Python Online Free Ramzan Course 2025
Taught by: Shaida Muhammad

Sets

Day 10 - Python Basics

Shaida Muhammad

Friday, March 14, 2025

ShaidaSherpao@gmail.com

Agenda

1. What are sets?
2. Creating and accessing sets
3. Set operations (union, intersection, difference)
4. Set methods (add(), remove(), discard(), etc.)
5. Hands-on practice

What are Sets?

- **Definition:** A set is a collection of unique items with no duplicates.
- **Features:**
  - Unordered: Items are not stored in a specific order.
  - Mutable: Items can be added or removed.
  - Can contain different data types.
- **Example:**

```
fruits = {"apple", "banana", "cherry"}
```

Creating and Accessing Sets

- **Creating a Set:**

```
my_set = {1, 2, 3, "apple", True}
```

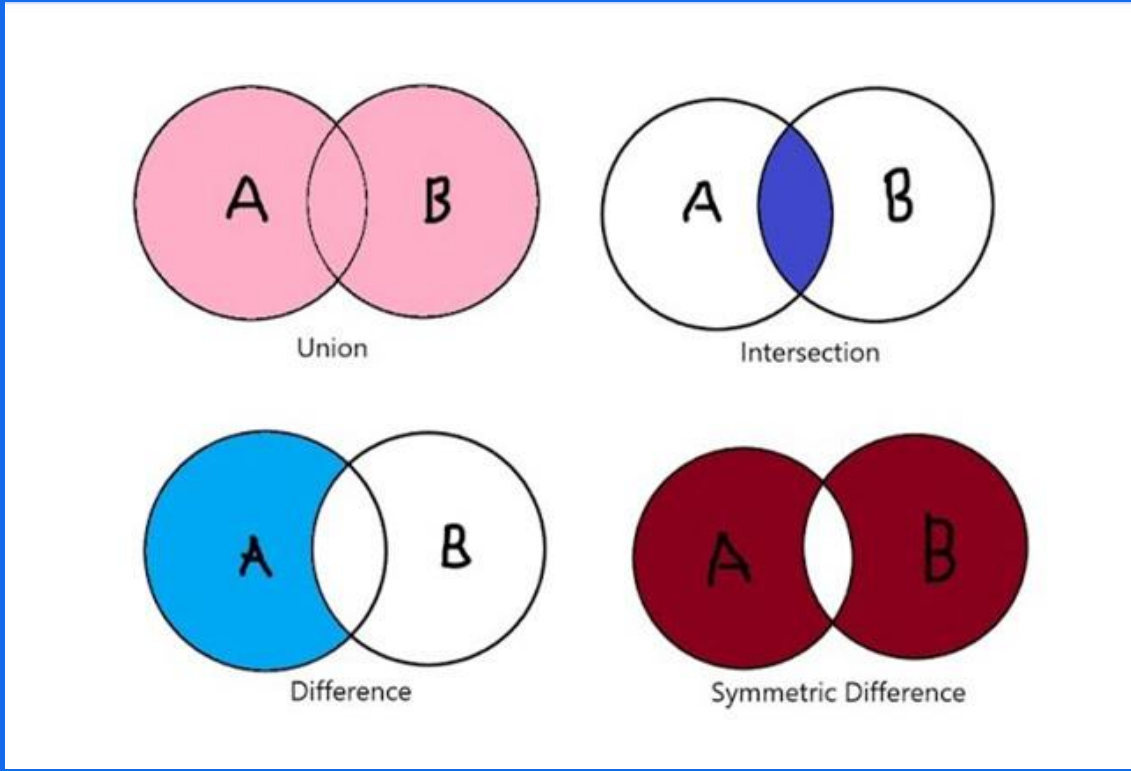
OR

```
my_set = set([1, 2, 3, "apple", True])
```

- **Accessing Items:**
  - Sets are unordered, so indexing is not supported.
  - Use loops to access items.

```
for item in my_set:
    print(item)
```

Set Operations



Set Operations ...

- **Union (`|` or `union()`):** Combines two sets.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
print(set1 | set2)  # Output: {1, 2, 3, 4, 5}
print(set1.union(set2))  # Alternative method
```

- **Intersection (`&` or `intersection()`):** Returns common items.

