# print(“Python!”)

By Leanna Szypowski

# Why Python?

- Curious to learn something new
- Lots of neat extensions (*PyGame*)
- Previous failed attempt

```
        print("2 is equal to 2")

indents are needed to define blocks of code, instead of {}

variables

x = 5
y = "Hello!"

no need to define what type it is (int, float etc, but you can if you do this)

x = int(5)
y = str("Hello!")

strings are known as str unlike both C# and Java (string and String respectively)
#this is how comments work
x, y, z = 1, 2, 3 #declare multiple variables at once

lists are like arrays and can hold different data types
numbers = [1, 2, 3]
testList = ["string", true, 55]
functions/methods

def testFunction():
        print("this is a function")

to call the function
testFunction()

arrays are lists (as stated above)
also dict? Dictionaries I assume. Should learn more about them

len() returns the length of a string
x = "Hi"
print(len(x))
#will print out 2

for loop
#this prints every letter in the word letters
```

- Syntax the first day -> 3.9.2 Documentation (python.org)
- Lack of ; and { }
- Say hello to *indents* and :!
- Note: Concatenation sucks

# First “Program”

- Basics: Numbers from console -> + list
- Start of a calculator (which I make later on)
- Getting used to basic syntax

```
# declares lists that hold numbers and operations
numbers = []
operations = []

# takes in a number and adds it to the list
numInput = float(input())
numbers.append(numInput)

# takes in an operation and adds it to the list
operationInput = str(input())
operations.append(operationInput)

numOfInputs = 0

while operationInput != "=":
    # takes in input and saves it to numbers list
    numInput = float(input())
    numbers.append(numInput)


    # takes in input and saves it to numbers list
    operationInput = str(input())
    operations.append(operationInput)
    # adds one to number of inputs
    numOfInputs += 1
```

# Program #1 - Calculator:

- Made calculator
- Any amount of values & operations
- Basic list manipulation (*del, append() etc.*)
- Calculator didn't know BEDMAS

*(more like DMAS but still)*

# Program #1 – Calculator - Improved

```
answer = 0
while numOfInputs != 0:
    x = 0
    current = 0
    while operations[x] != "*" and operations[x] != "/" and x != numOfInputs - 1:
        x += 1

    # does all * and / before + and -
    if operations[x] == "*":
        current += numbers[x] * numbers[x + 1]
        del numbers[x]
        del numbers[x]
        i = operations[x]
        del operations[x]
        operations.append(i)
        numOfInputs -= 1
        numbers.insert(x, current)

    elif operations[x] == "/":
        current += numbers[x] / numbers[x + 1]
        del numbers[x]
        del numbers[x]
        i = operations[x]
        del operations[x]
        operations.append(i)
        numOfInputs -= 1
        numbers.insert(x, current)

    # when all * and / have been dealt with, breaks out of loop
    elif operations[x] == "+" or operations[x] == "-":
        answer = numbers[0]
        break
```

- Spent way too long making it do BEDMAS *(ahem DMAS)*
- It functions however!
- Code is horrendous and unoptimized…
- Demo -> firstTest.py

```
36
37  # for every letter in the word, adds it to the list that stores the word
38  # and adds another slot to the guesses display
39  # if it's a space, adds a space character
40  for char in word:
41      guessWord.append(char)
42      if char != ' ':
43          displayGuesses.append('_')
44      else:
45          displayGuesses.append(' ')
46
47  # 0 = game is not complete, 1 is its done
48  complete = 0
49
50  # while game is on
51  while complete == 0:
52      # clears console
53      os.system('cls')
54
55      # displays guesses (aka combines list into string)
56      print(' '.join(displayGuesses))
57
58      # displays amount of guesses before death and asks used for guess
59      txt = 'Wrong guesses before death: {}'
60      print(txt.format(totalHanged - hanged))
```

