**Dictionary Methods in Python**

**1. Accessing Elements**

* **dict.get(key, default=None)**  
  Returns the value for a key if it exists; otherwise, returns the specified default value.

my\_dict = {"name": "Alice", "age": 25}

print(my\_dict.get("age")) # Output: 25

print(my\_dict.get("city", "Not Found")) # Output: Not Found

**2. Adding and Updating Elements**

* **dict[key] = value**  
  Adds a new key-value pair or updates the value for an existing key.

my\_dict = {"name": "Alice"}

my\_dict["age"] = 25 # Adding a new key-value pair

my\_dict["name"] = "Bob" # Updating an existing key

print(my\_dict) # Output: {'name': 'Bob', 'age': 25}

* **dict.update(other\_dict)**  
  Updates the dictionary with key-value pairs from another dictionary or iterable.

my\_dict = {"name": "Alice", "age": 25}

my\_dict.update({"age": 26, "city": "New York"})

print(my\_dict) # Output: {'name': 'Alice', 'age': 26, 'city': 'New York'}

**3. Deleting Elements**

* **dict.pop(key, default=None)**  
  Removes a key and returns its value. If the key doesn't exist, returns the default value.

my\_dict = {"name": "Alice", "age": 25}

age = my\_dict.pop("age")

print(age) # Output: 25

print(my\_dict) # Output: {'name': 'Alice'}

* **dict.popitem()**  
  Removes and returns the last key-value pair as a tuple. Raises an error if the dictionary is empty.

my\_dict = {"name": "Alice", "age": 25}

item = my\_dict.popitem()

print(item) # Output: ('age', 25)

* **del dict[key]**  
  Deletes a key-value pair from the dictionary.

my\_dict = {"name": "Alice", "age": 25}

del my\_dict["age"]

print(my\_dict) # Output: {'name': 'Alice'}

* **dict.clear()**  
  Removes all elements from the dictionary.

my\_dict = {"name": "Alice", "age": 25}

my\_dict.clear()

print(my\_dict) # Output: {}

**4. Retrieving Keys, Values, and Items**

* **dict.keys()**  
  Returns a view object of all keys in the dictionary.

my\_dict = {"name": "Alice", "age": 25}

print(my\_dict.keys()) # Output: dict\_keys(['name', 'age'])

* **dict.values()**  
  Returns a view object of all values in the dictionary.

print(my\_dict.values())

# Output: dict\_values(['Alice', 25])

* **dict.items()**  
  Returns a view object of all key-value pairs as tuples.

print(my\_dict.items())

# Output: dict\_items([('name', 'Alice'), ('age', 25)])

**5. Checking for Keys**

* **key in dict**  
  Checks if a key exists in the dictionary.

my\_dict = {"name": "Alice", "age": 25}

print("name" in my\_dict) # Output: True

print("city" in my\_dict) # Output: False

**6. Creating Dictionaries**

* **dict.fromkeys(iterable, value=None)**  
  Creates a dictionary with keys from an iterable and all values set to the specified value.

keys = ["a", "b", "c"]

my\_dict = dict.fromkeys(keys, 0)

print(my\_dict) # Output: {'a': 0, 'b': 0, 'c': 0}

**7. Copying a Dictionary**

* **dict.copy()**  
  Creates a shallow copy of the dictionary.

my\_dict = {"name": "Alice", "age": 25}

copy\_dict = my\_dict.copy()

print(copy\_dict) # Output: {'name': 'Alice', 'age': 25}

**8. Advanced Methods**

* **dict.setdefault(key, default=None)**  
  Returns the value of a key if it exists; otherwise, adds the key with the specified default value.

my\_dict = {"name": "Alice"}

print(my\_dict.setdefault("age", 25)) # Output: 25

print(my\_dict) # Output: {'name': 'Alice', 'age': 25}

* **dict.get(key, default=None)**  
  Retrieves the value of a key, returning a default value if the key doesn’t exist.

print(my\_dict.get("age", "Not Found")) # Output: 25