1. In this Exercise, you learned how to use if-elif-else statements to run different tasks based on conditions that you define. Now practice that skill by writing a script for a simple travel app using an if-elif-else statement for the following situation:

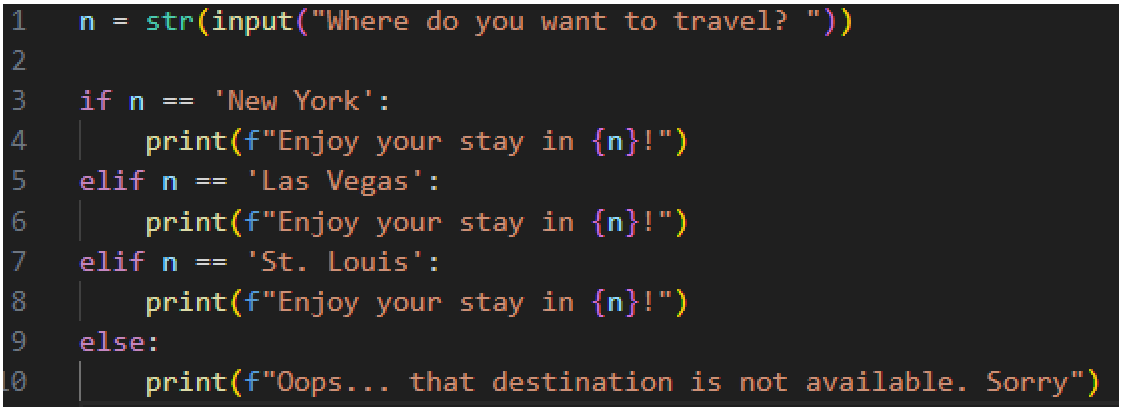
- The script should ask the user where they want to travel.

- The user's input should be checked for 3 different travel destinations that you define.

- If the user's input is one of those 3 destinations, the following statement should be printed: "Enjoy your stay in ______!"

- If the user's input is something other than the defined destinations, the following statement should be printed: "Oops, that destination is not currently available."

Write your script here. (Hint: remember what you learned about indents!)



```
n = str(input("Where do you want to travel? "))

if n == 'New York':
    print(f"Enjoy your stay in {n}!")
elif n == 'Las Vegas':
    print(f"Enjoy your stay in {n}!")
elif n == 'St. Louis':
    print(f"Enjoy your stay in {n}!")
else:
    print(f"Oops... that destination is not available. Sorry")
```

2. Imagine you're at a job interview for a Python developer role. The interviewer says "Explain logical operators in Python". Draft how you would respond.

As with all programming languages, logical operators are a fundamental tool that provide routing to programs. If a user's input or some form of data can dictate the information or data that is served, this can better utilize business resources. Operators will compare two or more things, and can also string together multiple comparisons for complex event cases. Whether something is 'equal to' or 'greater/less than', or the negatives of these- NOT 'equal to' or 'greater/less than.' These operators can split resources to efficiently provide information or direct data down a path that reduces total overhead due to not needing to waste time.

3. What are functions in Python? When and why are they useful?

Functions in Python store code that is often reused throughout a program. Instead of defining an operation repeatedly, Python lets the programmer use a ‘def’ block to condense these otherwise repeated chunks down to one instance of the chunk. These functions can be called and can return data back to each specific call, often given parameters to begin with.

4. In the section for Exercise 1 in this Learning Journal, you were asked in question 3 to set some goals for yourself while you complete this course. In preparation for your next mentor call, make some notes on how you’ve progressed towards your goals so far.

I have learned how to write with loops and how to properly access attributes within lists and dictionaries, but I should make sure that I understand notation for this completely instead of guessing the notation and telling myself that I’ll fix it later.