**INSTRUCTOR SHEET**

**\*\* Optional Steps \*\***

Step 0.1: Getting a screen to show up (just a black one that dissappears)

if \_\_name\_\_ == "\_\_main\_\_":

game = FlappyBird()

updateScreen()

Step 0.2: Add a background (optional strings are “night” and “day”)

if \_\_name\_\_ == "\_\_main\_\_":

game = FlappyBird()

game.loadBackground("day")

updateScreen()

Step 0.3: Make sure screen doesn’t close (you will need to use Ctrl-C to close it)

if \_\_name\_\_ == "\_\_main\_\_":

game = FlappyBird()

game.loadBackground("day")

while True:

updateScreen()

Step 0.4: Add the bird (options are “red”, “blue”, “yellow”)

if \_\_name\_\_ == "\_\_main\_\_":

game = FlappyBird()

game.loadBackground("day")

game.loadBird("red")

while True:

updateScreen()

**\*\*\* Steps Decided for now \*\*\***

Step 1: Getting a Game name to show up ()

'''This is a String Varible'''

## This variable controls the name of the game - you can  change its value choose the name of the game

game\_name = "Flappy Bird"

Step 2: Getting the background to show ()

'''This is a String Varible'''

## This variable controls the background of the game - you can  change its value choose between the night and day background

game\_background = "day"

Step 3: Getting the bird color to show ()

'''This is a String Varible'''

## This variable controls the color of the bird - you can  change its value choose between the different color of the bird

game\_color = "red"

Step 4: Setting the speed of the game ()

'''This is a Integer Varible'''

## This variable controls speed of the game - you can  change its value to make the game go faster or slower

game\_speed = 3

Step 5: Setting the length of the gap between the pipes of the game

'''This is a Integer Varible'''

## This variable controls length of the gap between the pipes in the game - you can  change its value to make the height of the gap larger or smaller between pipes

pipe\_gap = 3

Step 6: User interaction (Allowing for the user to escape the game using the escape button or just clicking the red button)

The following

'''Conditional Statements'''

## write a conditional statement that allows the user to escape the game use from API following functions - escapePressed, closeGame

def check\_escape(buttonsPressed):

   ##if escapePressed(buttonsPressed):

   ##closeGame()

Step 7: User interaction (Allowing for the user to click and make the bird jump)

## write a conditional statement that allows the user to make the bird fly when mouse is clicked API following functions - mouseClick, birdNotDead, birdJump

## note - remeber to check if the bird is dead in the if statement

def click\_to\_move(game, buttonsPressed):

   ##if mouseClick(buttonsPressed) and game.birdNotDead():

   ##game.birdJump()

Step 8: Get collision detection (Allowing for the user to see the score change for each pipe passed by the bird) \* (if you don’t add the *and* statement at the end of if mouseClicked the bird will “die” but keep going)

## write a conditional statement that checks weather the bird passed the wall and increament the game.score variable API following functions - wallPassed, birdNotDead

## note - remeber to check if the bird is dead in the if statement

def score\_update(game):

   ##if game.wallPassed() and game.birdNotDead():

   ##game.score = game.score + 1 #This can be a fcn but we think its good for them to learn

Step 9: Allowing for the bird to die when it hits the pipe

## write a conditional statement that checks weather the bird hit one of the pipes  API following functions - checkHitBottomPipe, checkHitBottomPipe

## note - two condition in the conditional statment

## advance use or to combine two conditional statments

def Pipe\_hit(game):

   ##if game\_end\_bottom == True or game\_end\_top == True:

   ##game.over = True