CS180 Lab 7 - Robot Lab 3: Conditionals and Loops Part II

Lab for week: 7
Lab created by: James Wilkinson

Learning objectives

a. Learn how to display instructions to a user with a time delay
b. Learn how to use the random number generator
c. Learn how to use the `switch` statement
d. Learn how to create a compound condition
e. Learn how to calibrate the Finch's light sensors

Files

`Lab07.student.tar.gz`

Useful Information

1. `Finch()` - Class that controls the Finch robot. Some methods that are important to this lab are below. More can be found at: `http://www.finchrobot.com/javadoc/index.html`
    a. `getLeftLightSensor()` - Returns an integer from 0 - 255 indicating the light level seen by that sensor. 0 is no light and 255 is maximum light. The light sensors are the small circular holes on top of the finch that look like there is a zig-zag of wires inside. Left and right are relative to the Finch as the Finch's beak is pointing away from you.
    b. `getRightLightSensor()` - Same as `getLeftLightSensor()` except used for the right light sensor
    c. `isLeftLightSensor(int limit)` - If the left light sensor detects light **greater** than `limit`, return `true`. Otherwise return `false`.
    d. `isRightLightSensor(int limit)` - Same as above except for the right light sensor
    e. `sleep(int sleepTime)` - Sleeps (pauses execution) for `sleepTime` ms.
    f. `quit()` - Closes the Finch connection. Use at the end of the program.
2. `Switch-Case` - This can be used instead of a series of `if-else if` statements. This is useful if the variable being tested is an `int` and you are testing for exact matches. If you want to test for inequalities, `if` blocks would probably be a better fit. The syntax is as follows (note that you don't necessarily have to use `0` and `1` for cases, those are just examples):

```
        switch(variable){
              case 0:
                    System.out.println("variable is 0.");
                    break;
              case 1:
                    System.out.println("variable is 1.");
                    break;
              default:
                    System.out.println("variable is neither 0 nor
1.");
                    break;
        }
```

Setup

1. Create a `Lab07` directory in your finch directory
2. Extract the `Lab07.tar.gz` file contents into your `Lab07` directory

Coding Part I

You may code all of this in the main() method.

1. Welcome the user to the program, for example, "Welcome to Finch Reflexes!", sleep for 500ms and then change the beak's color to white.
2. Instruct the user to make sure that both light sensors are uncovered, sleep for 2 seconds and then get base light levels from the left and right sensors.
3. Instruct the user to cover the left light sensor, change the beak to blue, sleep for 2 seconds and then get a covered light level from the left sensor.
4. Instruct the user to cover the right light sensor, change the beak to red, sleep for 2 seconds and then get a covered light level from the right sensor.
5. Calculate the midpoint between the covered and base levels for each the left and right light sensors. Use these as the limits for checking to see if the light sensors are covered. *Example, if the value for uncovered left light is 100 and the light level for covered left is 50, then use 75 as the test number for whether the left sensor is covered.*
6. Tell the user to get ready to play, repeatedly flash the beak from white to black for 2 seconds.

Coding Part II

1. Create a `while` loop that continues until the user gets a reaction incorrect.
2. In the loop, randomly select an integer (0 or 1) corresponding to a direction, (left or right.)
3. If left, set the beak's color to `(0, green, 255)`. If right, set the beak's color to `(255, green, 0)`. You may use either a switch block or if block to do this. The `green` hue should be set to an integer ranging from 0 - 255 where 0 corresponds to `speed` = 0 and 255 to `speed` = 1000 with a linear correlation for any values in between.

Example: if the `speed` *is 100, then the* `green` *hue should be 255*(100/1000) = 25 or 26 (you may round the decimal however you'd like.)*

4. After (1000 - `speed)` milliseconds, check if the user covered **only** the correct sensor. If so, then the game should continue and `speed` should increase by 10. If the user covered both, neither or the wrong sensor, the game should end.
5. When the user gets a reaction incorrect, display the final score which should be the `speed` and also set the beak color to `(0, green, 0)` with green calculated in the same way as above.
6. Have the game sleep for 2.5 seconds and then quit.

Demonstration

1. Start program up and show that it properly goes through the calibration sequence
2. The Finch beak should flash and indicate that the game is starting
3. The Finch should alternate beak colors corresponding to left or right sensors
4. The blue/red colors should become lighter due to the green hue as game progresses
5. If an incorrect sensor is covered, the game should end displaying the score and a corresponding green hue.

Grading

Criteria	Points
Correct calibration sequence [5 points]	
Beak randomly changes between red and blue [5 points]	
Beak gets lighter from the green hue as game progresses [5 points]	
Game quits when wrong sensor covered [4 points]	
Game quits when both sensors covered [4 points]	
Game quits when no sensors covered [4 points]	
Game speeds up as time progresses [5 points]	
At end of game, beak glows shade of green corresponding to score for 2.5s [5 points]	

At end of game, correct score is displayed [5 points]	
Total Points [42 points]	

<End of CS 180 Lab 7>