Exercises for Introduction to Python Part II

Programming II - KI08
Deggendorf Institute of Technology
Prof. Dr. Florian Wahl

Exercise 1: If-then-else

- Store the numbers 1 through 9 in a list.
- Loop through the list.
- Use an if-elif-else chain inside the loop to print the proper ordinal ending for each number. Your output should read "1st 2nd 3rd 4th 5th 6th 7th 8th 9th", and each result should be on a separate line.

Exercise 2: Favorite Numbers

Use a dictionary to store people’s favorite numbers. Think of five names, and use them as keys in your dictionary. Think of a favorite number for each person, and store each as a value in your dictionary. Print each person’s name and their favorite number. For even more fun, poll a few friends and get some actual data for your program.

Exercise 3: Rivers

Make a dictionary containing three major rivers and the country each river runs through. One key-value pair might be 'nile': 'egypt'.

- Use a loop to print a sentence about each river, such as The Nile runs through Egypt.
- Use a loop to print the name of each river included in the dictionary.
- Use a loop to print the name of each country included in the dictionary.

Exercise 4: Rental Car

Write a program that asks the user what kind of rental car they would like. Print a message about that car, such as “Let me see if I can find you a Subaru.“

Exercise 5: Pizza Toppings

Write a loop that prompts the user to enter a series of pizza toppings until they enter a 'quit' value. As they enter each topping, print a message saying you’ll add that topping to their pizza.

Exercise 6: Number guessing

Use the random number generator to get a number and then ask the user to guess the number. The loop ends after 5 tries of when the user entered the correct number.

In a second version add hints whether to go higher or lower.

In [ ]:
```
# Hint: This is how to generate a random number

import random
random_number = random.randint(1, 10)
```

Exercise 7: Cars

Write a function that stores information about a car in a dictionary. The function should always receive a manufacturer and a model name. It should then accept an arbitrary number of keyword arguments. Call the function with the required information and two other name-value pairs, such as a color or an optional feature. Your function should work for a call like this one:

`car = make_car('subaru', 'outback', color='blue', tow_package=True)`

Print the dictionary that's returned to make sure all the information was stored correctly.

Exercise 8: Restaurant (we do this one together)

Make a class called Restaurant. The `__init__()` method for Restaurant should store two attributes: a `restaurant_name` and a `cuisine_type`. Make a method called `describe_restaurant()` that prints these two pieces of information, and a method called `open_restaurant()` that prints a message indicating that the restaurant is open. Make an instance called `restaurant` from your class. Print the two attributes individually, and then call both methods. `Create three different instances from the class, and call` describe_restaurant()` for each instance.

Exercise 9: Restaurant extended

Add an attribute called `number_served` with a default value of 0. Create an instance called restaurant from this class. Print the number of customers the restaurant has served, and then change this value and print it again. Add a method called `set_number_served()` that lets you set the number of customers that have been served. Call this method with a new number and print the value again. Add a method called `increment_number_served()` that lets you increment the number of customers who've been served. Call this method with any number you like that could represent how many customers were served in, say, a day of business.

Exercise 10: Ice cream stand

An ice cream stand is a specific kind of restaurant. Write a class called `IceCreamStand` that inherits from the `Restaurant` class. Add an attribute called `flavors` that stores a list of ice cream flavors. Write a method that displays these flavors. Create an instance of `IceCreamStand`, and call this method.