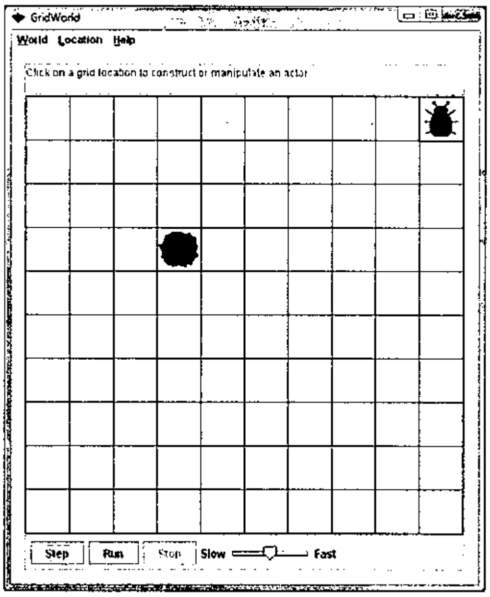
Unit 8. GridWorld Questions.

Part 1.

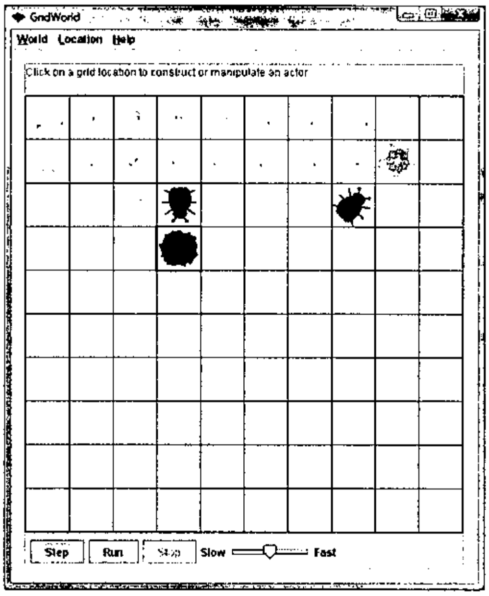


Click on a grid location to construct or manipulate an actor

1. What is the location of the bug?
row __0__, column __9__

2. What is the direction of the bug? __0__ (north)

3. If you click on the Step button once, what happens?
rock doesn't move
bug turns right 45°



Click on a grid location to construct or manipulate an actor

4. What is the location of the bug on the left?
row __2__, column __3__

5. What is the direction of the bug on the left? __180__

6. What is the direction of the bug on the right? __45__

7. If you click on the Step button once, what does the Bug on the left do? turn right 45°
(new direction 225°)

What does the Bug on the right do (be specific)?
steps to (1, 8) on top of flower. leaves flower behind

8. What does a Rock do when you call its act method? nothing

9. What does a Flower do when you call its act method? gets darker

Part 2. After running BoxBugRunner and examining the code, answer the following questions.

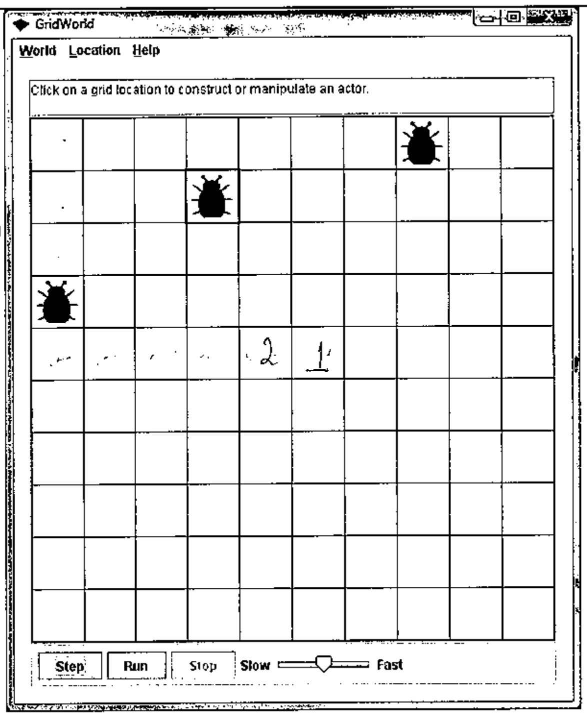
10. When a BoxBug is constructed, what direction is it facing? north