```
print(set1 & set2)  # Output: {3}
print(set1.intersection(set2))  # Alternative method
```

- **Difference (`-` or `difference()`):** Returns items in the first set but not in the second.

```
print(set1 - set2)  # Output: {1, 2}
print(set1.difference(set2))  # Alternative method
```

- **Symmetric Difference (`^` or `symmetric_difference()`):** Returns items that are in either set, but not both.

```
print(set1 ^ set2)  # Output: {1, 2, 4, 5}
print(set1.symmetric_difference(set2))  # Alternative method
```

Set Methods

- **Adding Items:**
  - `add()`: Adds a single item.
    ```
    fruits.add("orange")
    ```
  - `update()`: Adds multiple items.
    ```
    fruits.update(["mango", "grape"])
    ```
- **Removing Items:**
  - `remove()`: Removes an item (raises an error if the item doesn't exist).
    ```
    fruits.remove("banana")
    ```
  - `discard()`: Removes an item (no error if the item doesn't exist).
    ```
    fruits.discard("banana")
    ```
  - `pop()`: Removes and returns a random item.
    ```
    fruit = fruits.pop()
    ```
  - `clear()`: Removes all items.
    ```
    fruits.clear()
    ```
- **Copying a Set:**
  ```
  new_set = fruits.copy()
  print(new_set)
  ```

Set Comparisons

- **Checking if a set is a subset of another (`issubset()`):**

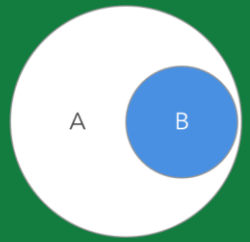
```
setA = {1, 2}
setB = {1, 2, 3, 4}
print(setA.issubset(setB))
# Output: True
```

- **Checking if a set is a superset of another (`issuperset()`):**

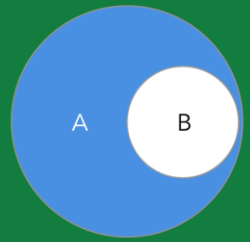
```
print(setB.issuperset(setA))
# Output: True
```

- **Checking if two sets are disjoint (`isdisjoint()`):**

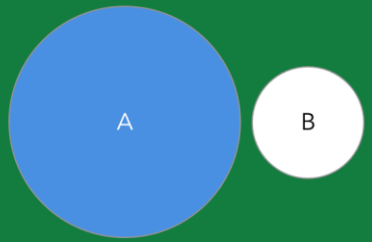
```
setX = {1, 2, 3}
setY = {4, 5, 6}
print(setX.isdisjoint(setY))
# Output: True
```



B is subset of A



A is superset of B



A is disjoint with B

Hands-On Practice

- **Task 1:** Create a set of your favorite fruits and print it.

```
fruits = {"apple", "banana", "cherry"}
print(fruits)
```

- **Task 2:** Add a new fruit to the set and print the updated set.

```
fruits.add("orange")
print(fruits)
```

- **Task 3:** Remove a fruit from the set and print the updated set.

```
fruits.discard("banana")
print(fruits)
```


- **Task 4:** Perform set operations (`union`, `intersection`, `difference`, `symmetric_difference`).

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
print(set1 | set2)  # Union
print(set1 & set2)  # Intersection
print(set1 - set2)  # Difference
print(set1 ^ set2)  # Symmetric Difference
```

- **Task 5:** Use `issubset()`, `issuperset()`, and `isdisjoint()`.

```
setA = {1, 2}
setB = {1, 2, 3, 4}
print(setA.issubset(setB))  # True
print(setB.issuperset(setA))  # True
print(setA.isdisjoint(setB))  # False
```

Recap

- Sets are unordered collections of unique items.
- Use add(), remove(), discard(), pop(), and clear() to modify sets.
- Use update() to add multiple items to a set.
- issubset(), issuperset(), and isdisjoint() help compare sets.
- Use copy() to make a duplicate set.

Q&A

- Do you have any questions?
- Share your thoughts.

Closing

Next class: Functions