What are

DATA STRUCTURES IN PYTHON?

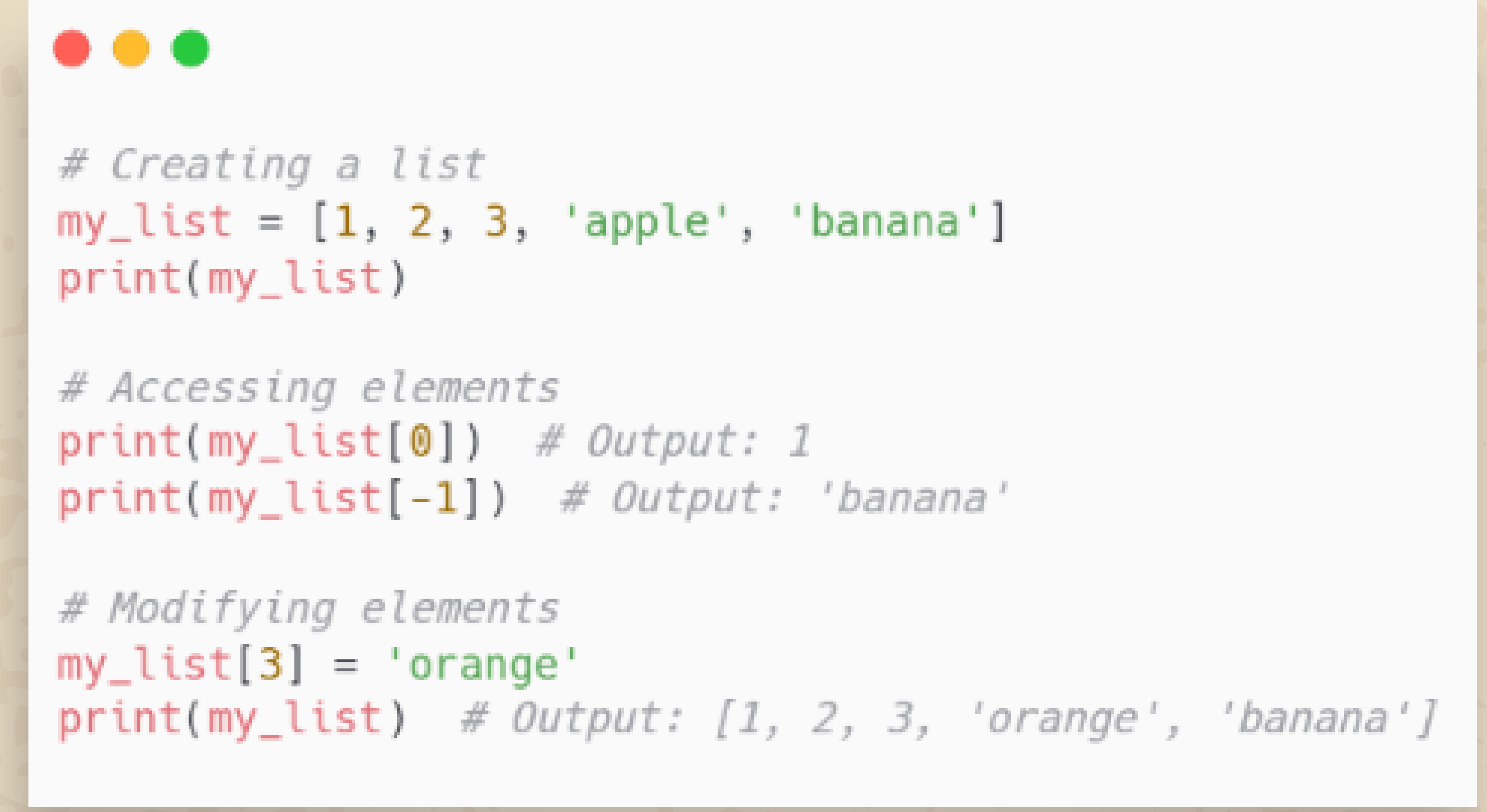
Understanding the Fundamentals

Introduction to Data Structures

Data structures are fundamental in programming. They organize and store data effectively, enabling efficient manipulation and retrieval.

LISTS

Lists are ordered collections of elements. Lists are mutable, meaning their elements can be modified after creation.