11. When a BoxBug is constructed, what color is it? red

12. Modify the BoxBug class so that when an object is constructed, it will be green by default.

you have to import java.awt.Color then use setColor(Color.GREEN); in constructor

13. Under what circumstances does a BoxBug not trace out a square box? when it runs into anything (wall, rock, other bug)


GridWorld
World Location Help
Click on a grid location to construct or manipulate an actor.
2
1
Step
Run
Stop
Slow
Fast

To the left are three SomeBugs.

```
import info.gridworld.actor.*;
import info.gridworld.grid.*;


public class SomeBug extends Bug{
        public void act() {
                super.act();
                super.act();
        }
}
```

14. After the Step button is clicked, what does the leftmost SomeBug do? 2 steps forward

What does the middle SomeBug do? forward 1 step, turn right 45°

What does the rightmost SomeBug do? turns 2 45° ; so ends facing right.

Part 3.

15. What is the value of n12? 270

16. What is the value of n21? 90

```
Location loc1 = new Location ( 4, 5 );
Location loc2 = new Location ( 4, 4 );
int n12 = loc1.getDirectionToward( loc2 );
int n21 = loc2.getDirectionToward( loc1 );
```

17. What is displayed?

800 , -2

```
Location loc3 = new Location ( 800, -3 );
Location loc4 = loc3.getAdjacentLocation( Location.EAST );
System.out.println( loc4.getRow() + ", " + loc4.getCol() );
```

loc 3 (800, -3)
loc 4 (800, -2)

18. Where is the setDirection method defined?

Actor class

19. If an ABug is added to the grid, what (if anything) does it do when the Step button is clicked?

exactly as a bug except they move left bc they start facing left

```
import info.gridworld.actor.*;
import info.gridworld.grid.*;

public class ABug extends Bug{

    public ABug() {
        setDirection( 270 );
    }
}
```

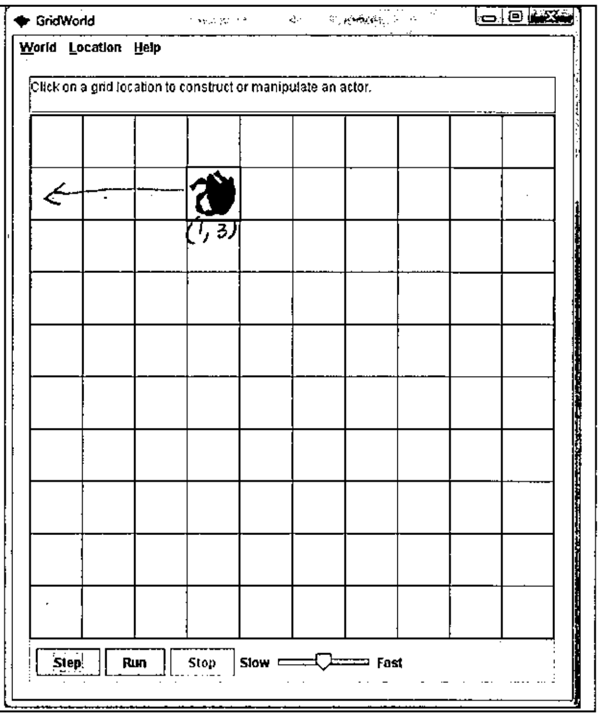
```
import info.gridworld.actor.*;
import info.gridworld.grid.*;

public class Actor_A extends Actor{

    public void act() {
        Grid<Actor> gr = getGrid();
        if (gr == null)
            return;

        int row = getLocation().getRow();
        moveTo( new Location( row, 0 ) );

        int n = gr.getNumCols();
        Location loc = new Location( row, n-1 );
        Rock r = new Rock(getColor());
        r.putSelfInGrid(gr, loc);
    }
}
```

