

```

1 package Assign_3;
2
3 import Media.*; // for Turtle and TurtleDisplay
4 import java.awt.*; // for Color objects and methods
5 import static java.lang.Math.*; // for math constants and functions
6 import static java.awt.Color.*; // for Color constants
7 import static Media.Turtle.*; // for FAST turtle
8
9 /** This class randomly generates a cityscape with 3 to 6 buildings each containing
10 5 to 15 stories.
11  *
12  * @author Sawyer Fenwick(st# 6005011)
13  * @version 1.0 November 2 2016
14  */
15 public class City {
16
17     // instance variables
18     private TurtleDisplay display;
19     private Turtle yertle;
20
21     /** This constructor creates the turtle object "yertle", creates a canvas of
22     500x500 and places yertle on the
23     * display. It calls on the "drawCityScape" method which draws the CityScape
24     by calling on several other
25     * methods. */
26
27     public City ( ) {
28
29         // statements including call of method
30         yertle = new Turtle(0);
31         display = new TurtleDisplay(yertle,500,500);
32         int buildings = (int) (3*random())+3;
33         drawCityScape(buildings);
34         display.close();
35
36     }; // constructor
37
38     /** This method creates a square
39     */
40     private void drawSquare ( ) {
41
42         // statements
43         yertle.penDown();
44
45         for (int i = 1; i <=4; i++){
46             yertle.forward(10);
47             yertle.left(PI/2);
48         }
49
50         yertle.penUp();
51
52     }; // drawSquare
53
54     /** This method draws a window built of 4 squares by calling on the
55     "drawSquare" method. */
56     private void drawWindow ( ) {
57
58         //statements
59         yertle.penDown();
60
61         for(int i = 1; i <=4; i++){
62             drawSquare();
63             yertle.right(PI/2);
64         }
65
66         yertle.penUp();
67
68     }; // drawWindow
69
70     C:\Users\sawye\Documents\_BrockU\COSC1P02\Assignments\Assign_3\City.java

```

```

65
66     /** This method draws a rectangle, which will be the outside of each building.
    It is passed the parameter
67     * "storie" which is a randomly generated number between 5 and 15. The storie
    is multiplied by
68     * 30 (the height of 1 storie) to find the proper height of a rectangle that
    will accomodate how many
69     * stories there will be. */
70     private void drawRectangle(int storie){
71
72         int width = 70;
73         int height;
74         height = storie*30;
75         yertle.penDown();
76
77         for (int i = 1; i <=2; i ++){
78             yertle.forward(width);
79             yertle.left(PI/2);
80             yertle.forward(height);
81             yertle.left(PI/2);
82         }
83
84         yertle.penUp();
85
86     }; //drawRectangle
87
88     /* This method draws however many stories is required to fill a building. */
89     private void drawStorie(int storie){
90
91         //local variables
92         int height;
93         height = storie*30;
94         //statements
95         yertle.left(PI/2);
96         yertle.forward(15);
97         yertle.right(PI/2);
98         yertle.forward(20);
99         drawWindow();
100        yertle.forward(30);
101        drawWindow();
102        yertle.left(PI);
103        yertle.forward(50);
104        yertle.right(PI/2);
105
106        for(int i = 2; i <= storie; i ++){
107
108            yertle.forward(30);
109            yertle.right(PI/2);
110            yertle.forward(20);
111            drawWindow();
112            yertle.forward(30);
113            drawWindow();
114            yertle.left(PI);
115            yertle.forward(50);
116            yertle.right(PI/2);
117
118        }
119
120        yertle.right(PI);
121        yertle.forward(height - 15);
122        yertle.left(PI/2);
123
124    }; // drawStorie
125
126    /* This method draws a completed building, using the "drawRectangle" and
    "drawStorie" methods. */
127    private void drawBuilding(int storie){
128
129        drawRectangle(storie);
130        drawStorie(storie);

```

```

131
132     } // drawBuilding
133
134     /* This method draws a complete CityScape, based on the randomly generated
135     numbers for how many buildings
136     * will exist and how many stories each building will have. It calls on the
137     "drawBuilding" method.*/
138     private void drawCityScape(int buildings){
139
140         yertle.moveTo(-210,-225); //"centering"
141
142         for(int i = 1; i <= buildings; i ++){
143             int storie = (int) (10*random())+5;
144             drawBuilding(storie);
145             yertle.forward(70);
146         }
147     } // drawCityScape
148
149     public static void main ( String[] args ) { City s = new City(); };
150 } // City

```