Unit 00 – Guitar Hero Project

Description: Over the next two months, you will recreate the popular video game "Guitar Hero". Use the class overview below to write your program. While you may work together in order to overcome struggles with programing, each student's program must be unique – any act of academic dishonesty (i.e. plagiarism, cheating, copying, allowing another student to copy or falsify documentation of any kind) is grounds for failure.

`Note` Class

- In general, notes fall from the top of the screen towards the bottom of the screen, disappearing when they hit the bottom of the screen.
- Notes do not need to keep track of any information.
- Notes, when built, need to be given information that will determine what color they are.
- Trick! If you name your `Note` pictures something like `note1.png`, `note2.png`, `note3.png`, … then you have a single integer come in to the constructor and set your image to "`note`" + the integer + "`.png`" and it will work for all colors.
- For a better look, the notes can fall at an angle. This is achievable pretty easily by using the `SmoothMover` class from Greenfoot.

`Button` Class

- In general, a single button (fret?) has a keyboard key associated with it that when pressed will make the button look like it is pressed down, and if strummed it will try to kill an intersecting note.
- A `Button` needs to keep track of which keyboard key is associated with the button.
- A `Button`, when built, need to be given information that will determine what color they are and will also determine what keyboard key they are associated with.
- Similar to `Note` class, the `Button` class could have a single integer come in which will determine what picture to use, but can also be used to set the keyboard key to "`F`" + integer to make the keys be `F1`, `F2`, `F3`, `F4`, `F5`.

`World` Class

- In general, the world needs to turn some music on, place the buttons on the screen, and slowly generate notes randomly on the top of the screen.
- The `World` can also display a score which the `Note` and `Button` classes can update if a `Note` is correctly or incorrectly pressed.
- The `World` should keep track of a number to represent how many acts have gone by.
- The `World` should generate a new `Note` after some set amount of acts (to keep a beat, this could be done a perfect number of acts, like every 50 acts, by using modulus).
- When generating a `Note`, the `World` should be able to randomly generate a 1, 2, 3, 4, or 5. Based on that number, it would be that type of `Note`, then place that note on the correct location on the screen (simple math formula can be used to figure out exactly where to place objects without using a bunch of if statements).
- To add variety, you can make double `Notes` appear.
- To add variety, you can make notes appear on a specific beat, but some beats get skipped. For example, if notes appear every 20 acts, some of those could be randomly skipped.

Deadline: <u>Thursday, October 19th or Friday, October 20th</u>