

I. RockHound using ch4IProgramI

- A) Create a class called RockHound that extends Critter. A RockHound gets the actors to be processed in the same way as a Critter. It removes only rocks in that list from the grid. A RockHound moves like a Critter.

Save the project as GWch4Iprog1

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

(teacher signature)

II. Glutton using ch4IProgramII

- A) Create a class called Glutton that extends Critter. A Glutton gets the actors to be processed in the same way as a Critter. It removes everything (including Glutton's) in that list from the grid. A Glutton moves like a Critter.

Save the project as GWch4Iprog2

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

(teacher signature)

III. DyingCritter using ch4IProgramIII

- A) Create a class called DyingCritter that extends Critter. Override the processActors method so that a DyingCritter acts like a Critter except that it must eat within 5 moves or it dies. Each time it acts without eating it changes color: green, then yellow, then red, then black, then removed from grid.

Note: The color is reset back to blue once it eats.

Save the project as GWch4Iprog3

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

(teacher signature)

Turn in this sheet to be graded!

I. ChameleonCriter1 using ch4Lab1 and 2

- A) Create a class called ChameleonCriter1 that extends Critter. Modify the makeMove method in ChameleonCriter1 to drop a SimpleFlower, of the same color as the ChameleonCriter1, in the old location - i.e. after the ChameleonCriter1 moves, put a SimpleFlower in its old location only if the ChameleonCriter1 actually moved to a new location. Do not change the ChameleonCriter class. Also, a ChameleonCriter1 does not eat SimpleFlowers.
Hint 1: A variable is needed to keep track of the ChameleonCriter1's old location.
Hint 2: use the code like: `if(!oldLoc.equals(loc))`

Save the project as GWch4IIprog1

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

(teacher signature)

II. ChameleonCriter2 using ch4Lab1 and 2

- A) Create a class called ChameleonCriter2 that extends Critter. Modify the processActors method of ChameleonCriter2 so that if the list of actors to process is empty, the color of the ChameleonCriter2 will darken (like a flower).

Save the project as GWch4IIprog2

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

(teacher signature)

III. ChameleonKid using ch4prog2

- A) Create a class called ChameleonKid that extends ChameleonCriter2 as modified in program II above. ChameleonKid changes its color to the color of one of the actors only immediately in front or behind. If there is no actor in either of these locations, then the ChameleonKid darkens like the modified ChameleonCriter2.

Save the project as GWch4IIprog3

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

(teacher signature)

Turn in this sheet to be graded!