# Program #2 – Hangman

- + A proper hangman display
- + “Bad letters” display
- Later -> added different difficulties
- Demo -> hangman(thegame).py

```
50  # while game is on
51  while complete == 0:
52      # clears console
53      os.system('cls')
54
55      # displays guesses (aka combines list into string)
56      print(' '.join(displayGuesses))
57
58      # displays amount of guesses before death and asks used for guess
59      txt = 'Wrong guesses before death: {}'
60      print(txt.format(totalHanged - hanged))
61
62      # displays current state of "hangman" depending on wrong guesses and difficulty
63      if hanged == 0:
64          print('''
65          _______
66          |/      |
67          |
68          |
69          |
70          |
71          |
72          ''')
73      elif hanged == 1:
74          print('''
75          _______
76          |/      |
77          |      (_)
78          |
79          |
80          |
81          |
82          ''')
83      elif hanged == 2 and difficulty != 2 or hanged == 3 and difficulty == 2:
84          print('''
85          _______
```

Progr

Learnt

Qu

```
# declares tic-tac-toe board
slot = ["[1]", "[2]", "[3]", "[4]", "[5]", "[6]", "[7]", "[8]", "[9]"]

# declares whether someone has won or not
winner = 0

# function to check if there is a winner
def checkForWinner():
    # specifies to use the global variable
    global winner
    # checks to see is there are 3 in a row horizontally
    if slot[0] == slot[1] == slot[2]:
        winner = slot[0]
    elif slot[3] == slot[4] == slot[5]:
        winner = slot[3]
    elif slot[6] == slot[7] == slot[8]:
        winner = slot[6]

    # checks to see is there are 3 in a row vertically
    if slot[0] == slot[3] == slot[6]:
        winner = slot[0]
    elif slot[1] == slot[4] == slot[7]:
        winner = slot[1]
    elif slot[2] == slot[5] == slot[8]:
        winner = slot[2]

    # checks to see is there are 3 in a row diagonally
    if slot[0] == slot[4] == slot[8]:
        winner = slot[0]
    elif slot[2] == slot[4] == slot[6]:
        winner = slot[2]

```


pygame

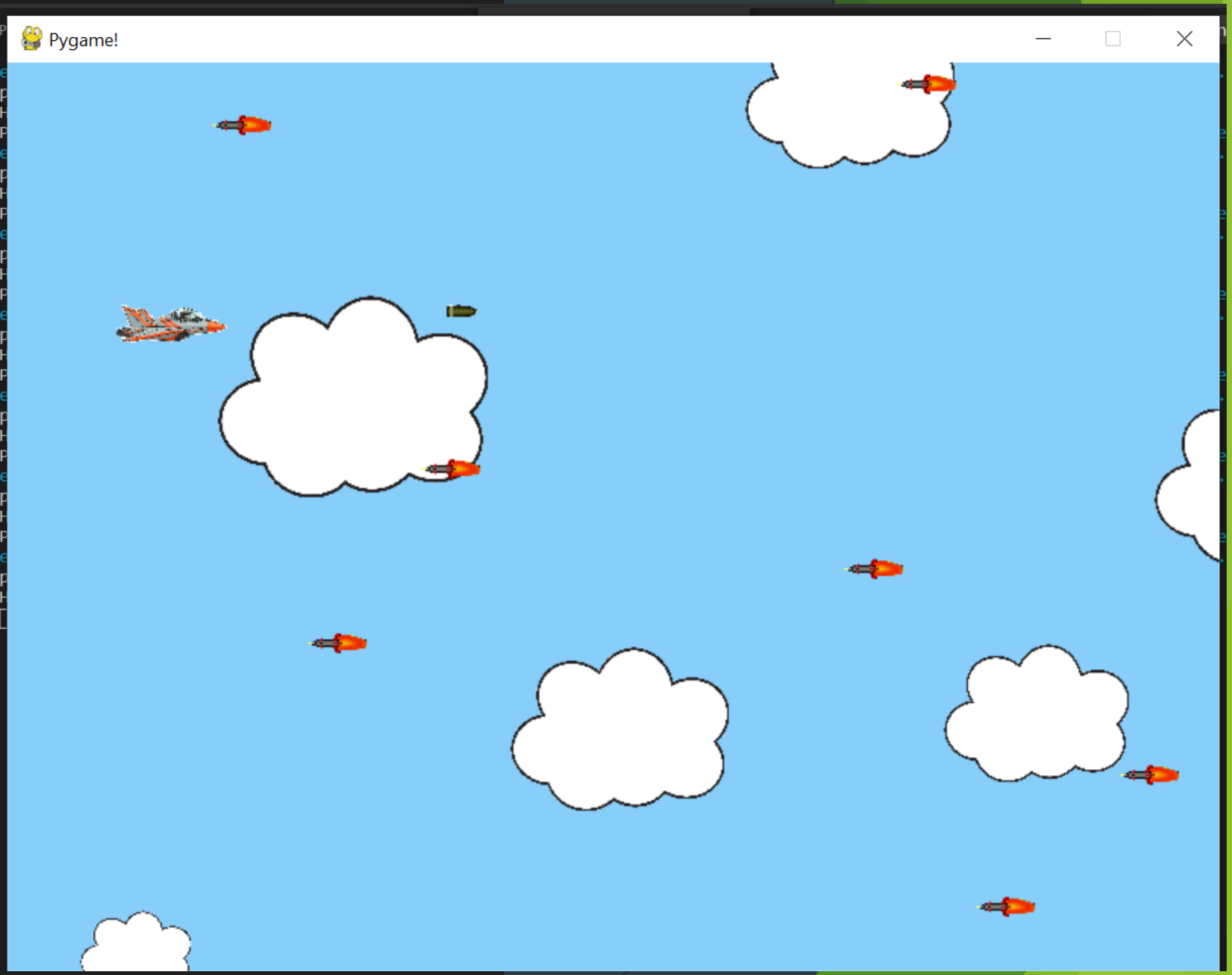
# Learning PyGame:

