**INSTRUCTOR SHEET**

Step 1: Getting a screen to show up (just a black one that dissappears)

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

game = FlappyBird()

updateScreen()

Step 2: Add a background (optional strings are “night” and “day”)

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

game = FlappyBird()

game.loadBackground("day")

updateScreen()

Step 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 4: Add the bird (options are “red”, “blue”, “yellow”)

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

game = FlappyBird()

game.loadBackground("day")

game.loadBird("red")

while True:

updateScreen()

Step 5: User interaction (make the bird jump on click, and close on escape key or X button)

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

game = FlappyBird()

game.loadBackground("day")

game.loadBird("red")

clock = game.getClock()

while True:

clock.tick(60)

buttonsPressed = checkWhichButtonsPressed()

if escapePressed(buttonsPressed):

closeGame()

if mouseClick(buttonsPressed):

game.birdJump()

game.flap()

updateScreen()

Step 6: Get some walls up and moving (the variable *wallGap* determines the height between pipes and *gameSpeed* determines how quickly the pipes come at you). There is no death checking yet.

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

game = FlappyBird()

gameSpeed = 2

wallGap = 150

game.loadBackground("day")

game.loadBird("red")

game.loadWalls(wallGap)

clock = game.getClock()

while True:

clock.tick(60)

buttonsPressed = checkWhichButtonsPressed()

if escapePressed(buttonsPressed):

closeGame()

if mouseClick(buttonsPressed):

game.birdJump()

game.flap()

game.updateWalls(gameSpeed)

updateScreen()

Step 7: Get collision detection/losing working (if you don’t add the *and* statement at the end of if mouseClicked the bird will “die” but keep going)

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

game = FlappyBird()

gameSpeed = 2

wallGap = 150

game.loadBackground("day")

game.loadBird("red")

game.loadWalls(wallGap)

clock = game.getClock()

while True:

clock.tick(60)

buttonsPressed = checkWhichButtonsPressed()

if escapePressed(buttonsPressed):

closeGame()

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

game.birdJump()

game.flap()

game.updateWalls(gameSpeed)

if game.checkHitBottomPipe() == True:

game.over = True

if game.checkHitTopPipe() == True:

game.over = True

if game.birdOffScreen():

game.resetGame()

updateScreen()

Step 8: Add scoring

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

game = FlappyBird()

gameSpeed = 2

wallGap = 150

game.loadBackground("day")

game.loadBird("red")

game.loadWalls(wallGap)

clock = game.getClock()

while True:

clock.tick(60)

buttonsPressed = checkWhichButtonsPressed()

if escapePressed(buttonsPressed):

closeGame()

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

game.birdJump()

game.flap()

game.updateWalls(gameSpeed)

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

game.score = game.score + 1

updateScoreDisplay(game)

if game.checkHitBottomPipe() == True:

game.over = True

if game.checkHitTopPipe() == True:

game.over = True

if game.birdOffScreen():

game.resetGame()

updateScreen()

Step 9/10: Add game over, and game start screens

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

game = FlappyBird()

gameSpeed = 2

wallGap = 150

game.loadBackground("day")

game.loadBird("red")

game.loadGameStart()

checkForStart(game)

game.loadWalls(wallGap)

clock = game.getClock()

while True:

clock.tick(60)

buttonsPressed = checkWhichButtonsPressed()

if escapePressed(buttonsPressed):

closeGame()

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

game.birdJump()

game.flap()

game.updateWalls(gameSpeed)

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

updateScoreDisplay(game)

if game.checkHitBottomPipe() == True:

game.over = True

if game.checkHitTopPipe() == True:

game.over = True

#If bird out of bounds

if game.birdOffScreen():

game.loadGameOver()

newGameCheck(game)

updateScreen()