Python Programming Basics: Series 3.1

Functions and Dictionary Iteration Continued

Table of contents

1 Introduction

Welcome to Series 3.1 of our Python programming journey! In this series, we'll continue exploring:

- **Functions**: Reinforcing how to define and call them in scripts.
- **Dictionary Iteration**: Practicing how to iterate through dictionaries using `.keys()`, `.values()`, and `.items()` with `for` loops.

We'll build upon what we've learned in previous series and solidify our understanding through practical exercises.

Let's get started!

---

2 Functions and Dictionaries: A Quick Recap

Before we dive into the exercises, let's revisit some key concepts about functions and dictionaries.

2.1 Defining and Calling Functions

A **function** is a reusable block of code that performs a specific task. Functions help us:

- Organize code.
- Make code reusable.
- Improve readability.

2.1.1 Defining a Function

```python
def function_name(parameters):
    # Function body
    return result
```

- **`def`**: Keyword to define a function.
- **`function_name`**: Name of the function.
- **`parameters`**: Inputs to the function (optional).
- **`return`**: Outputs a value from the function (optional).

2.1.2 Calling a Function

```python
result = function_name(arguments)
```

2.2 Dictionaries and Iteration

A **dictionary** is a collection of key-value pairs.

2.2.1 Iterating Over a Dictionary

- **Keys**: Use `.keys()` to get all the keys.
- **Values**: Use `.values()` to get all the values.
- **Items**: Use `.items()` to get key-value pairs.

2.2.2 Example

```python
student_grades = {
    "Alice": 85,
    "Bob": 92,
    "Charlie": 78
}

# Iterating over keys
for name in student_grades.keys():
    print(name)

# Iterating over values
for grade in student_grades.values():
    print(grade)

# Iterating over key-value pairs
for name, grade in student_grades.items():
    print(f"{name}: {grade}")
```

```
Alice
Bob
Charlie
85
92
78
Alice: 85
Bob: 92
Charlie: 78
```

---

3 Exercises

3.1 Exercise 1: Calculating Total Price

Question

You are given a dictionary `shopping_cart` where the keys are item names and the values are the prices of the items.

```
shopping_cart = {
    "book": 12.99,
    "pen": 1.49,
    "notebook": 4.99,
    "eraser": 0.99
}
```

3.1.1 Part A

- Write a function named `calculate_total` that:
  - Accepts a dictionary of items and their prices.
  - Returns the total price of all items.

3.1.2 Part B

- In the main script:
  - Call the `calculate_total` function with `shopping_cart`.
  - Print the total price in the format: `"Total price: $X.XX"`.

3.1.3 Part C

- Iterate over the `shopping_cart` dictionary and print each item and its price in the format: `"- Item: $Price"`.

Hints:

- Use `for` loops to iterate over the dictionary.
- Use `+=` to accumulate the total price.
- Format the price to two decimal places using `f"{price:.2f}"`.

```
shopping_cart = {
    "book": 12.99,
    "pen": 1.49,
    "notebook": 4.99,
    "eraser": 0.99
}

# Part A: Define the function calculate_total
def calculate_total(cart: dict) → float:
    # Function body
    ?
```

```python
# Main script
if __name__ == "__main__":
    # Part B: Call the function and print total price
    total_price = ?
    print(f"Total price: ${?}")

    # Part C: Iterate and print each item and its price
    for item, price in ?:
        print(f"- {item}: ${?}")

    # Expected output:
    # - book: $12.99
    # - pen: $1.49
    # - notebook: $4.99
    # - eraser: $0.99
```

3.2 Exercise 2: Analyzing Student Grades

Question

You have a dictionary `student_grades` where the keys are student names and the values are their grades.

```python
student_grades = {
    "Alice": 88,
    "Bob": 72,
    "Charlie": 95,
    "Diana": 85,
    "Evan": 79
}
```

3.2.1 Part A

- Write a function named `calculate_average` that:
  - Accepts a dictionary of student grades.
  - Returns the average grade.

3.2.2 Part B

- In the main script:
  - Call the `calculate_average` function with `student_grades`.
  - Print the average grade in the format: `"Average grade: X.XX"`.

3.2.3 Part C

- Iterate over `student_grades` and print the names of students who scored above the average, in the format: `"- Student Name: Grade"`.

Hints:

- To calculate the average, store the total of all grades in a variable and divide by the number of grades.
- Use an `if` statement inside the `for` loop in the main script to check if the grade is above average.
- Format the average to two decimal places using `f"{average:.2f}"`.

```python
student_grades = {
    "Alice": 88,
    "Bob": 72,
    "Charlie": 95,
    "Diana": 85,
    "Evan": 79
}

# Part A: Define the function calculate_average
def calculate_average(grades: dict) -> float:
    # Function body
    ?

# Main script
if __name__ == "__main__":
    # Part B: Call the function and print average grade
    average_grade = calculate_average(student_grades)
    print(f"Average grade: ?")

    # Part C: Iterate and print students who scored above average
    print("Students who scored above average:")
    for student, grade in student_grades.items():
        if ?:
            print(f"- {student}: {grade}")
    # Expected output:
    # Average grade: 83.80
    # Students who scored above average:
    # - Alice: 88
    # - Charlie: 95
    # - Diana: 85
```

---

4 Conclusion

In this follow-up series, we've reinforced our understanding of:

- **Functions**: Writing functions that accept dictionaries as parameters and return computed values.
- **Dictionary Iteration**: Practicing iterating over dictionaries to access keys, values, and items.

By working through these exercises, you've gained more experience in handling dictionaries and functions, which are essential skills in Python programming.

---

5 Additional Resources

- **Python Official Documentation**:
  - Defining Functions
  - Dictionaries
- **W3Schools Python Tutorial**:
  - Python Functions
  - Python Dictionaries

---

6 Acknowledgments

We hope this series has helped you strengthen your Python programming skills. Keep practicing, and continue exploring the exciting world of programming!

---

7 End of Series 3.1

Keep up the excellent work, and stay curious!