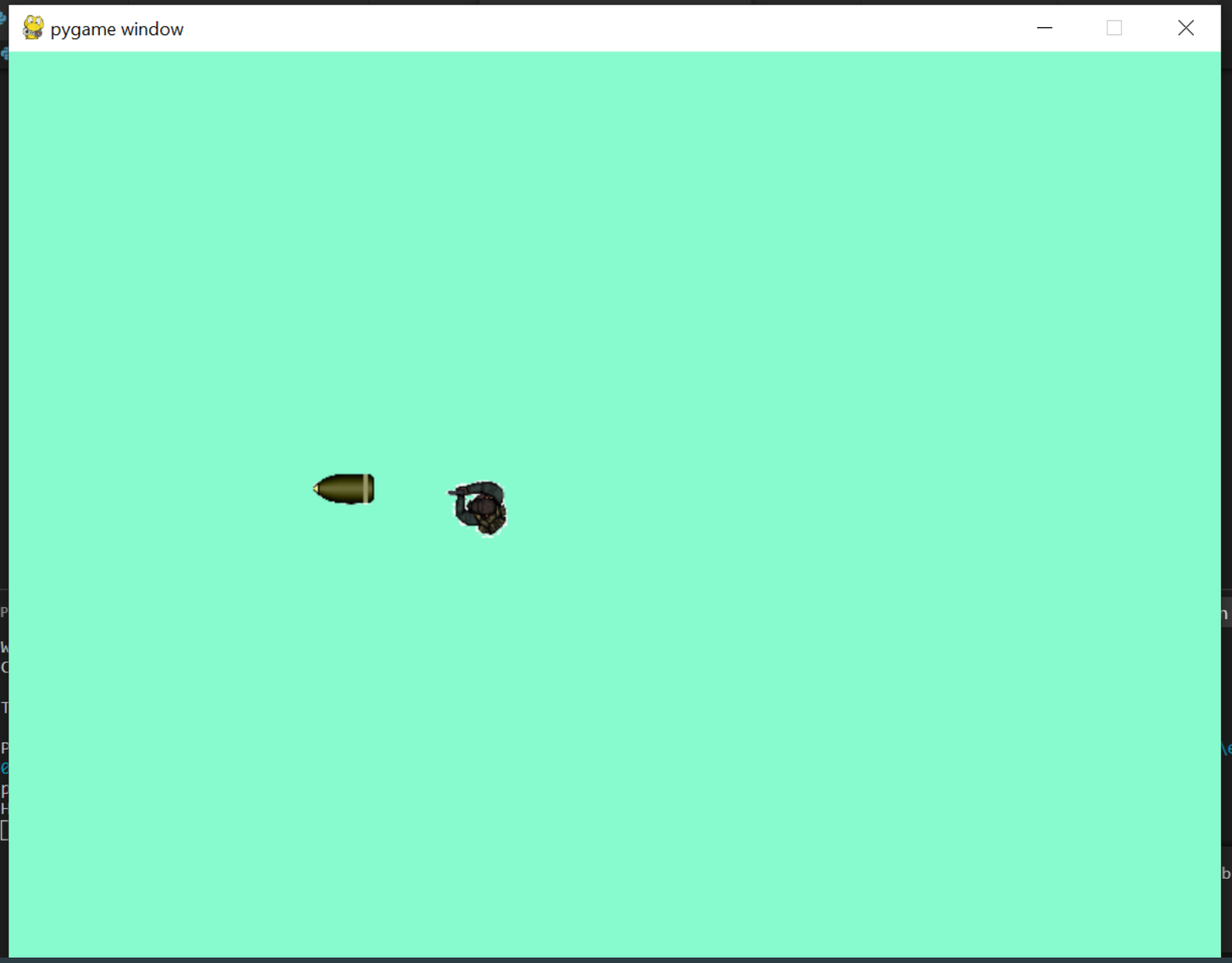
- Downloaded PyGame via Microsoft PowerShell
- Similar to unity
- Learnt extremely basic Pygame concepts

