Day 6: Data Structures - Dictionaries

Overview of Dictionaries

- **Definition**: A dictionary is an unordered, mutable, and indexed collection of key-value pairs in Python.
- **Syntax**:
- dictionary = {key1: value1, key2: value2, ...}
- **Characteristics**:
  - Keys must be unique and immutable (e.g., strings, numbers, or tuples).
  - Values can be of any data type and can be duplicated.
  - Dictionaries are dynamic and can be nested.

Creating and Accessing Dictionaries

1. **Creating a Dictionary**:
2. my_dict = {"name": "John", "age": 25, "city": "New York"}
3. **Accessing Values**:
   - Using keys:
   - print(my_dict["name"]) # Output: John
   - Using get() method (avoids KeyError):
   - print(my_dict.get("age")) # Output: 25
   - print(my_dict.get("country", "Not Found")) # Output: Not Found

Updating and Modifying Dictionaries

- **Adding New Key-Value Pairs**:
- my_dict["country"] = "USA"
- **Updating Existing Values**:
- my_dict["age"] = 26
- **Removing Items**:
- my_dict.pop("city") # Removes the key 'city'

- del my_dict["age"] # Deletes 'age'
- my_dict.clear()   # Clears all items in the dictionary

Dictionary Methods

1. **keys()**: Returns a view object of all keys in the dictionary.
2. print(my_dict.keys()) # Output: dict_keys(['name', 'age'])
3. **values()**: Returns a view object of all values in the dictionary.
4. print(my_dict.values()) # Output: dict_values(['John', 25])
5. **items()**: Returns a view object of key-value pairs (as tuples).
6. print(my_dict.items()) # Output: dict_items([('name', 'John'), ('age', 25)])

Iterating Through Dictionaries

1. **Keys Only**:
2. for key in my_dict.keys():
3.   print(key)
4. **Values Only**:
5. for value in my_dict.values():
6.   print(value)
7. **Key-Value Pairs**:
8. for key, value in my_dict.items():
9.   print(f"{key}: {value}")

Examples and Exercises

1. **Example**:
2. student = {
3.   "name": "Alice",
4.   "age": 22,
5.   "courses": ["Math", "Science"]
6. }

7.

8. # Accessing data

9. print(student["name"])   # Output: Alice

10. print(student.get("grade", "Not Found")) # Output: Not Found

11.

12. # Adding data

13. student["grade"] = "A"

14.

15. # Iterating

16. for key, value in student.items():

17.   print(f"{key}: {value}")

18. **Exercise 1**: Create a dictionary of 5 countries and their capitals. Print all keys, values, and key-value pairs.

19. **Exercise 2**: Write a program to count the frequency of characters in a string using a dictionary.

20. **Exercise 3**: Create a dictionary that stores students' names as keys and their scores as values. Find the student with the highest score.

Summary

- Dictionaries are versatile and efficient for storing and accessing data using key-value pairs.
- Methods like keys(), values(), and items() make dictionary traversal and manipulation straightforward.
- Practice by solving real-life problems like contact management, inventory tracking, or frequency analysis.

Let me know if you'd like this structured differently or need additional exercises!