in Actor class

in Grid<> interface

Actor class

Actor class



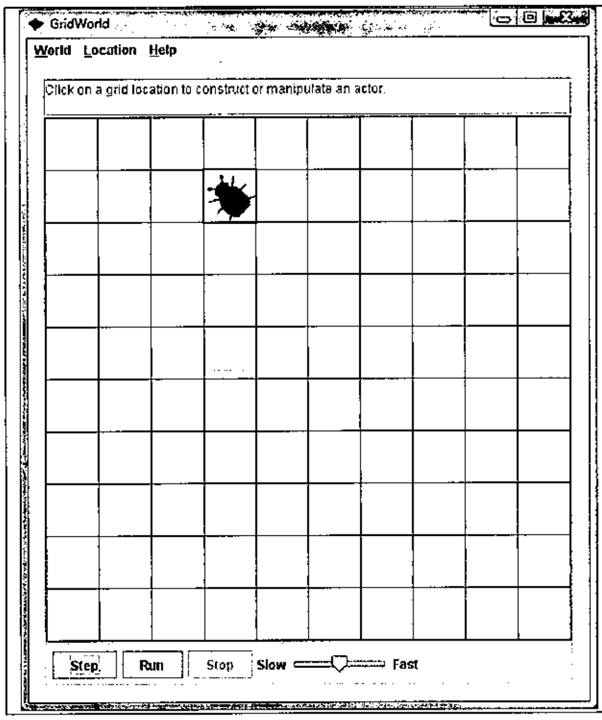
Click on a grid location to construct or manipulate an actor.

Check the information in the appendix before answering problems 20 and 21.

20. The grid above has an Actor_A in it. What happens when the Step button is clicked?

moves the actor to col 0 of present row, adds a rock at the end of the row.

21. What happens if the Step button is clicked a second time? nothing would appear to happen



Click on a grid location to construct or manipulate an actor.

```
import info.gridworld.actor.*;
import info.gridworld.grid.*;

public class Buggy extends Bug{
        public Buggy(){
                setDirection( -45 );
        }

        public void turn() {
                setDirection(getDirection() +
                        Location.LEFT);
        }
}
```

The grid contains one Buggy.

22. After you click the Step button once, Buggy will be at row 0, column 2, and have a direction of -45 (give a number).

23. After you click the Step button a second time, Buggy will be at row 0, column 2, and have a direction of -90 (give a number).

24. If Buggy moves, does it leave a flower behind? Explain. yes because bug does & buggy extends Bug

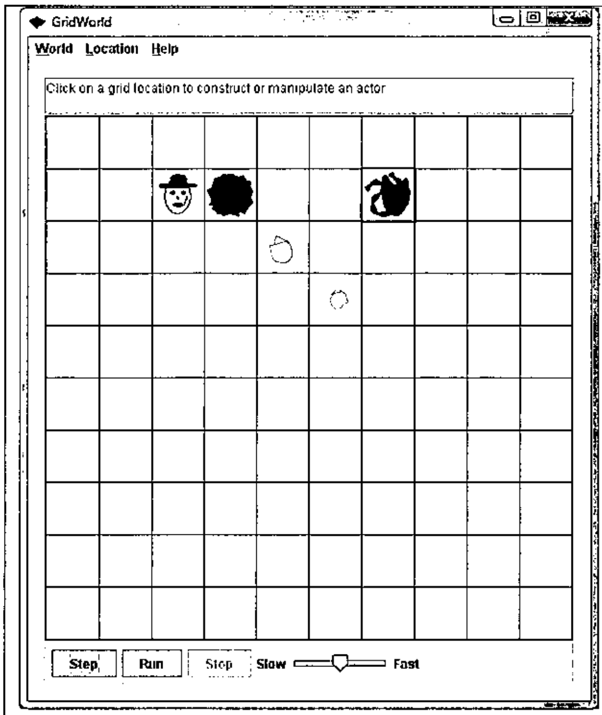
25. This class compiles and may run for awhile but eventually you get a runtime error. Why?

