In [10]:

```
# Experiment-7: Create a list of numbers and perform various operatio
# removing elements, and accessing elements by index.

# Create a list of numbers
numbers = [1, 3, 5, 7, 9]

# Print the original list
print("\nOriginal list:", numbers)

# Add elements
numbers.append(11)  # Add to the end
numbers.insert(2, 4)  # Insert at index 2

# Print the list after adding
print("\nList after adding:", numbers)

# Remove elements
numbers.remove(3)  # Remove the first occurrence of 3
numbers.pop(1)  # Remove the element at index 1

# Print the list after removing
print("\nList after removing:", numbers)

# Access elements by index
first_element = numbers[0]
last_element = numbers[-1]

# Print accessed elements
print("\nFirst element:", first_element)
print("Last element:", last_element)

# Accessing out of bounds raises an IndexError
try:
    out_of_bounds = numbers[10]
except IndexError:
    print("\nOut of bounds access: IndexError")

# Iterate through the list
print("\nIterating through list:")
for number in numbers:
    print("Number:", number)
```

```
Original list: [1, 3, 5, 7, 9]

List after adding: [1, 3, 4, 5, 7, 9, 11]

List after removing: [1, 5, 7, 9, 11]

First element: 1
Last element: 11

Out of bounds access: IndexError

Iterating through list:
Number: 1
Number: 5
Number: 7
Number: 9
Number: 11
```

In [11]:

```
# Experiment-8: Create a tuple to represent an immutable collection a
# try accessing elements and performing basic operations.

# Create a tuple
person = ("John", 30, "Software Developer")

# Access elements by index
name = person[0]
age = person[1]
profession = person[2]

print(f"Name: {name}")
print(f"Age: {age}")
print(f"Profession: {profession}")

# Accessing out of bounds raises IndexError
print("\nAccessing Out of Bounds:")
try:
    job_title = person[3]
except IndexError:
    print("Out of bounds access: IndexError")

# Iterate through the tuple
print("\nIterating through tuple:", end = " ")
for element in person:
    print("Element:", element)

# Check membership (returns True if found)
is_developer = "Developer" in person
print(f"\nIs 'Developer' in the tuple? {is_developer}")
```

```
Name: John
Age: 30
Profession: Software Developer

Accessing Out of Bounds:
Out of bounds access: IndexError

Iterating through tuple: Element: John
Element: 30
Element: Software Developer

Is 'Developer' in the tuple? False
```

In [16]:

```
# Experiment-9: Create a set to store unique elements and perform set
# operations such as union, intersection, and difference.
# Create sets of numbers
set1 = {1, 2, 3, 4, 5}
set2 = {3, 4, 5, 6, 7}

# Print the original sets
print("\nSet 1:", set1)
print("Set 2:", set2)

# Union (combines all unique elements)
union_set = set1 | set2  # or use union() method
print("\nUnion:", union_set)

# Intersection (elements present in both sets)
intersection_set = set1 & set2  # or use intersection() method
print("\nIntersection:", intersection_set)

# Difference (elements in set1 but not in set2)
difference_set = set1 - set2  # or use difference() method
print("\nDifference (Set 1 - Set 2):", difference_set)

# Difference (elements in set2 but not in set1)
difference_set = set2 - set1  # or use difference() method
print("Difference (Set 2 - Set 1):", difference_set)

# Check membership (True if element exists)
is_in_set1 = 2 in set1
print("\nIs 2 in Set 1? ", is_in_set1)

# Add elements to a set (duplicates are automatically removed)
set1.add(8)
print("Set 1 after adding 8:", set1)

# Remove elements from a set
set2.remove(5)  # Raises KeyError if element not found
print("Set 2 after removing 5:", set2)

# Discard elements from a set (silently removes, no error)
set1.discard(10)  # No error even if 10 doesn't exist
print("Set 1 after discarding 10:", set1)
```

```
# Sets are unordered - iterating doesn't guarantee order
print("\nIterating through set 1:")
for element in set1:
    print("Element:", element)
```

```
Set 1: {1, 2, 3, 4, 5}
Set 2: {3, 4, 5, 6, 7}

Union: {1, 2, 3, 4, 5, 6, 7}

Intersection: {3, 4, 5}

Difference (Set 1 - Set 2): {1, 2}
Difference (Set 2 - Set 1): {6, 7}

Is 2 in Set 1?  True
Set 1 after adding 8: {1, 2, 3, 4, 5, 8}
Set 2 after removing 5: {3, 4, 6, 7}
Set 1 after discarding 10: {1, 2, 3, 4, 5, 8}

Iterating through set 1:
Element: 1
Element: 2
Element: 3
Element: 4
Element: 5
Element: 8
```

In [18]:

```
# Experiment-10: Create a dictionary to represent key-value pairs and
# perform operations like adding, updating, and accessing values.

# Create a dictionary with student information
student = {"name": "Aritra", "age": 21, "major": "Computer Science"}

# Print the original dictionary
print("Original dictionary:", student)

# Access values by key
name = student["name"]
age = student["age"]
major = student["major"]

print(f"Name: {name}")
print(f"Age: {age}")
print(f"Major: {major}")

# Add a new key-value pair
student["GPA"] = 9.54

# Update an existing value
student["age"] = 24

# Print the updated dictionary
print("\nUpdated dictionary:", student)

# Check if a key exists
has_phone_number = "phone_number" in student

# Access using a non-existent key raises KeyError
try:
    phone_number = student["phone_number"]
except KeyError:
    print("\nKeyError: 'phone_number' not found")

# Delete a key-value pair
del student["age"]

# Print the dictionary after deletion
print("\nDictionary after deleting age:", student)
```

```
# Get all keys and values using methods
keys = list(student.keys())
values = list(student.values())

print("\nKeys:", keys)
print("Values:", values)

# Iterate through key-value pairs
print("\nIterating through dictionary:")
for key, value in student.items():
    print(f"{key}: {value}")
```

```
Original dictionary: {'name': 'Aritra', 'age': 21, 'major': 'Computer Science'}
Name: Aritra
Age: 21
Major: Computer Science

Updated dictionary: {'name': 'Aritra', 'age': 24, 'major': 'Computer Science', 'GPA': 9.5

KeyError: 'phone_number' not found

Dictionary after deleting age: {'name': 'Aritra', 'major': 'Computer Science', 'GPA': 9.5

Keys: ['name', 'major', 'GPA']
Values: ['Aritra', 'Computer Science', 9.54]

Iterating through dictionary:
name: Aritra
major: Computer Science
GPA: 9.54
```