## 1. What is a set in Python? How does it differ from a list and a tuple? What are its key properties?

**Set: An unordered collection of unique items in Python.**

**Key Properties:**

**No duplicate elements**

**Unordered (cannot use indexing)**

**Mutable (can add/remove elements)**

**Stores only immutable objects**

**List → A mutable (changeable) ordered collection of items.**

```
Uses more memory because of mutability.
```

**Tuple → An immutable (unchangeable) ordered collection of items.**

```
Uses less memory.
```

# EXAMPLE:-

In [3]:
```
set={1,2,3,3}
print(set)
```

```
{1, 2, 3}
```

In [5]:
```
a = {1, 2, 3}
b = {3, 4, 5}
print(a & b)
```

```
{3}
```

## 2. What is a frozenset? How does it differ from a regular set? When might you use a frozenset?

frozenset: An immutable version of a set. A frozenset is an immutable version of a set — once created, it cannot be changed.

**Differences:**

Set → Mutable, can be changed Frozenset → Immutable, cannot be changed

**Use when:**

You need a set as a dictionary key You want to store it in another set

In [6]:
```
fs = frozenset([1, 2, 3])
print(fs)
```

```
frozenset({1, 2, 3})
```

### 3. Can you have mutable data types as elements of a set? Why or why not? What about frozensets?

In [ ]:
```
No, you can't have mutable types (like lists or dictionaries) inside a set becau
frozenset: Same rule – elements must be immutable.
```

### 4. Explain the purpose and usage of the split() method for strings. Provide an example.

In [ ]:
```
Purpose: Splits a string into a list of words (or parts) based on a separator.
Default separator is space.
```

In [7]:
```
text = "HII GOOD MORNING"
words = text.split()
print(words)
```

```
['HII', 'GOOD', 'MORNING']
```

### 5. Explain the purpose and usage of the join() method for strings. Provide an example.

In [ ]:
```
Purpose: Joins elements of a list (or iterable) into a single string with a spec
```

In [9]:
```
words = ["hii", "shree" ]
sentence = " ".join(words)
print(sentence)
```

```
hii shree
```

### 6. What is a dictionary in Python? How does it store data? What are the key properties of a dictionary?

In [ ]:
```
Dictionary: A collection of key-value pairs.
Stores data in a way that each key maps to a value.
Key properties:
Keys are unique
Keys must be immutable
Values can be of any type
Dictionaries are mutable
```

### 7. What are the requirements for keys in a Python dictionary? Why are these requirements in place? Can you use a list as a key?

In [ ]:
```
Requirements:
Must be immutable (hashable)
Must be unique

Why: Dictionary uses hashing for fast lookup, which needs immutable keys.
List as a key?  No, because lists are mutable.
```

### 8. Create two sets, set1 with elements [1, 2, 3, 4, 5] and set2 with elements [4, 5, 6, 7, 8]. Convert these lists to sets.

In [13]:
```
set1 = set ([1, 2, 3, 4, 5])
set2 = set ([4, 5, 6, 7, 8])
```

```
print(set1)
print(set2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[13], line 1
----> 1 set1 = set ([1, 2, 3, 4, 5])
      2 set2 = set ([4, 5, 6, 7, 8])
      3 print(set1)

TypeError: 'set' object is not callable
```

### 9. Find and print the common elements between set1 and set2 created in the previous question.

In [14]:
```
sentence = input("Enter a sentence: ")
words = sentence.split()
unique_words = set(words)
print("Unique words:", unique_words)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[14], line 3
      1 sentence = input("Enter a sentence: ")
      2 words = sentence.split()
----> 3 unique_words = set(words)
      4 print("Unique words:", unique_words)

TypeError: 'set' object is not callable
```

### 11. You have a list of words: ["Hello", "World", "Python"]. Use the join() method to create a single string with these words separated by a hyphen "-".

In [16]:
```
words = ["Hello", "World", "Python"]
result = "-".join(words)
print(result)
```

```
Hello-World-Python
```

### 12. Create a dictionary that stores the capital cities of three countries.

In [23]:
```
capitals = {
    "USA": "Washington D.C.",
    "France": "Paris",
    "Japan": "Tokyo"
}
print(capitals )
```

```
{'USA': 'Washington D.C.', 'France': 'Paris', 'Japan': 'Tokyo'}
```

### 13. Given the dictionary from the previous question, ask the user to enter a country and print its capital city. Handle the case where the country is not in the dictionary.

In [21]:
```
country = input("Enter country name: ")
if country in capitals:
    print("Capital:", capitals[country])
```

```
else:
    print("Sorry, country not found.")
```

```
Sorry, country not found.
```

In [ ]: