**Module 9 –** Dictionaries

Contents

[Dictionaries 1](#_heading=h.c8mgltvjc224)

[9.1 What Is a Dictionary? 1](#_heading=h.6kzp7iih62n1)

[9.2 Accessing Values 1](#_heading=h.hr7qvl8a6yn1)

[9.3 Updating a Dictionary 2](#_heading=h.mhkxy22040em)

[9.4 Dictionary Methods 2](#_heading=h.5ja9292lodr2)

[9.5 Looping Through a Dictionary 2](#_heading=h.ovwy8fqq36id)

[9.6 Nested Dictionaries 3](#_heading=h.a1ctovuwlgb)

[9.7 Sorting a Dictionary 3](#_heading=h.wi23e0yj5gkv)

# Dictionaries

[Dictionaries](https://www.w3schools.com/python/python_dictionaries.asp) are an essential data type in Python. They allow you to store key-value pairs, making data retrieval and association easy and efficient.

## 9.1 What Is a Dictionary?

A [dictionary](https://docs.python.org/3/tutorial/datastructures.html#dictionaries) in Python is a mutable, unordered collection that maps unique keys to values. Keys must be immutable types (like strings, numbers, or tuples), and values can be any type. You can also use the dict() [constructor](https://docs.python.org/3/tutorial/datastructures.html#dictionaries).

|  |
| --- |
| person = {“name”: “Neda”, “age”: 30, “city”: “Savannah”}  info = dict(name=”Bob”, age=25) |

## 9.2 Accessing Values

You can [access a value](https://www.geeksforgeeks.org/python/python-accessing-key-value-in-dictionary/) by referring to its key:

|  |
| --- |
| print(person[“name”]) # Output: Alice |

Using [.get()](https://www.w3schools.com/python/ref_dictionary_get.asp) is safer when you're unsure if the key exists:

|  |
| --- |
| print(person.get(“job”, “Not specified”)) # Output: Not specified |

## 9.3 Updating a Dictionary

You can change the value of an existing key or [add a new key-value pair](https://www.w3schools.com/python/python_dictionaries_add.asp):

|  |
| --- |
| person[“age”] = 31  person[“email”] = “neda@gsu.edu” |

To [remove items](https://www.w3schools.com/python/gloss_python_remove_dictionary_items.asp):

|  |
| --- |
| person.pop(“city”) # Removes “city”  del person[“name”] # Deletes “name” |

## 9.4 Dictionary Methods

Common built-in dictionary methods:

get(): Returns value for key or default if not found

|  |
| --- |
| person = {"name": "Alice", "age": 25}  print(person.get("age")) # Output: 25  print(person.get("city", "N/A")) # Output: N/A (default used) |

keys(): Returns view of all keys

|  |
| --- |
| person = {"name": "Alice", "age": 25}  print(person.keys()) # Output: dict\_keys(['name', 'age']) |

values(): Returns view of all values

|  |
| --- |
| person = {"name": "Alice", "age": 25}  print(person.values()) # Output: dict\_values(['Alice', 25]) |

items(): Returns view of key-value pairs

|  |
| --- |
| person = {"name": "Alice", "age": 25}  print(person.items()) # Output: dict\_items([('name', 'Alice'), ('age', 25)]) |

update(): Merges another dict

|  |
| --- |
| person = {"name": "Alice", "age": 25}  person.update({"age": 26, "city": "Savannah"})  print(person) # Output: {'name': 'Alice', 'age': 26, 'city': 'Savannah'} |

pop(): Removes item by key

|  |
| --- |
| person = {"name": "Alice", "age": 25}  age = person.pop("age")  print(age) # Output: 25  print(person) # Output: {'name': 'Alice'} |

clear(): Removes all items

|  |
| --- |
| person = {"name": "Alice", "age": 25}  person.clear()  print(person) # Output: {} |

len(): Return the number of items in the dictionary

|  |
| --- |
| person = {"name": "Alice", "age": 25}  print(len(person)) # Output: 2 |

## 9.5 Looping Through a Dictionary

You can [iterate through keys, values, or both](https://www.geeksforgeeks.org/python/iterate-over-a-dictionary-in-python/):

|  |
| --- |
| for key in person:  print(key, person[key])  for key, value in person.items():  print(key, value)  for value in person.values():  print(value)  for key in person.keys():  print(key) |

## 9.6 Nested Dictionaries

A [nested dictionary](https://www.w3schools.com/python/python_dictionaries_nested.asp) is a dictionary inside another dictionary:

|  |
| --- |
| users = {  “Neda”: {“age”: 25, “email”: [neda@gsu.edu](mailto:neda@gsu.edu)},  “Elizabeth”: {“age”: 30, “email”: [elizabeth@gsu.edu](mailto:elizabeth@gsu.edu)}  } |

