DATA SCIENCE COURSE TUTORIAL # 27

3.17.3 Sets

What is a Set?

A set is a collection of unordered, unindexed, and unique items. Sets do not allow duplicate values, and their order is not guaranteed.

Example:

```
fruits = {"apple", "banana", "mango", "apple"}
print(fruits)   # {'apple', 'banana', 'mango'} (duplicate 'apple' removed)
```

Accessing Set Items

You cannot access items in a set using an index because sets are unordered. Instead, you can loop through the set.

Example:

```
for fruit in fruits:
    print(fruit)
```

Adding Items to a Set

You can add new items to a set using the `add()` or `update()` methods.

Example:

```
fruits.add("grape")
fruits.update(["kiwi", "orange"])
print(fruits)
```

Removing Items from a Set

Sets provide different methods to remove items.

Example:

```
fruits.remove("banana")    # Removes 'banana', error if not found
fruits.discard("apple")    # Removes 'apple', no error if not found
fruits.pop()               # Removes a random item
```

You can also clear all items:

```
fruits.clear()
```

Set Operations

Sets are very useful for mathematical operations like union, intersection, and difference.

Example:

```
A = {1, 2, 3, 4}
B = {3, 4, 5, 6}

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

Set Methods and Functions

Example:

```
numbers = {10, 20, 30, 40}

print(len(numbers))   # 4 (length of set)
numbers.add(50)
print(numbers)        # {40, 10, 50, 20, 30}
```

Other useful methods:

- `copy()` → Returns a copy of the set.
- `issubset()` → Checks if one set is a subset of another.
- `issuperset()` → Checks if a set contains another set.
- `isdisjoint()` → Returns True if two sets have no items in common.

Example:

```
X = {1, 2}
Y = {1, 2, 3, 4}

print(X.issubset(Y))     # True
print(Y.issuperset(X))   # True
print(X.isdisjoint({5})) # True
```