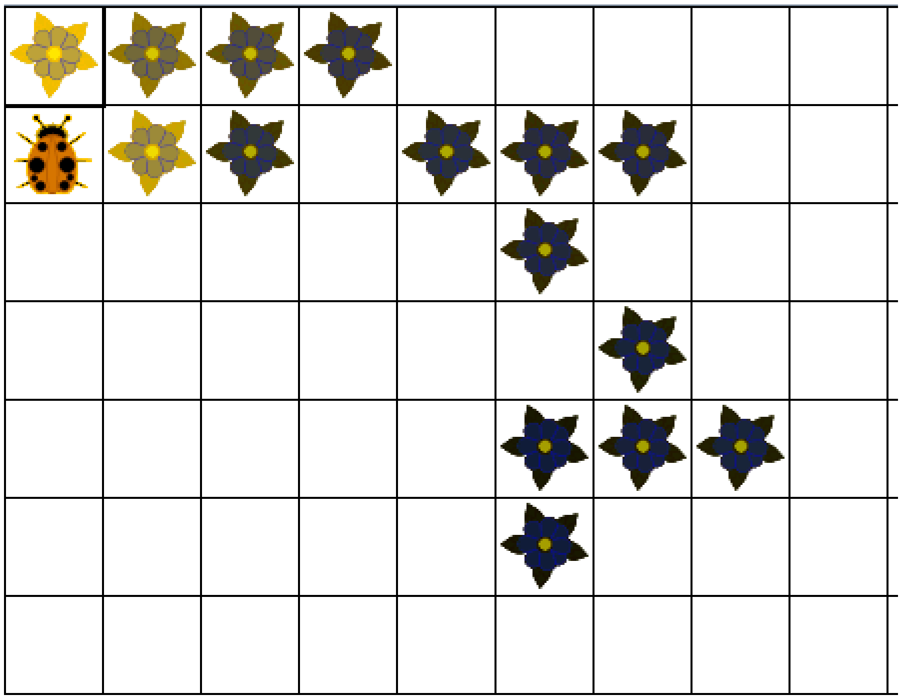


4. DancingBug



```java
public class DancingBug extends Bug
{
    private int steps;
    private int[] moveArray;

    /**
     * Constructs a box bug that traces a dancing pattern of a given side length
     * @param length the side length
     */
    public DancingBug(int[] array)
    {
        steps = 0;
        moveArray = array;
    }

    /**
     * Moves to the next location of the dancing pattern.
     */
    public void act()
    {
        // Turn the bug according to the array.
        for ( int i = 0; i < moveArray[steps]; i++ ) {
            turn();
        }
        steps++;
        // If reaching the end of the array, back to the front.
        if ( steps == moveArray.length )
        {
            steps -= moveArray.length;
        }
        // Bug act.
        if ( canMove() )
        {
            move();
        }
        else
        {
            turn();
        }
    }
}
```

Pattern: (moveArray = {0, 1, 2, 3, 4, 5})



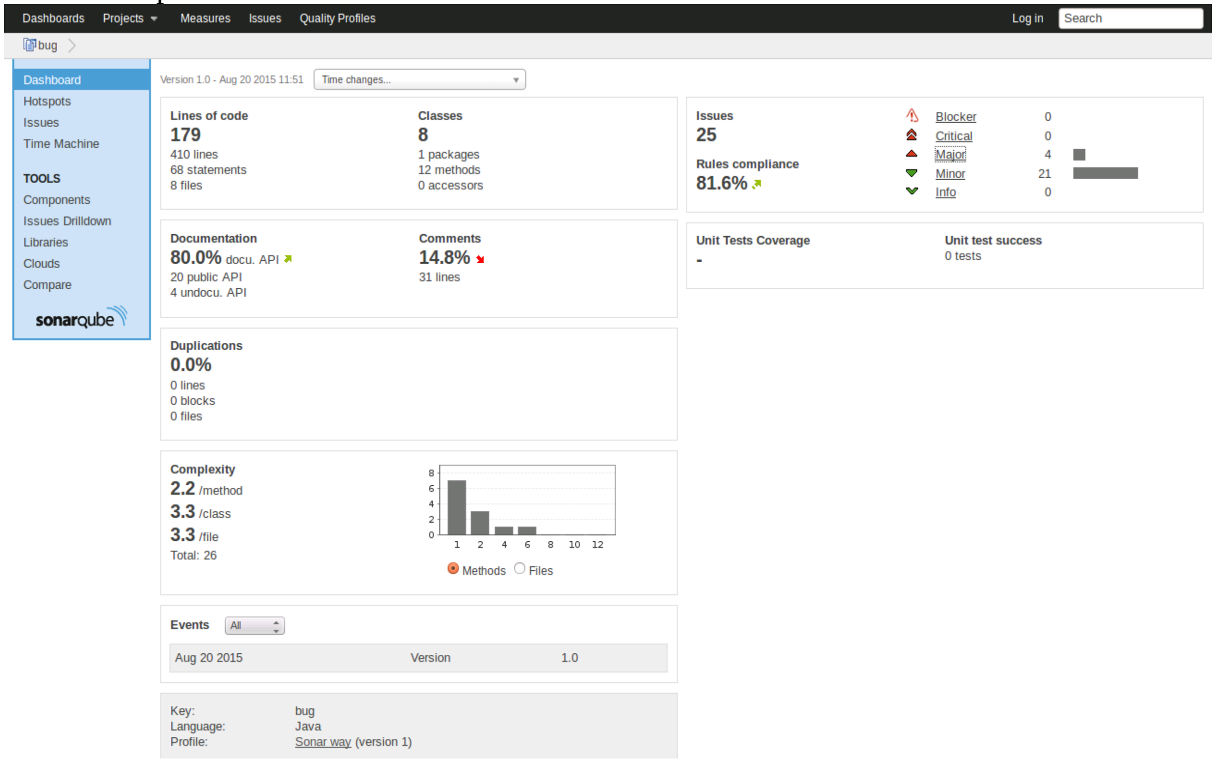
5. Firstly, create an instance of BoxBug and pass the sideLength as parameter.

```
BoxBug cat = new BoxBug(4);
```

Secondly, add the bug to the world and specify its lacation.

```
world.add(new Location(3, 5), cat);
```

Sonar Report:



Version 1.0 - Aug 20 2015 11:51

Lines of code
179
410 lines
68 statements
8 files

Classes
8
1 packages
12 methods
0 accessors

Issues
25

Rules compliance
81.6%

Blocker	0
Critical	0
Major	4
Minor	21
Info	0

Documentation
80.0% docu. API
20 public API
4 undocu. API

Comments
14.8%
31 lines

Unit Tests Coverage
-

Unit test success
0 tests

Duplications
0.0%
0 lines
0 blocks
0 files

Complexity
2.2 /method
3.3 /class
3.3 /file
Total: 26

Events All

Aug 20 2015	Version	1.0

Key:	bug
Language:	Java
Profile:	Sonar way (version 1)