Set 3

Assume the following statements when answering the following questions.
Location loc1 = new Location(4, 3);
Location loc2 = new Location(3, 4);

1. How would you access the row value for loc1?

loc1.getRow()

2. What is the value of b after the following statement is executed?
boolean b = loc1.equals(loc2);

false

3. What is the value of loc3 after the following statement is executed?
Location loc3 = loc2.getAdjacentLocation(Location.SOUTH);

Location(4, 4)

4. What is the value of dir after the following statement is executed?
int dir = loc1.getDirectionToward(new Location(6, 5));

135 (southeast)

5. How does the getAdjacentLocation method know which adjacent location to return?

First, round the direction to closest multiple of 45.
Then, use a branch to determine what dc and dr should be.
Finally, add dc and dr to the original location to get the adjacent location.

Set 4

1. How can you obtain a count of the objects in a grid? How can you obtain a count of the empty locations in a bounded grid?

getOccupiedLocations().length
getNumRow() * getNumCols() - getOccupiedLocations().length

2. How can you check if location (10,10) is in a grid?

isValid(new Location(10, 10))

3. Grid contains method declarations, but no code is supplied in the methods. Why? Where can you find the implementations of these methods?

Because it is an abstract class. BoundedGrid and UnboundedGrid.

4. All methods that return multiple objects return them in an ArrayList. Do you think it would be a better design to return the objects in an array? Explain your answer.

Because in these methods we can't know the number of objects in advance. To use array we will

have to iterate one more time to determine the size of the array.

Set 5

1. Name three properties of every actor.

grid, location, direction

2. When an actor is constructed, what is its direction and color?

Location.NORTH
Color.BLUE

3. Why do you think that the Actor class was created as a class instead of an interface?

Most method of Actor 's implementation is alomst determined, except that act() is free.
For ActorWorld, Actor needs to be an abstract class.
Actor has member variable.

4. Can an actor put itself into a grid twice without first removing itself? Can an actor remove itself from a grid twice? Can an actor be placed into a grid, remove itself, and then put itself back? Try it out. What happens?

No. The method putSelfInGrid will first check if the actor is currently in a grid.
No. The method removeSelfFromGrid will first check if the the actor is currently in a grid.
Yes. It succeed and nothing went wrong.

5. How can an actor turn 90 degrees to the right?

setDirection((getDirection() + 90) % 360).

Set 6

1. Which statement(s) in the canMove method ensures that a bug does not try to move out of its grid?

```
if (!gr.isValid(next))
   return false;
```

2. Which statement(s) in the canMove method determines that a bug will not walk into a rock?

```
return (neighbor == null) || (neighbor instanceof Flower);
```

3. Which methods of the Grid interface are invoked by the canMove method and why?

isValid, get
To check if the destination is in the grid and does not contain a non-flower object.

4. Which method of the Location class is invoked by the canMove method and why?

getAdjacentLocation

To find out the location of the destination.

5. Which methods inherited from the Actor class are invoked in the canMove method?

getGrid, getLocation, getDirection

6. What happens in the move method when the location immediately in front of the bug is out of the grid?

The bug removes itself from the grid.

7. Is the variable loc needed in the move method, or could it be avoided by calling getLocation() multiple times?

Yes. Because the location may has changed, and getLocation() returns the current location, so loc is used to remember the original one.

8. Why do you think the flowers that are dropped by a bug have the same color as the bug?

The flowers dropped is constructed with the bug's color.

9. When a bug removes itself from the grid, will it place a flower into its previous location?

Yes, it will.

10. Which statement(s) in the move method places the flower into the grid at the bug's previous location?

```
Flower flower = new Flower(getColor());
flower.putSelfInGrid(gr, loc);
```

11. If a bug needs to turn 180 degrees, how many times should it call the turn method?

4.