```
In [19]:   # Create a dictionary with some key-value pairs
           person = {
               "name": "Alice",
               "age": 30,
               "city": "New York"
           }

           # Print the entire dictionary
           print("Dictionary:", person)
```

Dictionary: {'name': 'Alice', 'age': 30, 'city': 'New York'}

```
In [20]:   # Create a dictionary with some key-value pairs
           person = {
               "name": "Bob",
               "age": 25,
               "city": "Los Angeles"
           }

           # Access values using keys
           name = person["name"]
           age = person["age"]

           # Print the accessed values
           print(f"Name: {name}")
           print(f"Age: {age}")
```

Name: Bob
Age: 25

```
In [21]:   # Create an initial dictionary
           person = {
               "name": "Carol",
               "age": 28
           }

           # Add a new key-value pair
           person["city"] = "Chicago"

           # Update an existing key-value pair
           person["age"] = 29

           # Print the updated dictionary
           print("Updated Dictionary:", person)
```

Updated Dictionary: {'name': 'Carol', 'age': 29, 'city': 'Chicago'}

```
In [22]:   # Create a dictionary with some key-value pairs
           person = {
               "name": "David",
               "age": 35,
               "city": "San Francisco"
           }

           # Remove a key-value pair using `del`
           del person["city"]

           # Print the dictionary after removal
           print("Dictionary after removal:", person)
```

Dictionary after removal: {'name': 'David', 'age': 35}

```
In [23]:   # Create a dictionary with some key-value pairs
           person = {
               "name": "Eva",
```

```python
        "age": 40,
        "city": "Seattle"
    }

    # Check if a key exists in the dictionary
    if "city" in person:
        print("City:", person["city"])
    else:
        print("City key not found.")
```

```
City: Seattle
```

In [24]:
```python
# Create a dictionary with some key-value pairs
person = {
    "name": "Frank",
    "age": 32,
    "city": "Boston"
}

# Iterate over keys and values
for key, value in person.items():
    print(f"{key}: {value}")
```

```
name: Frank
age: 32
city: Boston
```

In [25]:
```python
# Create a dictionary with some key-value pairs
person = {
    "name": "Grace",
    "age": 27,
    "city": "Denver"
}

# Get keys and values as lists
keys = list(person.keys())
values = list(person.values())

# Print the lists
print("Keys:", keys)
print("Values:", values)
```

```
Keys: ['name', 'age', 'city']
Values: ['Grace', 27, 'Denver']
```

In [26]:
```python
# Define two dictionaries
dict1 = {
    "name": "Hannah",
    "age": 22
}
dict2 = {
    "city": "Austin",
    "occupation": "Engineer"
}

# Merge dictionaries using the update() method
dict1.update(dict2)

# Print the merged dictionary
print("Merged Dictionary:", dict1)
```

```
Merged Dictionary: {'name': 'Hannah', 'age': 22, 'city': 'Austin', 'occupation': 'Engine
er'}
```

In [27]:
```python
# Create a dictionary with squares of numbers using dictionary comprehension
squares = {x: x ** 2 for x in range(1, 6)}
```

```python
# Print the dictionary of squares
print("Squares Dictionary:", squares)
```

Squares Dictionary: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

In [29]:
```python
# Create a dictionary with numerical values
scores = {
    "Alice": 90,
    "Bob": 85,
    "Charlie": 92
}

# Find the key with the maximum value
max_key = max(scores, key=scores.get)
max_value = scores[max_key]

# Print the result
print(f"Highest score is by {max_key} with a score of {max_value}")
```

Highest score is by Charlie with a score of 92

In [30]:
```python
# Create a dictionary with some key-value pairs
original_dict = {
    "a": 1,
    "b": 2,
    "c": 3
}

# Invert the dictionary: keys become values and values become keys
inverted_dict = {v: k for k, v in original_dict.items()}

# Print the inverted dictionary
print("Inverted Dictionary:", inverted_dict)
```

Inverted Dictionary: {1: 'a', 2: 'b', 3: 'c'}

In [ ]: