Python Collection Types

📚 1. List

- **Ordered, Mutable, Allows Duplicate Values**

✅ Basic Methods with Examples:

```
# Create a list
my_list = [10, 20, 30, 40, 50]

# 1. append() - Add an element at the end
my_list.append(60)
print(my_list)  # [10, 20, 30, 40, 50, 60]

# 2. insert() - Add an element at a specific index
my_list.insert(2, 25)
print(my_list)  # [10, 20, 25, 30, 40, 50, 60]

# 3. extend() - Add multiple elements at the end
my_list.extend([70, 80])
print(my_list)  # [10, 20, 25, 30, 40, 50, 60, 70, 80]

# 4. remove() - Remove the first occurrence of a value
my_list.remove(25)
print(my_list)  # [10, 20, 30, 40, 50, 60, 70, 80]

# 5. pop() - Remove and return an element at a specific index
popped_element = my_list.pop(3)
print(popped_element)  # 40
print(my_list)  # [10, 20, 30, 50, 60, 70, 80]

# 6. index() - Get the index of a specific element
index = my_list.index(60)
print(index)  # 4
```

```
# 7. count() - Count occurrences of a value
count = my_list.count(20)
print(count)  # 1

# 8. sort() - Sort the list in ascending order
my_list.sort()
print(my_list)  # [10, 20, 30, 50, 60, 70, 80]

# 9. reverse() - Reverse the list
my_list.reverse()
print(my_list)  # [80, 70, 60, 50, 30, 20, 10]

# 10. clear() - Remove all elements from the list
my_list.clear()
print(my_list)  # []
```

📚 2. Tuple

- **Ordered, Immutable, Allows Duplicate Values**

✅ Basic Methods with Examples:

```
# Create a tuple
my_tuple = (10, 20, 30, 40, 50)

# 1. index() - Get the index of an element
index = my_tuple.index(30)
print(index)  # 2

# 2. count() - Count occurrences of a value
count = my_tuple.count(20)
print(count)  # 1

# Tuple unpacking
a, b, c, d, e = my_tuple
print(a, b, c, d, e)  # 10 20 30 40 50
```

```
# Concatenation of tuples
new_tuple = my_tuple + (60, 70)
print(new_tuple)  # (10, 20, 30, 40, 50, 60, 70)

# Tuple slicing
sliced_tuple = my_tuple[1:4]
print(sliced_tuple)  # (20, 30, 40)
```

📚 3. Set

- **Unordered, Mutable, No Duplicate Values**

✅ Basic Methods with Examples:

```
# Create a set
my_set = {10, 20, 30, 40, 50}

# 1. add() - Add a single element
my_set.add(60)
print(my_set)  # {10, 20, 30, 40, 50, 60}

# 2. update() - Add multiple elements
my_set.update([70, 80])
print(my_set)  # {10, 20, 30, 40, 50, 60, 70, 80}

# 3. remove() - Remove an element (raises error if not found)
my_set.remove(30)
print(my_set)  # {10, 20, 40, 50, 60, 70, 80}

# 4. discard() - Remove an element (no error if not found)
my_set.discard(90)  # No error
print(my_set)

# 5. pop() - Remove and return a random element
popped_element = my_set.pop()
```

```
print(popped_element)  # Random element
print(my_set)

# 6. clear() - Remove all elements
my_set.clear()
print(my_set)  # set()

# Set operations
A = {1, 2, 3, 4}
B = {3, 4, 5, 6}

# 7. union()
print(A.union(B))  # {1, 2, 3, 4, 5, 6}

# 8. intersection()
print(A.intersection(B))  # {3, 4}

# 9. difference()
print(A.difference(B))  # {1, 2}

# 10. symmetric_difference()
print(A.symmetric_difference(B))  # {1, 2, 5, 6}
```

📚 4. Dictionary

- **Key-Value Pairs, Unordered, Mutable, No Duplicate Keys**

✅ Basic Methods with Examples:

```
# Create a dictionary
my_dict = {'name': 'Sanket', 'age': 30, 'city': 'Rajkot'}

# 1. keys() - Get all keys
print(my_dict.keys())  # dict_keys(['name', 'age', 'city'])

# 2. values() - Get all values
```

```
print(my_dict.values())  # dict_values(['Sanket', 30, 'Rajkot'])

# 3. items() - Get all key-value pairs
print(my_dict.items())  # dict_items([('name', 'Sanket'), ('age', 30),
('city', 'Rajkot')])

# 4. get() - Get the value for a key
print(my_dict.get('name'))  # Sanket

# 5. update() - Update dictionary with another dictionary
my_dict.update({'age': 31, 'country': 'India'})
print(my_dict)  # {'name': 'Sanket', 'age': 31, 'city': 'Rajkot',
'country': 'India'}

# 6. pop() - Remove and return the value of a key
age = my_dict.pop('age')
print(age)  # 31
print(my_dict)

# 7. popitem() - Remove and return the last inserted key-value pair
item = my_dict.popitem()
print(item)  # ('country', 'India')

# 8. setdefault() - Get the value of a key, or set a default if not
found
value = my_dict.setdefault('city', 'Ahmedabad')
print(value)  # Rajkot
print(my_dict)

# 9. clear() - Remove all key-value pairs
my_dict.clear()
print(my_dict)  # {}
```