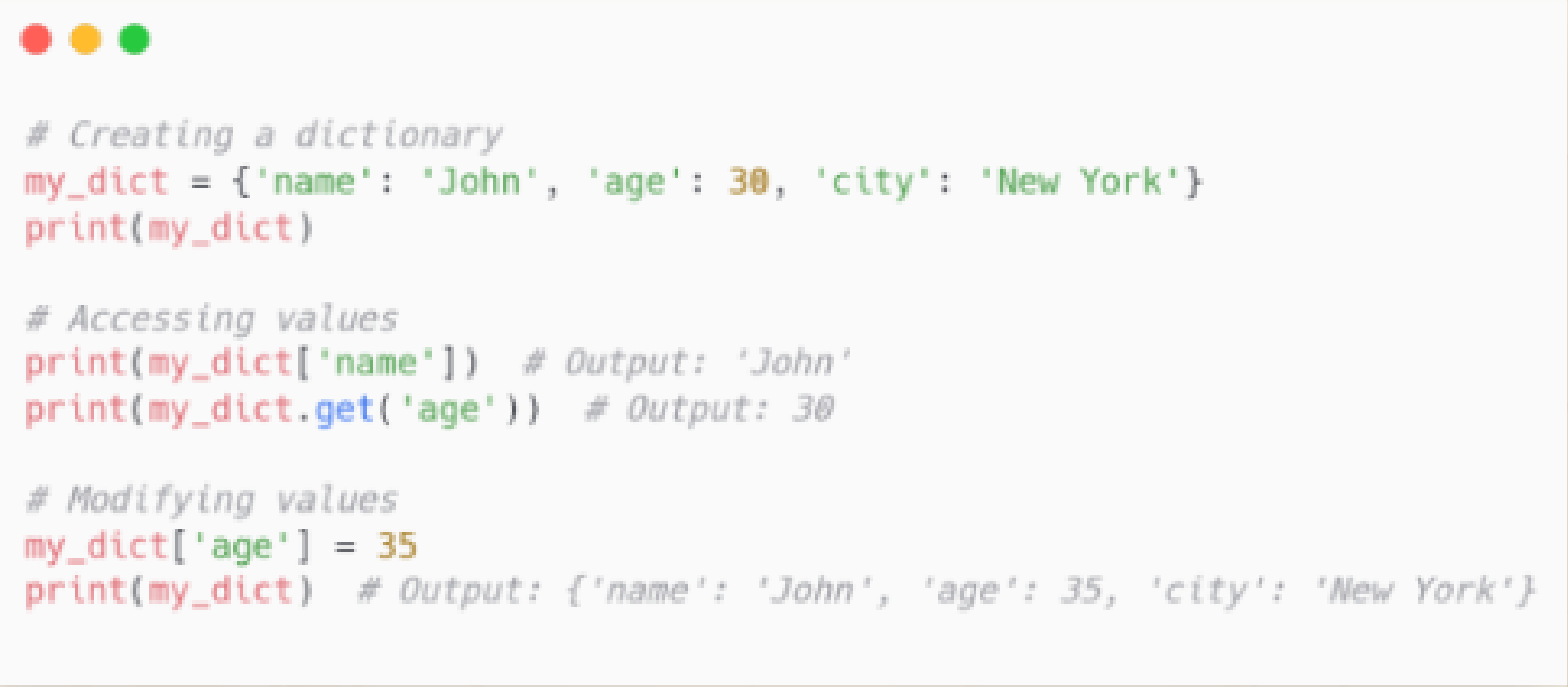
```python
# Creating a list
my_list = [1, 2, 3, 'apple', 'banana']
print(my_list)

# Accessing elements
print(my_list[0])  # Output: 1
print(my_list[-1])  # Output: 'banana'

# Modifying elements
my_list[3] = 'orange'
print(my_list)  # Output: [1, 2, 3, 'orange', 'banana']
```

DICTIONARIES

Dictionaries store key-value pairs. They are unordered collections that enable key-based actions such as element addition, access, and removal.



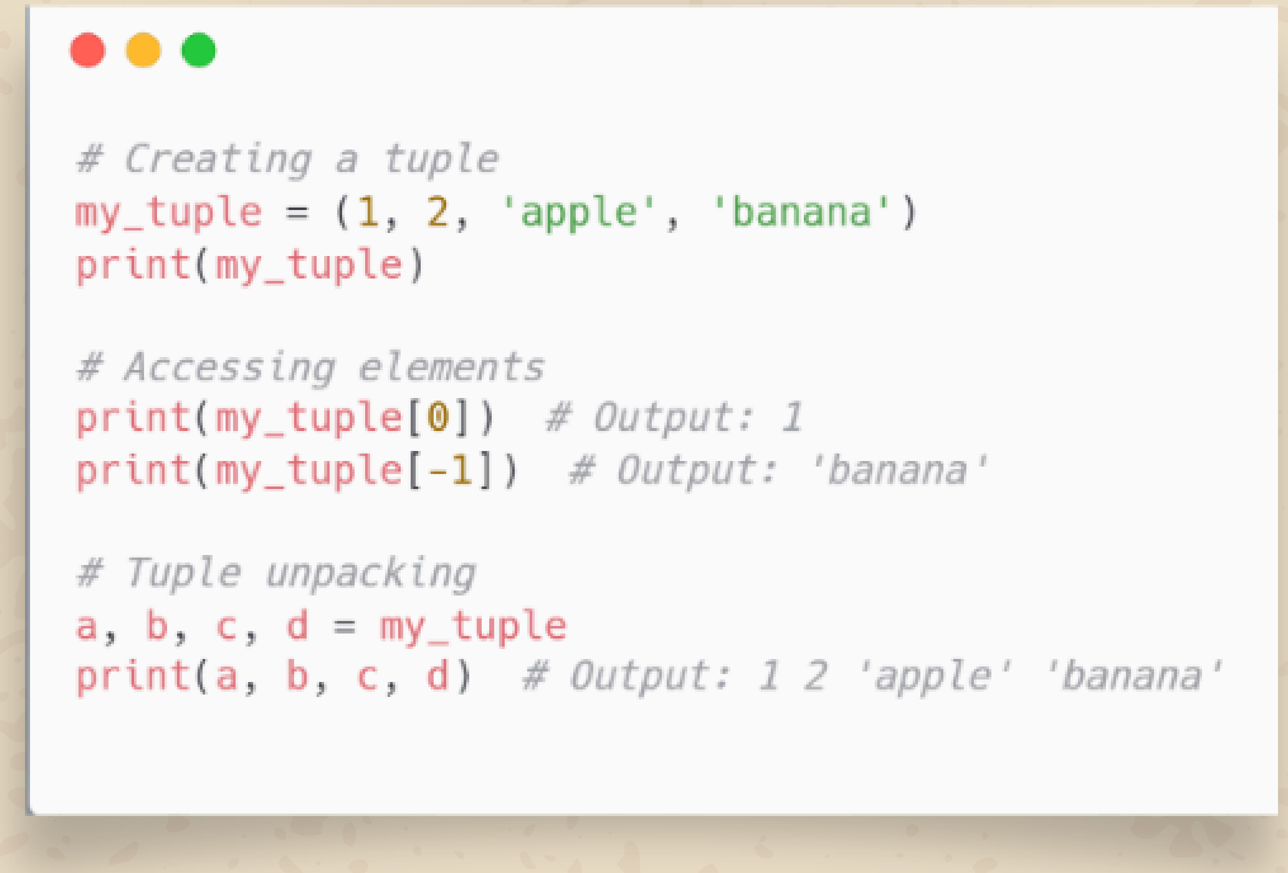
```
# Creating a dictionary
my_dict = {'name': 'John', 'age': 30, 'city': 'New York'}
print(my_dict)

# Accessing values
print(my_dict['name'])  # Output: 'John'
print(my_dict.get('age'))  # Output: 30

# Modifying values
my_dict['age'] = 35
print(my_dict)  # Output: {'name': 'John', 'age': 35, 'city': 'New York'}
```

TUPLES

Tuples are similar to lists but tuples' elements are immutable, meaning they cannot be changed or modified once they are created. Tuple is fast as compared to list because of its immutability.



