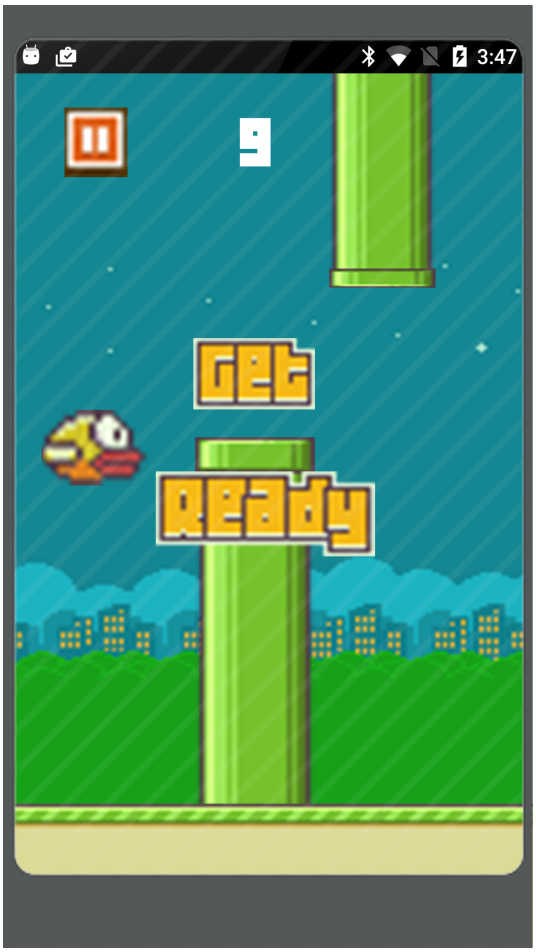
Test 2 - Flappy Bird Clone - 5% (3 hours)

Today your task is to recreate the hit mobile game ***Flappy Bird***, within the given time of the class. This is considered a **test**, so you cannot share code or work in groups. All work must be done individually. This is an open book test, so feel free to use previous labs as guidance. **Failure to submit the test by the end of the class will result in a zero.**

Download the lab materials on Blackboard. You may use the starter project on Blackboard named **LabTest.zip** or create a fresh LibGDX project and use the **classes** and **images** inside **LabTest.zip**

NOTE: Before starting, review the changes made to the core game classes. I've added collision and physics methods in ActorBeta.java so take a look at those. *(Take a look at **ActorBeta.java, ScreenBeta.java, LevelOne.java, Plane.java**)*

Using the given classes within the project and the YouTube video as reference, you must **try your best to recreate an exact replica of this version of Flappy Bird (https://www.youtube.com/watch?v=fQoJZuBwrkU)**, which includes:



1. **Using the assets provided, create** the start screen using a Screen class. Once the player presses start, the game should begin. For now, score button does not have to do anything. **You will be awarded full marks for using a repeating action to replicate the title hover animation used here. *(HINT: look at the Youtube video for reference at the 0:10 mark)* (0.5)**

2. When the game starts, quickly show the instructions for about 3 seconds **(*HINT*: look at the Youtube video for reference at the 0:12 mark) (0.25)**

3. The bird must be animated and jumping like how he does in Flappy Bird. I have provided a function called **boost()** that can get you started with this. ***(HINT: You may need to adjust the function's values slightly to make it more like the real game)* (0.5)**

4. All background layers in the scene must scroll infinitely at a pace similar to the real game ***(including the pipes and ground)*** **(0.5)**

5. Using the assets provided create pipes. The pipes positions should be spawned horizontally with different Y values as the game progresses. ***(HINT: look at the Youtube video for reference)*** **(0.5)**

6. Using the assets provided create a score sprite. Score should increase each time the bird successfully passes a pipe. ***(HINT: Use the number sprites provided for this)*** **(0.25)**

7. A sound should play each time the bird passes a pipe. ***(HINT: You can use the sparkle.mp3 sound located in assets)*** **(0.25)**

8. A sound should play when the bird collides with the pipe or hits the ground. I don't mind id you use the same sound or find a different one from a previous lab. **(*HINT*: The logic to perform collisions is in the ActorBeta class, use one of the methods that performs overlapping) (0.25)**

9. Once the bird collides with a pipe the game will end. Using the assets provided you should replicate the game over screen. You will be awarded full points for using a sequence and/or parallel actions to re-create how this. ***(HINT: look at the Youtube video for reference at the 0:48 mark)*** **(0.5)**

10. Using the assets provided create a pause/resume button. This button should be located at the top left of the screen and should allow me to pause the game. Once paused, the pause button should become a resume button so I can resume, and vice versa. ***(HINT: The logic to pause the game is located somewhere within the ScreenBeta class)*** **(0.5)**

11. When the game is ended, pressing the Okay button will restart the game. You should add the share button here too button it doesn't have to do anything for now. **(*HINT*: You do not have to restart the game from the very beginning) (0.5)**

12. Using the assets provided display the best score the player has gotten so far, and the score they got during the previous attempt. ***(HINT: look at the Youtube video for reference at the 0:48 mark)*** **(0.5)**

Submit this as StudentName_StudentNumber.zip. Make sure code is running and compiling before submitting. Code submitted with compiler errors will result in a zero. Anything submitted past the due time will not be accepted.