Lesson 5: Practice Thinking Logically

What programming "tools" have we learned?

- Variables:
  - Integers, floats, strings
  - Lists and dictionaries
- If-else conditions
- Loops - while and for loops
- Functions - parameters, return values, default arguments
- Libraries - using free code someone else wrote by importing it into our code

Today's goal: Work on our logical thinking skills. We want to be used to thinking about solutions to problems in ways that can translate easily to code → we do this by thinking one step at a time.

We might use some library functions we haven't learned yet, but no new major concepts will be introduced.

We want to break the habit of jumping immediately into the code. We want to first know the solution and then translate that to code.

For each problem, first write "pseudocode" using comments in your Python file **and then write the code.** Pseudocode is basically English that represents what the code will do**.**

Some pseudocode for typing on a keyboard might look like this:

```
# Save the user settings

# if a key is pressed, then:

#       check the user's language preference

#       letter = language[key]

#        input correct letter

# otherwise:

#      dont do anything
```

Problems

1. You have a string like “12/05” which represents a date on the calendar. Your job is to make another string from this one that contains the month and day (i.e. for this “December 5”). **Hint:** remember we have a way of splitting strings into pieces.

2. You have two variables, called “x” and “y”. Your job is to swap them. If x=5 and y=6, when your code is done running, x=6 and y=5.

3. Write a function **def contains(some_list, item)** that returns True if the item is in that list, otherwise it returns False.
    a. Could your approach change if I told you the list is sorted from lowest to highest?

4. Write a function with signature **def fib(n):** that returns the n’th Fibonacci number. Fibonacci numbers are the sequence that goes: 1, 1, 2, 3, 5, 8, 13, 21, … (starts with 1’s and then adds the previous 2 to get the next).

5. I know what price I bought bitcoin at. I want a program to tell me how much profit/loss I have made. If I enter “2000” that means I bought it at $2000 USD. Assume you have a dictionary that contains a key ‘price’, whose value is a string of the current bitcoin price.

    {'trade_id': 160406939, 'price': '50061.38', 'size': '0.00017922', 'time': '2021-04-23T18:08:13.710476Z', 'bid': '50059.01', 'ask': '50061.38', 'volume': '44396.44568113'}

    a. If you run in terminal `**pip install requests**` you can then use the following code to get the live price (it will set ‘r’ to the dictionary above).

```
import requests

r = requests.get('https://api.pro.coinbase.com/products/BTC-USD/ticker').json()
```

6. Write a program that has some list of numbers and a variable called NUM_ROTATIONS. “Rotate” the list NUM_ROTATIONS times. So if the list is [1, 2, 3] and NUM_ROTATIONS=1, your job is to create a list [3, 1, 2]. **Hint: In python you can “slice” lists… my_list[0:2] would give me the first 2 elements.** Think about how you could rotate it 1 time and then generalize that.

7. We want to make a ball bounce up and down. If you **pip install pygame** and run this, you should see a ball move off the screen. Change the program so the ball bounces up and down off the floor.

```
import pygame
pygame.init()

WINDOW_WIDTH, WINDOW_HEIGHT = 500, 1000
screen = pygame.display.set_mode([WINDOW_WIDTH, WINDOW_HEIGHT])

running = True

CIRCLE_COLOR = (0, 0, 255)
CIRCLE_RADIUS = 50
CIRCLE_X = 250
CIRCLE_Y = 250

X_VELOCITY = 0.01
Y_VELOCITY = -0.01

while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    screen.fill((255, 255, 255))
    pygame.draw.circle(screen, CIRCLE_COLOR, (CIRCLE_X, CIRCLE_Y), CIRCLE_RADIUS)
    pygame.display.flip()

    '''
    Update the ball position !
    '''
    CIRCLE_X += X_VELOCITY
    CIRCLE_Y += Y_VELOCITY

pygame.quit()
```

8. We now want the ball to bounce diagonally off the walls. X_VELOCITY and Y_VELOCITY will be set to some float and your job is to update the code so that when the ball hits the edge of the windows, it changes direction so it never leaves the window.