

**I. SpiralBug**

Write a class `SpiralBug` that drops flowers in a outward increasing spiral pattern in an unbounded grid.

Hint 1: Imitate `BoxBug`, but adjust the side length when the `SpiralBug` turns.

Hint 2: The following `SpiralBugRunner` class can be used with the suggested `UnboundedGrid`.

```
import info.gridworld.actor.Actor;
import info.gridworld.grid.UnboundedGrid;
import info.gridworld.actor.ActorWorld;
import info.gridworld.grid.Location;
public class SpiralBugRunner
{
    public static void main(String[ ] args)
    {
        UnboundedGrid grid = new UnboundedGrid<Actor>( );
        ActorWorld world = new ActorWorld(grid);
        etc.
    }
}
```

Save the project as `GWch2prog1`

When perfect, show your teacher the coding and output (run)

\_\_\_\_\_ (teacher signature)

**II. ZBug**

Write a class `ZBug` to implement bugs that move in a “Z” pattern, starting in the top left corner. After completing one “Z” pattern, a `ZBug` should stop moving. In any step, if a `ZBug` can’t move and is still attempting to complete its “Z” pattern, the `ZBug` does not move and should not turn to start a new side. Supply the length of the “Z” as a parameter in the constructor. The following image shows a “Z” pattern of length 4.

Hints: 1) Notice that a `ZBug` needs to be facing east before beginning its “Z” pattern.

2) Use three instance fields:

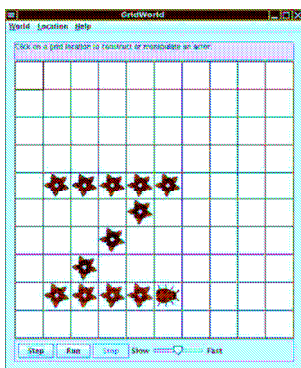
**private int** `segmentLength`; // the number of flowers in each segment

**private int** `steps`; // the number of steps in the current side

**private int** `segment`; // which segment of the Z the `ZBug` is on

3) Use:

`setDirection(Location.SOUTHWEST);` and `setDirection(Location.EAST);` twice



Save the project as `GWch2prog2`

When perfect, show your teacher the coding and output (run)

\_\_\_\_\_ (teacher signature)

**Turn in this sheet to be graded!**