Exercise 1.3: Functions and Other Operations in Python

Learning Goals

- Implement conditional statements in Python to determine program flow
- Use loops to reduce time and effort in Python programming
- Write functions to organize Python code

Reflection Questions

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!)*

```
user_preference = input("Do you want to travel? Enter 'Y' for yes and 'N' for no: ")

If user_preference.upper() == 'Y':
  destination = input("Where would you like to travel? ")
  If destination.capitalize() == "NYC" or "LA" or "Chicago":
    print("Enjoy your stay in " + destination)
  else:
    print("Oops, that destination is not currently available.")
else:
  print("Thank you for checking!")
```

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.

Ans: Logical operators in Python are tools that let you combine conditions for decision-making. The and operator is used to check if both conditions are true, the or operator checks if at least one condition is true, and the not operator negates the truth value of a condition. These operators are often used in if statements to control the flow of your program based on various conditions.

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

   Ans: Functions in Python are like mini-programs that perform specific tasks, enabling code reuse and organization. They simplify complex operations, take inputs (parameters), and can return outputs. By encapsulating logic into functions, code becomes cleaner, more maintainable, and easier to test.

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.

   Ans: I learned that Python's readability, extensive libraries, cross-platform adaptability, and versatility across domains like web development and data analysis make it a favored language. Its user-friendly syntax, rapid development support, and strong community add to its appeal.