bc no check using gr.isValid(location)

26. What method of the Grid interface could be used to prevent this error from happening?

isValid(loc2)

```
import info.gridworld.actor.*;
import info.gridworld.grid.*;
import java.util.*;

public class Bad_Actor extends Actor{
    public void act() {
        Grid<Actor> gr = getGrid();
        if (gr == null)
            return;

        Location loc1 = getLocation();
        int dir = getDirection();
        Location loc2 = loc1.getAdjacentLocation( dir );
        moveTo( loc2 );
    }
}
```



Click on a grid location to construct or manipulate an actor

The above grid contains (from left to right) a Mean_Actor, a Rock, and a (regular) Actor.

```
import info.gridworld.actor.*;
import info.gridworld.grid.*;
import java.util.*;

public class Mean_Actor extends Actor{
  public void act() {
      Grid<Actor> gr = getGrid();
      if (gr == null)
          return;

      ArrayList<Location> a =
gr.getOccupiedAdjacentLocations(getLocation());

if ( a.size() > 0 ){
          int n = (int)( a.size() * Math.random() );
          Location loc = a.get(n);
          moveTo( loc );
      }
  }
}
```

27. After you click the Step button once, the Mean_Actor will be at

row 1 and column 3.

28. After you click the Step button a second time, the Mean_Actor will be at

row 1 and column 3.

29. If a Mean_Actor moves, does it leave a flower behind? Explain. no because mean Actor extends Actor not Bug

Part 4 (Critters) study Critter Java

30. Suppose you were going to write a FrontCritter that only eats something directly in front of it (and they cannot be a Critter or a Rock). It moves just like a regular Critter. What method would be the best method to override?

[✓ a)] getActors
b) processActors
c) getMoveLocations
d) selectMoveLocation
e) makeMove

40. After a Critter moves from Location(2, 4) to Location(3, 5), what will its direction be? ___
same direction

After the Step button is clicked once ...

41. Where will the top Critter be? __________

(4, 3)

42. Where will the bottom Critter be? ________

row 7 col 5 - 7

or row 8

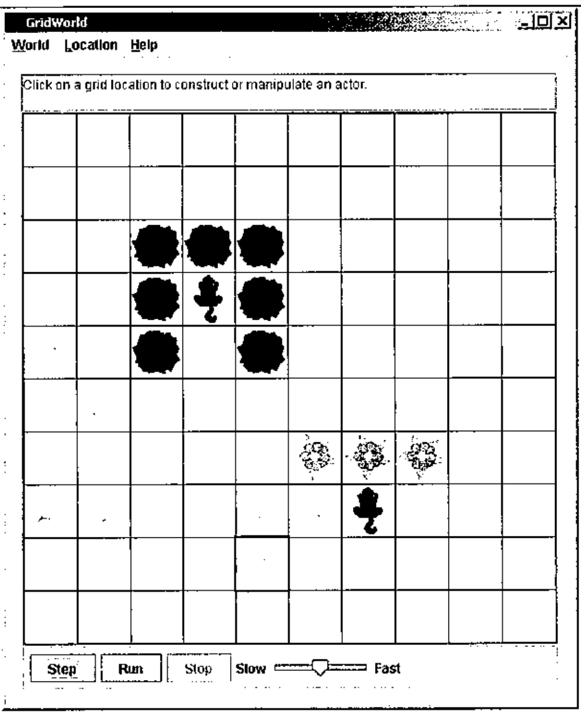
43. What happens to the 7 Rocks? __________

nothing

44. What happens to the 3 Flowers? ________

a FrontCritter would eat one flower

regular Critter eats all flowers



Click on a grid location to construct or manipulate an actor.