## 9.7 Sorting a Dictionary

You can [sort a dictionary](https://www.freecodecamp.org/news/python-sort-dictionary-by-key/) by values or by keys.

|  |
| --- |
| #sort by value  my\_dict = {‘b’: 2, ‘a’: 1, ‘c’: 3}  sorted\_dict\_byValue = dict(sorted(my\_dict.items(), key=lambda x:x[1]))  print(sorted\_dict\_byValue)  #sort by key  my\_dict = {‘b’: 3, ‘a’: 2, ‘c’: 1}  sorted\_dict\_byKey = dict(sorted{my\_dict.items()))  print(sorted\_dict\_byKey) |

Programming Exercises:

1. **Create a Dictionary:**

Create a dictionary called student with keys: "name", "age", and "major". Print the dictionary.

1. **Access Dictionary Values**

Using the given dictionary:

person = {"name": "Joe", "age": 30, "city": "Savannah"}

Print Joe’s city using his key and use the get() method to access a key called “zip code”. Return “N/A” if it doesn’t exist.

1. **Update Values**

Use the given dictionary:

car = {"brand": "Toyota", "year": 2015}

Change the year to 2020, add a new key “color” with the value “black” and print the updated dictionary.

1. **Remove Items**

Use the given dictionary:

person = {"name": "Joe", "age": 30, "city": "Savannah"}

Use the pop() method to remove the city and del to remove the age, then print the dictionary.

1. **Use Additional Methods**

Use the given dictionary:

pet = {"type": "cat", "name": "Darcy", "age": 5}

Print all keys using .keys(). Print all values using .values(). Print all key-value pairs using .items(). Print how many items are in the dictionary using len().

1. **Merge Dictionaries**

Merge the following dictionaries and print the result:

dict1 = {"a": 1, "b": 2}

dict2 = {"b": 3, "c": 4}

1. **Loop Through Dictionary**

Given the following dictionary:

student = {"name": "Beth", "age": 21, "grade": "A"}

Use a loop to print each key and value on a new line.

1. **Clear a Dictionary**

Create any dictionary, print it, then clear it using .clear(). Print it again to confirm it’s empty.

1. **Nested Dictionaries**

Create a dictionary named library that contains two books, each represented by a nested dictionary. Print the author of Book2, add a third book, then loop through all books and print their titles

1. **Dictionary Sorting**

Given the following dictionary:

grades = {"Jane": 85, "Alice": 92, "Harry": 78, "Tom": 90}

Sort the dictionary by values (the numeric grades) from lowest to highest, then print the dictionary. Next, sort the dictionary by keys (student names) alphabetically and print.

**Programming Exercise Solutions:**

**1.**

|  |
| --- |
| student = {"name": "Gabi", "age": 22, "major": "Education"}  print(student) |

**2.**

|  |
| --- |
| person = {"name": "Joe", "age": 30, "city": "Savannah"}  print(person["city"])  print(person.get("zip code", "N/A")) |

**3.**

|  |
| --- |
| car = {"brand": "Toyota", "year": 2015}  car["year"] = 2020  car["color"] = "black"  print(car) |

**4.**

|  |
| --- |
| person = {"name": "Joe", "age": 30, "city": "Savannah"}  person.pop("city")  del person["age"]  print(person) |

**5.**

|  |
| --- |
| pet = {"type": "cat", "name": "Darcy", "age": 5}  print(pet.keys())  print(pet.values())  print(pet.items())  print(len(pet)) |

**6.**

|  |
| --- |
| dict1 = {"a": 1, "b": 2}  dict2 = {"b": 3, "c": 4}  dict1.update(dict2)  print(dict1) |

**7.**

|  |
| --- |
| student = {"name": "Beth", "age": 21, "grade": "A"}  for key, value in student.items():  print(key, ":", value) |

**8.**

|  |
| --- |
| car = {"brand": "Honda", "model": "Civic", "year": 2019}  print("Before clear:", car)  car.clear()  print("After clear:", car) |

**9.**

|  |
| --- |
| library = {  "Book1": {"title": "1984", "author": "George Orwell"},  "Book2": {"title": "Dune", "author": "Frank Herbert"}  }  print(library["Book2"]["author"])  library["Book3"] = {"title": "Pride and Prejudice", "author": "Jane Austen"}  for book, info in library.items():  print(info["title"]) |

**10.**

|  |
| --- |
| grades = {"Jane": 85, "Alice": 92, "Harry": 78, "Tom": 90}  sorted\_by\_value = dict(sorted(grades.items(), key=lambda x: x[1]))  print("Sorted by grade:", sorted\_by\_value)  sorted\_by\_key = dict(sorted(grades.items()))  print("Sorted by name:", sorted\_by\_key) |