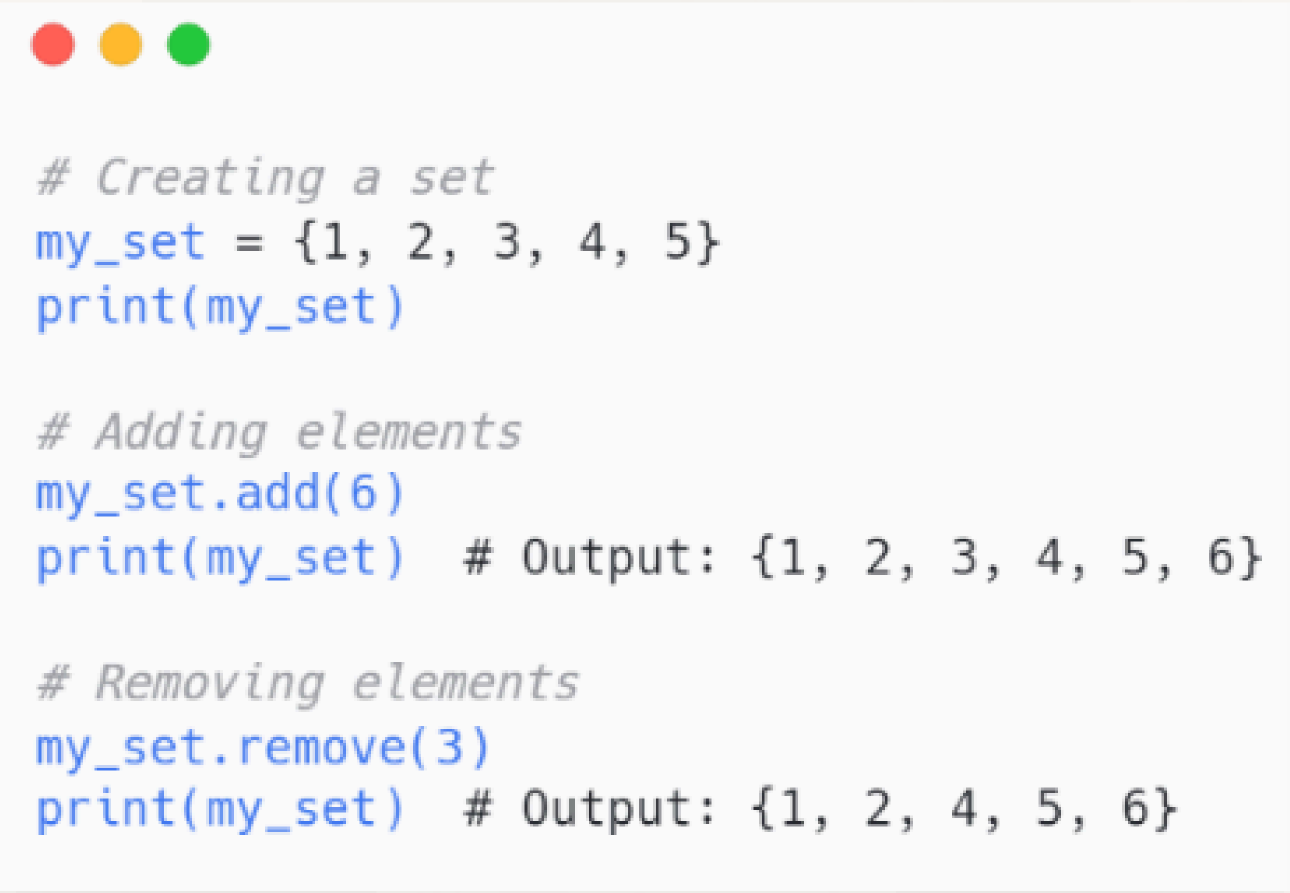
```
# Creating a tuple
my_tuple = (1, 2, 'apple', 'banana')
print(my_tuple)

# Accessing elements
print(my_tuple[0])  # Output: 1
print(my_tuple[-1])  # Output: 'banana'

# Tuple unpacking
a, b, c, d = my_tuple
print(a, b, c, d)  # Output: 1 2 'apple' 'banana'
```

SETS

Sets is a collection of unordered and unique elements. It is very useful when we want to store unique elements and ensure data uniqueness. Set offers efficient mathematical operations like union, intersection, and symmetric difference.



```
# Creating a set
my_set = {1, 2, 3, 4, 5}
print(my_set)

# Adding elements
my_set.add(6)
print(my_set)  # Output: {1, 2, 3, 4, 5, 6}

# Removing elements
my_set.remove(3)
print(my_set)  # Output: {1, 2, 4, 5, 6}
```

CONCLUSION

With lists, dictionaries, tuples, and sets, Python provides a robust suite of data structures to suit various needs. Mastering them enhances your programming capabilities and efficiency.