Lab07: Shark Lab — Due: Dec 17, 2018 at 8:00am

Your one and only program this time is known as "The Shark Lab" but actually consists of a number of files that you will submit. For this lab, you will create a "water world" in which a shark will attempt to eat three fish. The rules are as follows:

1. The water world is a 10×10 grid of squares using Python's default coordinate. The shark will start at location (7,2) and the three fish will start at locations determined by the user.

2. Each fish will start with an initial (random) compass direction (N, S, W, E), moving one square in that direction each turn. When they hit a wall, they will merely reverse direction. If a fish ends a turn facing a wall, it will reverse direction, moving one square away the next. Fish only move along rows and columns, never diagonally.

3. The shark will always move two spaces towards the nearest fish (nearest = fewest spaces/moves away) with a random choice made if there are two or more fish tied for closest. Sharks will always move diagonally unless they are on the same row or column as a fish they are pursuing. It is possible for a shark to switch which fish it is pursuing if another one is closer during the next turn. If two are equally close, it will stay after the one it was already chasing.

4. The sequence of each turn is as follows: the fish move one space each and then the shark moves two spaces (not necessarily in the same direction). If at any point the shark is at the same location as a fish, the fish is eaten and the shark's move ends. Obviously a fish ceases to move (and should be removed) once it is eaten.

   HOWEVER, if a fish starts its turn within three spaces of a shark, it turns and flees in the most-opposite compass direction possible (N, S, W, E). If the shark is on an exact diagonal, the fish will randomly choose either of the possible fleeing directions. In addition, any fish in "flee" mode has the ability to move through walls, meaning that if a fish hits a hall, that fish will pass through it and come back onto the grid on the other side. A fish is in "flee" mode regardless of whether the shark is actually after it or another fish. Fish should check for flee mode only before they move, but a fish immediately exits flee mode after passing through a wall.

All of this will be accomplished on a single window that will have entry fields for your three fish (you can call them anything but I will use alpha, beta, and gamma) as well as the grid itself. You should have a button for "Start" (after entering your three sets of coordinates), a button for "Move Fish" or "Move Shark" (which will carry out a turn for whichever is next), and a "Quit" button that is always available. Your window should also have a message area for text prompts to the user.

Your program will begin be opening the window with the shark image already at location (7,2). You should have a message instructing the user to enter the coordinates of the three fish (the move button should be inactive at this point). There should be three entry objects for handling the coordinate input and a non-crashing UI to handle invalid entries (all coordinate pairs should be distinct and entered as "m,n" with m and n as digits 0-9 and no parentheses required). Once the user clicks "Start" (and the entries are valid), the fish should now likewise show on the grid. The Start button should now be inactive and the Move button should be active.

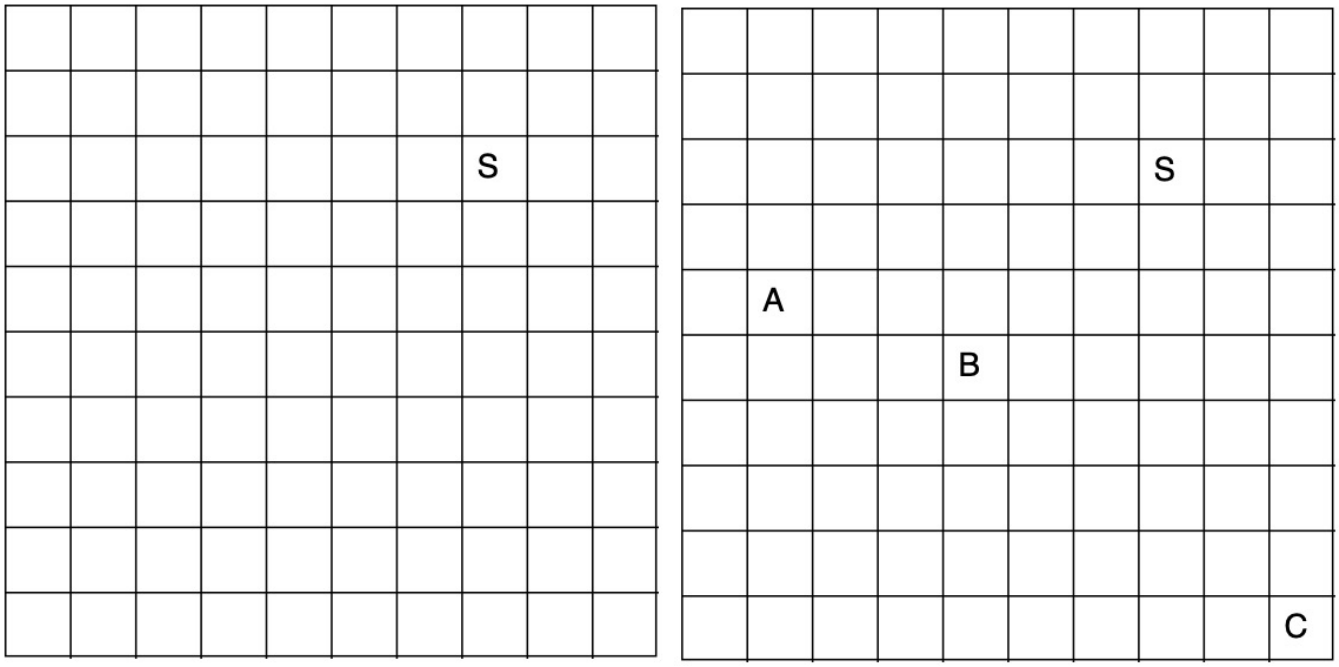
Once all fish are eaten, your program should state such and ask the user if they want to try

again. At this point, the Move button should be inactive and the Start button active.

