Regis University CC&IS
CS210 Introduction to Programming
Alice Programming Assignment 4: Decisions

This program will modify your previous animation from Alice Assn 3, adding decisions.
The program will now decide whether to make the fish move left or right, and whether to loop or zig zag. If looping, it will check that the looping distance is not too large. If zig zagging, it will decide whether to zig zag an additional time, based upon how far it moved. The water pressure will be computed differently if the fish is in sea water. And finally, the program will check the entered opacity is in the correct range.

Program Requirements

Begin with your Alice file from Alice programming assignment 3.

Modify the fish's initial X position to be 0.0.

Modify the Swimmer **calcPressure** function method to account for the fact that sea water is denser than fresh water.

- At the top of the function, create a variable to hold the value of the ***pressure to add*** per foot of water depth, and initialize it to the fresh water value of 0.433 used previously.
- Have the Swimmer ask the user whether it is in sea water (in your prompt, tell the user to how to enter the answer) and store the answer in a variable.
- Add an **if** statement to check the user's answer
    - If the user says the fish is in sea water, change the value stored in the ***pressure to add*** variable to be 0.445 psi per foot (pressure for sea water).
    - Otherwise, do not change anything.
- Modify your pressure calculation to use the ***pressure to add*** variable you created in the math equation.

Modify **myFirstMethod**:

- Delete the first statement that made the fish turn left.
- Replace the deleted statement with statements to:
    - Have the fish say that it can move left or right.
    - Prompt the user whether to choose whether the fish should move towards the left or towards the right (in your prompt, tell the user to enter the word "left" or the word "right") and store their answer in a variable.
    - Add **if/else** code to check the user's answer.
        - If the answer was left, turn the fish right (left from the viewer's perspective is right from the fish's perspective).
        - Otherwise, turn the fish left.
- Next add code to prompt the user to enter whether the fish should loop (via scene 1) or zig zag (via scene 2) and read the answer as 1 or 2. (NOTE: the prompt should tell the user what entering 1 means, and what entering 2 means).
- Store the user's answer in a variable.

- Add **nested if/else** code to check the user's answer.
    - Call the method for the correct scene, chosen by the user.
        - If the answer did not match one of the options, have the Swimmer say that the answer was invalid and that it will not do either action.

  NOTE:
  doScene3 should still be called from **myFirstMethod**, no matter what the user chooses to do for scenes 1 and 2.

Modify the **doScene1** method:

- Modify the variable that holds the fish's depth. Instead of being a random number, between 1 and 100, set it to be the fixed value of 10.0.
- ***After*** prompting for and reading the user's input of the distance to loop,
  but ***before*** the call to the **loopDeLoop** procedure,
  add **nested if** code to check the loop distance entered by the user.
    - If the distance entered is over 4:
        - Tell the user that there may not be enough room for that size loop
        - Prompt the user, asking if they want to enter a different value (in your prompt, tell the user how to enter his/her answer).
            - If they say that they do want to enter a different value,
              read and store the new value back into the same loop distance variable.
            - Otherwise, just use the original loop distance entered by the user
    - Otherwise, if the distance entered is 4 or less, just continue.
- Execute the rest of the original statements in the method to call the **loopDeLoop** procedure, etc.

Modify the **doScene2** method:

- Add **if** code to the end of the method to:
    - Check the total distance the fish swam.
    - If the distance is 2 or less
        - Make the fish say something and then zig zag again, using the same arguments to the **zigZag** procedure as before.
    - Otherwise, do nothing else

  NOTE: When testing, you will have to modify your object's height to see if this code works.

Modify the **doScene3** method:

- Add an **if/else** statement to check the user's answer is between 0 and 1, inclusive of 0 and 1.
    - If so, call **changeAppearance** for both objects with the user's input as the argument (same as before)
    - If not:
        - Have the fish say that the opacity cannot be set to the value entered, and so the opacity will not be changed.
        - Call **changeAppearance** for both objects with an argument of 1 so the opacity will not change.

Run, test, and debug your Alice program, until it works correctly.

Submission

This programming assignment is due by midnight of the date listed on the Assignments page.

Submit your program source code (the **.a3p** file) to the **Alice Prog Assn 4** dropbox (located under the **Dropbox** tab in the online course).

Before submitting your program file, you MUST re-name it as follows:

`LastnameAliceAssn4.a3p`

For example: **`SmithAliceAssn4.a3p`**

Grading

The rubric that will be used to grade your program is linked on the same assignment page that you downloaded this file from.

WARNING:
Programs submitted more than 5 days past the due date will ***not*** *be accepted,*
and will receive a grade of 0.