The Answer of Part4

Set 7

The source code for the Critter class is in the critters directory

1. What methods are implemented in Critter?
   Ans: act(), getActors(), processActors(), getMoveLocations(), selectMoveLocation()and makeMove().

2. What are the five basic actions common to all critters when they act?
   Ans: getActors(), processActors(), getMoveLocations(), selectMoveLocation() and makeMove().

3. Should subclasses of Critter override the getActors method? Explain.
   Ans: Some subclasses should, like the class CrabCriter, which only process the Actors on then front, left and right location.

4. Describe the way that a critter could process actors.
   Ans: Find the ArrayList of actors, then call the processActor() method, with the ArrayList as the parameter. The Actors can be processed by calling their methods.

5. What three methods must be invoked to make a critter move? Explain each of these methods.
   Ans: getMoveLocations() -> to find the locations to be chosen to make move.
        selectMoveLocations() -> to choose a location from the available locations to move.
        makeMove() -> to move to the destination.

6. Why is there no Critter constructor?
   Ans: Because the Critter class has no special value different from the class Actor.
        So it doesn't need constructor to init value.

Set 8

The source code for the ChameleonCritter class is in the critters directory

1. Why does act cause a ChameleonCritter to act differently from a Critter even though ChameleonCritter does not override act?
   Ans: Beacause the method called by act method are overriden.

2. Why does the makeMove method of ChameleonCritter call super.makeMove?
   Ans: Because super.makeMove is the mothod of the supclass Critter. Excepting direction, the implement of makeMove of ChameleonCritter is the same of class Critter, so calling super.makeMove.

3. How would you make the ChameleonCritter drop flowers in its old location when it moves?
   Ans: I would override the method makeMove to implement it.

4. Why doesn't ChameleonCritter override the getActors method?
   Ans: Because it get actor in the same way with it super class Critter: All the neighbor actors.

5. Which class contains the getLocation method?
   Ans: Actor class.

6. How can a Critter access its own grid?
   Ans: Using getGrid() method.

Set 9

The source code for the CrabCritter class is reproduced at the end of this part of GridWorld.

1. Why doesn't CrabCritter override the processActors method?
   Ans: Because it has the same way to process actors. That's eating them, excepting rock and critter.

2. Describe the process a CrabCritter uses to find and eat other actors. Does it always eat all neighboring actors? Explain.
   Ans: First, get the neighbor actors.
      Second, from the neighboring actors, choose the actors that are not

flower or rock and remove them from the grid. If the neiboring actor is flower or critter, it will not be eaten.

3. Why is the getLocationsInDirections method used in CrabCritter?
Ans: To find all the Location in left and right.

4. If a CrabCritter has location (3, 4) and faces south, what are the possible locations for actors that are returned by a call to the getActors method?
Ans: (4, 3), (4, 5) and (4, 4).

5. What are the similarities and differences between the movements of a CrabCritter and a Critter?
Ans: Similarities: They both eat actors except critters and rocks. They only move 1 step in each act when they can move. The both randomly select a location to move.
Deferences: Critter can move in all directions, while CrabCritter can only move in left or right.

6. How does a CrabCritter determine when it turns instead of moving?
Ans: When the distination location is the location of itself, the CrabCritter will turn instead of moving.

7. Why don't the CrabCritter objects eat each other?
Ans: Because CarbCritter is the extended class of Critter, in the processActor method from Critter, Critter can not be eaten, so CrabCritter don't eat each other.