I am expecting an actual image GIF for both the shark and fish, though it does not need to be complicated. It is acceptable if you just find a GIF you like on the internet as long as you provide proper attribution in your coding comments.

IMPORTANT: your shark icon does not need to indicate a direction, but your fish should be rotated so that it is always visually clear in which direction they are headed. When a fish is in flee mode, there should be a visual cue that it is doing so (e.g. it turns red, has a screaming mouth, etc.). Refer to page 120 of your textbook or the Graphics API for other info about displaying images.

To clarify how the shark and fish should behave, here is an example. The first image shows what your grid should look like when the window first opens and the second is what it should look like after placement of alpha, beta, and gamma at (1,4), (4,5), and (9,9), respectively.



In this scenario, alpha would have any random starting direction and move one space in that direction. Beta would begin in "flee" mode and would head either South or West to try to escape. Gamma could begin facing any direction but will ultimately move either North or West one space. After the fish move, the shark would move to (5,4) as it pursues Beta. One click moves the fish (simultaneously), another moves the shark.

Specs for this Program

To complete this lab, you will be working with a partner to write and submit the following modules

- Fish.py: this module will contain definitions for the Fish class and all supporting logic and methods
- Shark.py: this module is like Fish.py but for the shark

- SharkGUI.py: this module should handle the drawing (and redrawing) of the grid as well as handling all GUI events like button clicks
- SharkRunner.py: this module contains the main function that runs the whole thing
- Button.py
- plus any supplementary image files

As before, no functions/methods should be more than 100 lines of code, including comments.

SharkRunner should import all three of the other files. None of the others should import each other (or SharkRunner). All window drawing should be done in SharkGUI and the window may not be passed as a parameter to any module/method outside of SharkGUI – this will force you to edit your Button class (or create a new one).

In dividing up the labor, one of you will be responsible for both Fish and SharkRunner while the other does Shark and SharkGUI. Either of you may provide the button class (and you may certainly use what you have already done) but **every submitted file should have only one author!** You are welcome to talk to each other and get ideas and help, especially during debugging, but only the author writes code. The image files you may divide up as you see fit but each of you must be responsible for at least one of them.

When your program is finished, you and your partner will do a demo in front of the class that will involve each of you walking us through the code you wrote, while fielding questions that come up, and then actually running the program with us watching.

Grading

This project is worth 15 points.

- You and your partner will earn the same scores for the following: 8.5 points for having your program run correctly as specified in this write-up and 0.5 points for aesthetics
- You will earn individual scores for the following: 3.0 points for your demo presentation, 1.0 point for sufficient commenting, 0.5 points for adhering to the 100 lines/method rule, and 1.5 points for actively engaging during others' demos (this final element refers to not just paying attention but also asking intelligent questions during the code walkthroughs)

FAQ

What exactly happens when a fish gets to a wall? When the fish gets to the wall, it remains in that position and direction. The following move it would turn around and move one space away from the wall.

Can two fish occupy the same space after one turn? No. Remember that while it looks as if they are moving at the same time, you are actually moving the fish one at a time. If the spot to which the fish would move is taken, it just does not move, but stays pointing in the same direction. EXCEPTION: if the fish is in Flee Mode and it has a choice of directions, it should choose the

other if one of the directions is blocked!

Do the fish ever win? Yes! It is possible for the fish to survive a sort of stalemate position. It's up to you to figure out when that happenes. If the shark and fish are in a stalemate position, your program must be able to identify it (though it does not need to be immediate) and indicate that the Shark dies from hunger. Congratulate the appropriately named fish with a nice message.

Do the fish need to be distinguishable on the grid? Yes, but this can be done in a variety of ways. You decide.

What is the precise fish behavior during Flee mode? Remember, Fish check for "Flee" only at the beginning of their turn. They remain in "Flee" until they are either eaten or pass through a wall. If the former, they are removed from the grid, if the latter, they immediately leave "Flee" at the end of their move. Should the Shark decide to chase another fish, it is possible that a fleeing fish will find itself more than 3 spaces away from the Shark at the beginning of a turn, in which case it should return to normal mode.

Any details on the randomness thing? Here?s one way of thinking about it: if two choices are equal, it should only be a random thing if the being in question was not already headed in one of the directions or towards one of the fish. Example: if a fish is heading North, and finds itself now in Flee mode with choices being North or East, the fish would remain heading North. If the flee choices are East and South, it would choose one of those randomly (both of these subject to the exception concerning blocked Fish above). If a shark is chasing fish Beta and finds itself now with a choice of Beta or Gamma which are equally close, it should still chase Beta. If it was not chasing either fish, then it should choose randomly. Also note that it is possible for the Shark to be equally far from all three fish at the same time.

What messages are you expecting in the message area? At a minimum, you should have an initial message instructing the user to enter the fish coordinates, one to let the user know that which (if any) coordinate entries were invalid, a message informing the user which fish has just been eaten, and a message when the battle is over indicating the winner. You are welcome to add further messages (I encourage it!).