**AP Java Marine Biology Simulation**

**Worksheet Ch 4**

1. Please the read first paragraph and the Problem Specification on page 63. The two new kinds of fish are darter fish and slow fish. In the space to the right of the environment, please indicate the new location and direction for each darter fish:

(0, 1 East) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(1, 0 North) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(1, 1 East) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(2, 0 East) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. At the bottom of page 63 the author lists some public accessor methods which can be used by any type of fish. The other three public accessor methods are environment(), isInEnv() and toString() (Make sure you understand why this is so).

a. Why isn’t the method isInEnv() dependent on the type of fish (ie, why don’t we need a separate isInEnv() method for each new type of fish)? You might want to look at the code for this method on page 33 or in Appendix B5. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b. Why might you want to override the toString() method for each of these new types of Fish? (ie, what doesn’t the toString() method in Fish do that might be helpful?) Again, you might want to look at the code for this method on Appendix B5. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Continue reading pp 63 and 64. At the end of the first full paragraph on page 64, the author states “...the DarterFish class could redefine the generateChild method...”. Take a look at the original generateChild method from the Fish class on page 55 and indicate the changes you would make to create a version for the DarterFish class.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. After reading the explanation on the last half of page 64, answer these questions.

a. The author states in the last bulleted point that the methods location, direction, etc. are not redefined. Why don’t these have to be redefined? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b. A darter fish tries to execute the breed method. Explain how it can breed even though it doesn’t have its own breed method. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

c. Why doesn’t a darter fish need its own breed method? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

d. Why doesn’t a darter fish needs its own die method (ie, what is the die method really doing - look on page 56 or in Appendix B8)? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Read page 65. The class diagram for DarterFish indicates that emptyNeighbors is inherited from the Fish class (there isn’t an overlapping box as there is with move, for example). Why doesn’t the DarterFish class require a different emptyNeighbors method? (You might want to examine the code for this on pg. 36 or in appendix B). \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Circle the correct class that each method is dynamically bound to:

a. A DarterFish calls the act method; this is bound to Fish / DarterFish

b. Inside the act method, the breed method is bound to Fish / DarterFish

c. Inside the breed method, the method emptyNeighbors is bound to

Fish / DarterFish

d. Still in the breed method, the generateChild method is bound to

Fish / DarterFish

7. Read page 66. In the second paragraph (starting with “A subclass inherits...”) the author mentions the items inherited.

a. What is the one part of the Fish class that isn’t mentioned and in fact not inherited? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b. If this (these) were inherited, what might be the result of using this (these?)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

8 At the end of the next paragraph, the author says that the code for method nextLocation at the bottom of the page is buggy.

a. Find the error and describe what happens if the code is left this way.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b. Without looking ahead, try to rewrite the only the if conditional logic so that nextLocation works as intended using the space below

9. Study the code for the DarterFish class move method shown on page 67.

a. Now look at the code for the Fish class move method on page 34 (or in appendix B). In the “then” or true portion of the if statement why are three statements that surround the changeLocation method call eliminated in the DarterFish version?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b. In the else portion, name the class that each method call is associated with.

changeDirection \_\_\_\_\_\_\_\_\_\_\_\_\_

direction \_\_\_\_\_\_\_\_\_\_\_\_\_\_

reverse \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

10. What if a darter fish alternated between the colors yellow and blue each time it changed direction (sort of like a chameleon).

a. Describe the changes you’d have to make to both the move method and to the DarterFish class. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b. When the fish changes color would you have to let the environment know of this change by using the environment’s recordMove method? Why or Why not? \_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

11. Finish reading page 67. At the bottom of the page study the code for one of the DarterFish constructors. Notice how the constructor for DarterFish uses the equivalent Fish constructor by using the keyword super. Also notice how a specific color is named. In the space below, look at the constructors for the Fish class (page 28 or Appendix B2-3) and write the equivalent DarterFish constructors in the space below.

12 Read the first half of page 68. On the back or on a separate sheet of paper answer Analysis Question Set 1 on page 68 (you only need to do #2, since you did #1 already).

13. Read the bottom half of page 68. What was the odd thing that Pat noticed when testing the DarterFish code? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(There are fish that actually do this under certain circumstances. Dolphins (not actually fish) and salmon swimming upstream to spawn come to mind - an interesting project: do some research to see if there are other fish that exhibit this type or some similarly “odd” behavior that could be simulated in the future).

14. The narrative indicates that debugging was turned on at the beginning of the step method of the Simulation class and then turned off at the end of this method. Write the two statements that would be used to do this. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

15. Also described were debugging statements placed into the method nextLocation. Write the debug statements that would be used to display useful information while debugging.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

16. Read pp 69 and 70. Try running the simulation with a different seed number (apparently Pat used 17, so you should use some other integer). Create a table like that on page 70 to see if your results approximate the results Pat achieved.

17. Work through Exercise Set 1 on page 71. Some of these you’ve already worked on. Exercises 2, 4, 5, 7 and the chameleon exercise noted above should be a minimum set.

18. Read pages 72 and 73 which detail changes made to implement SlowFish. You should notice that there’s lots of things done here which are similar to things you already did with either DarterFish (inheriting much of the Fish class) or implementing breeding and dying (using random numbers).

19. On page 72, Pat decided to implement a nextLocation method for the SlowFish class and shows that implementation on page 73 including the use of the keyword super.

a. Where else have you see the need to use the keyword super? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b. If Pat had decided to implement a move method in the SlowFish class instead of a nextLocation method, write the code for this move method (hint it’s quite similar to the code for nextLocation and about as brief).

protected void move()

{

20. Following the example on page 73 of a SlowFish constructor, try writing the other two constructors similar to the constructors for Fish.

21. On a separate sheet, do Analysis Question Set 2 on page 74.

22. Read pages 74 and 75. Near the top of page 75, the programmer says “I added a debugging statement...” Write a debugging statement that would provide the desired information and indicate where within the nextLocation code on page 73 you’d place your statement. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

23. On a separate sheet write your answers to Analysis Question Set 3 on page 75.

24. In the Exercise Set 2 on pages 76 and 77, there’s a whole bunch of good exercises to do to help you really understand the case study. Some comments:

* Make sure you know how to either substitute the earlier Fish class or comment out the breeding and dying sections as suggested in #2 (and also in an exercise with Darterfish). This kind of process to limit the scope of what you’re testing is invaluable in many other situations aside from the case study.
* Redefining toString is also a useful debugging tool for the case study.
* Running a test and creating a chart similar to Pat’s on the previous pages is an excellent way to get a better feel for the validity of your code.
* Implementing at least one of exercises 6, 7 or 8 should be a requirement in order to really understand the workings of the case study and be prepared for anything the AP test will throw at you in May. In particular with #7, you’ll need to consider whether or not you want to have DarterFish and SlowFish breed and die.
* Your teacher also has access to a specification for a FastFish in the teacher’s guide