Figure 1. There are 7 black rocks, 2 blue Critters, and 3 pink Flowers.

45. Will the MadCritter class run without errors? If Yes, how does a MadCritter act differently than a regular Critter? If no, what is the problem?

the only method overridden is getActors & it will never return anything, so nothing ever gets eaten.

```
public class MadCritter extends Critter {
    public ArrayList<Actor> getActors() {
        return null;
    }
}
```

46.How does a Bad Critter move?

he moves always right by 1 column

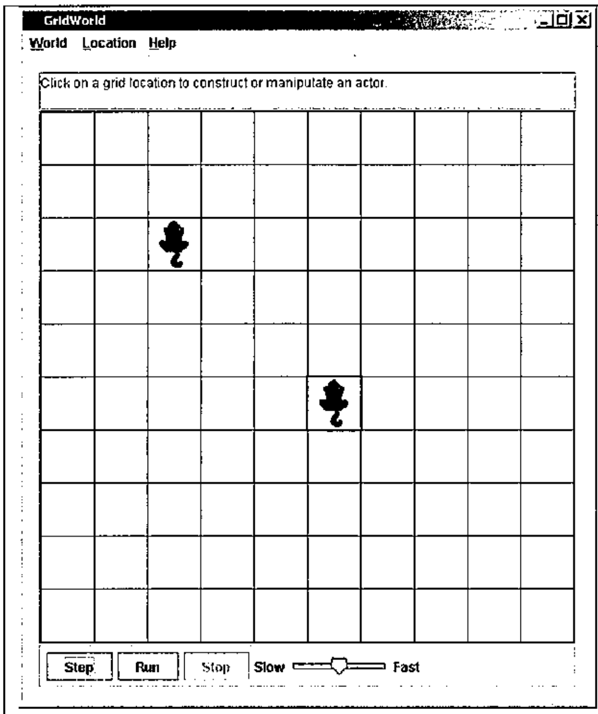
47. Will a BadCritter cause a program to eventually crash?

yes, will exceed the numCols on bounded grid

```
// this compiles

public class BadCritter extends Critter {
public ArrayList<Location> getMoveLocations(){
    ArrayList<Location> list = new ArrayList<Location>();
    Location loc = getLocation();
    list.add( new Location( loc.getRow(), loc.getCol() + 1 )
);
    return list;
    }
}
```

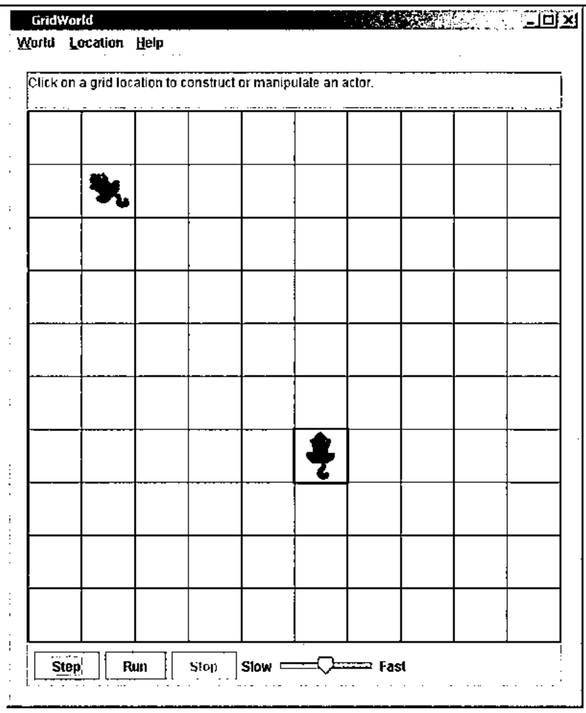
48. Suppose you were going to write a RandomCritter that behaves like a regular critter except that it can move to any random empty location in the grid. What method would be the best method to override?

