Python Data Structures Tutorial

Focus: Lists, Sets, Dictionaries, Tuples, and Strings

Designed for BSc AI/ML Students

Python Lists

- What is a List?
- - Ordered, mutable collection of items.
- - Can contain mixed data types.
- Syntax: my_list = [1, 2, 3, 'apple', True]
- Examples:
- 1. Create a list: fruits = ['apple', 'banana', 'cherry']
- 2. Access elements: print(fruits[0]) # Output: apple
- 3. Modify elements: fruits[1] = 'blueberry'
- 4. Add elements: fruits.append('orange')
- 5. Remove elements: fruits.remove('cherry')

Common List Operations

- 1. Slicing: print(fruits[1:3]) # Output: ['banana', 'cherry']
- 2. Length: print(len(fruits)) # Output: 3
- 3. Looping:
-   for fruit in fruits:
-       print(fruit)
- 4. Sorting: fruits.sort()
- 5. Reversing: fruits.reverse()

Python Sets

- What is a Set?
- - Unordered, mutable collection of unique items.
- - Cannot contain duplicates.
- Syntax: my_set = {1, 2, 3, 'apple'}
- Examples:
- 1. Create a set: colors = {'red', 'green', 'blue'}
- 2. Add elements: colors.add('yellow')
- 3. Remove elements: colors.remove('green')
- 4. Check membership: 'red' in colors # Output: True

Common Set Operations

- 1. Union: set1 = {1, 2, 3}; set2 = {3, 4, 5}; print(set1 | set2) # Output: {1, 2, 3, 4, 5}
- 2. Intersection: print(set1 & set2) # Output: {3}
- 3. Difference: print(set1 - set2) # Output: {1, 2}
- 4. Symmetric Difference: print(set1 ^ set2) # Output: {1, 2, 4, 5}
- 5. Looping:
-   for color in colors:
-       print(color)

Python Dictionaries

- What is a Dictionary?
- - Unordered, mutable collection of key-value pairs.
- - Keys must be unique and immutable.
- Syntax: my_dict = {'name': 'Alice', 'age': 25}
- Examples:
- 1. Create a dictionary: person = {'name': 'Alice', 'age': 25}
- 2. Access values: print(person['name']) # Output: Alice
- 3. Modify values: person['age'] = 26
- 4. Add new key-value pairs: person['city'] = 'New York'
- 5. Remove key-value pairs: del person['age']

Common Dictionary Operations

- 1. Looping through keys:
-   for key in person:
-       print(key)
- 2. Looping through key-value pairs:
-   for key, value in person.items():
-       print(key, value)
- 3. Check if key exists: 'name' in person # Output: True
- 4. Get all keys: print(person.keys())
- 5. Get all values: print(person.values())

Python Tuples

- What is a Tuple?
- - Ordered, immutable collection of items.
- - Often used for fixed data.
- Syntax: my_tuple = (1, 2, 3, 'apple')
- Examples:
- 1. Create a tuple: coordinates = (10, 20)
- 2. Access elements: print(coordinates[0]) # Output: 10
- 3. Tuples are immutable: coordinates[0] = 15 # This will raise an error
- 4. Looping:
-   for item in coordinates:
-       print(item)

Python Strings

- What is a String?
- - Immutable sequence of characters.
- - Enclosed in single or double quotes.
- Syntax: my_string = 'Hello, World!'
- Examples:
- 1\. Create a string: greeting = 'Hello, World!'
- 2\. Access characters: print(greeting[0]) # Output: H
- 3\. Strings are immutable: greeting[0] = 'h' # This will raise an error
- 4\. Common operations:
-   - Concatenation: 'Hello' + ' ' + 'World'
-   - Slicing: greeting[0:5] # Output: 'Hello'
-   - Length: len(greeting) # Output: 13

Practice Exercises

- 1. Create a list of your favorite movies and print the second movie.
- 2. Add a new movie to the list and remove the last one.
- 3. Create a set of unique numbers and perform a union with another set.
- 4. Create a dictionary of student names and their grades, then update a grade.
- 5. Create a tuple of coordinates and try to modify it (observe the error).
- 6. Create a string and perform slicing and concatenation.

Conclusion

- Key Takeaways:
- - Lists: Ordered, mutable collections.
- - Sets: Unordered, unique collections.
- - Dictionaries: Key-value pairs, mutable.
- - Tuples: Ordered, immutable collections.
- - Strings: Immutable sequences of characters.
- Next Steps: Practice using these data structures in your AI/ML projects!