# Program #4 – Ship vs Missiles

- Made my first real game using PyGame
- Followed a tutorial to learn the basics -> PyGame: A Primer on Game Programming in Python – Real Python
- Learnt about classes + big game concepts
- Again, similar concepts to Unity *(with differences of course)*

```
136                 if self.rect.right < 0:
137                         self.kill()
138
139
140 class Projectile(pygame.sprite.Sprite):
141         # runs once when initalized
142         def __init__(self):
143                 # allows access
144                 super(Projectile, self).__ini
145                 # declares surface and loads
146                 self.surf = pygame.image.load
147                 # scales image
148                 self.surf = pygame.transform.
149                 # removes image background
150                 self.surf.set_colorkey(white,
151                 # sets rect depending on play
152                 self.rect = self.surf.get_rec
153                 self.rect.y = player.rect.y
154                 self.rect.x = player.rect.x +
155
156         # runs every frame
157         def update(self):
158                 # moves projectile to the rig
159                 self.rect.x += 7.5
160                 # if object moves offscreen,
161                 if self.rect.left > screenWid
162                         self.kill
163
164
165 # initializes pygame mixer (for better custom
166 pygame.mixer.init()
167 # initializes pygame
168 pygame.init()
169
```






```
__init__()
image.load('player.jpg').convert()
transform.scale(self.surf, (50, 50))
elf.surf
key(white, RLEACCEL)
f.get_rect()
Width / 2
Height / 2

):
ding on keys pressed
]:
ion = 0
ve_ip(0, -3)
 pygame.transform.rotate(self.frozenSurf, 90)
N]:
ion = 2
ve_ip(0, 3)
 pygame.transform.rotate(self.frozenSurf, -90)

T]:
ion = 3
ve_ip(-3, 0)
 pygame.transform.rotate(self.frozenSurf, 180)

HT]:
ion = 1
ve_ip(3, 0)
 pygame.transform.rotate(self.frozenSurf, 0)


reen
 0:
ft = 0
 screenWidth:
```

# Any Questions?