[✓ a)] getMoveLocations
b) selectMoveLocation
c) makeMove
d) You would need to override more than one of these methods.



Click on a grid location to construct or manipulate an actor.

Figure 2.



Click on a grid location to construct or manipulate an actor.

Figure 3

49. Figure 2 shows two critters: one is regular critter and one is a chameleon critter. I changed the icons so they look the same. Figure 3 shows the same grid after one step. Select the TRUE statement.

a) The top critter must be a chameleon critter.
b) The bottom critter must be a chameleon critter.
c) Either one could be a chameleon critter.
d) There's been an awful mistake, neither one could be a chameleon critter.

50. Select the TRUE statement(s). An OddCritter ...

a) eats only 1 neighboring actor at a time (provided that there are neighbors).
[✓ b)] can eat rocks, critters, any actor.
c) moves just like a critter.
d) eventually throws a run-time error.

```
// This compiles
public class OddCritter extends Critter {
public void processActors(ArrayList<Actor> actors){
        if ( actors.size() == 0 )   return;
        int n = (int) ( actors.size() * Math.random() );
        actors.get( n ).removeSelfFromGrid();
}
}
```

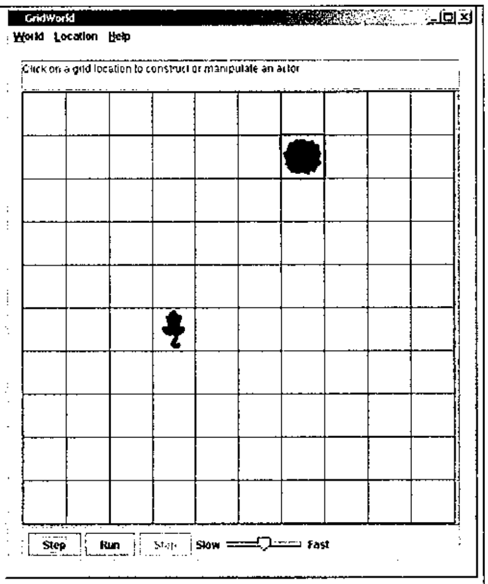
no turn called in Critter class

51. A BitterCritter ...

a) turns every time it moves.	T F
[✓ b)] acts exactly like a critter.	T F
c) eventually throws a run-time error.	T F
d) turns only if it eats an actor.	T F

```
// This compiles

public class BitterCritter extends Critter{
    public void turn(){
        int dir = getDirection();
        setDirection( dir + 90 );
    }
}
```

```
public class SitterCritter extends Critter  {
private Location x;

public ArrayList<Actor> getActors() {
    ArrayList<Actor> a = super.getActors();

    Grid<Actor> gr = getGrid();

    x = null;
    ArrayList<Location> b = gr.getOccupiedLocations();
    b.remove( getLocation() );
    if ( b.size() > 0 ){
        int n = (int)( b.size()*Math.random() );
        x = b.get( n );
    }

    return a;
}

public void makeMove( Location loc ) {
    super.makeMove( x );
}
}
```


GridWorld
World Location Help
Click on a grid location to construct or manipulate an actor
Step
Run
Stop
Slow
Fast

There is a SitterCritter and a Rock. This compiles and runs (at least for awhile).

52. When the Step button is clicked, what happens to the SitterCritter in the above grid? If it moves, where does it move? If the program crashes, explain.

to the rocks position (1,6)

53. When the Step button is clicked a second time, what happens to the SitterCritter in the above grid? If it moves, where does it move? If the program crashes, explain.

disappears bc moves to null location

54. What design principle(s) does the SitterCritter violate?

polymorphism/ hierarchy - if you extend a class, handle what can happen.