Python Programming

Unit 02 – Lecture 03: Sets and Dictionaries

Tofik Ali

School of Computer Science, UPES Dehradun

February 14, 2026

Repository: `https://github.com/tali7c/Python-Programming`

Quick Links

Core Concepts Demo Interactive Summary

Agenda

Learning Outcomes

- Create and use sets for uniqueness and membership

Learning Outcomes

- Create and use sets for uniqueness and membership
- Perform common set operations

Learning Outcomes

- Create and use sets for uniqueness and membership
- Perform common set operations
- Build dictionaries for key-value modeling

Learning Outcomes

- Create and use sets for uniqueness and membership
- Perform common set operations
- Build dictionaries for key-value modeling
- Sort and summarize collections with dicts

Sets: The Idea

- Unordered collection of unique elements

Sets: The Idea

- Unordered collection of unique elements
- Fast membership testing

Sets: The Idea

- Unordered collection of unique elements
- Fast membership testing
- Useful for de-duplication

Set Creation and Membership

- Create with braces or `set()`

```
colors = {"red", "blue", "red"}
print(colors)              # duplicates removed
print("red" in colors)     # True
```

Set Creation and Membership

- Create with braces or `set()`
- Use `in` to test membership

```
colors = {"red", "blue", "red"}
print(colors)              # duplicates removed
print("red" in colors)     # True
```

Set Operations

- Union: `A | B`

Set Operations

- Union: `A | B`
- Intersection: `A & B`

Set Operations

- Union: `A | B`
- Intersection: `A & B`
- Difference: `A - B`

Set Operations

- Union: `A | B`
- Intersection: `A & B`
- Difference: `A - B`
- Symmetric difference: `A ^ B`

Pitfall: Sets Are Unordered

- Sets do not support indexing (no `s[0]`)

```
s = {3, 1, 2}
print(sorted(s))      # [1, 2, 3]
```

Pitfall: Sets Are Unordered

- Sets do not support indexing (no `s[0]`)
- Use sets for membership/uniqueness, not ordering

```
s = {3, 1, 2}
print(sorted(s))      # [1, 2, 3]
```

Pitfall: Sets Are Unordered

- Sets do not support indexing (no `s[0]`)
- Use sets for membership/uniqueness, not ordering
- Convert to a sorted list when you need order

```
s = {3, 1, 2}
print(sorted(s))      # [1, 2, 3]
```

Dictionaries: Key-Value Model

- Map keys to values

```
student = {"name": "Asha", "marks": 88}
print(student["name"])
```

Dictionaries: Key-Value Model

- Map keys to values
- Keys must be immutable

```
student = {"name": "Asha", "marks": 88}
print(student["name"])
```

Dictionaries: Key-Value Model

- Map keys to values
- Keys must be immutable
- Fast lookup by key

```
student = {"name": "Asha", "marks": 88}
print(student["name"])
```

Dictionary Operations

- Access: `d[key]` or `d.get(key)`

Dictionary Operations

- Access: `d[key]` or `d.get(key)`
- Update: `d[key] = value`

Dictionary Operations

- Access: `d[key]` or `d.get(key)`
- Update: `d[key] = value`
- Iterate: `for k, v in d.items()`

Worked Example: Frequency Count Pattern

```
text = "to be or not to be"
freq = {}
for w in text.split():
    freq[w] = freq.get(w, 0) + 1
print(freq)
```

- `get(w, 0)` avoids `KeyError` for new words

Sorting Collections

- Sort dict items by key or value

```
counts = {"a": 3, "b": 1, "c": 2}
by_value = sorted(counts.items(), key=lambda x: x[1])
```

Sorting Collections

- Sort dict items by key or value
- `sorted(d.items(), key=...)`

```
counts = {"a": 3, "b": 1, "c": 2}
by_value = sorted(counts.items(), key=lambda x: x[1])
```

Applications

- Frequency counts with dictionaries

Applications

- Frequency counts with dictionaries
- Unique vocabulary with sets

Applications

- Frequency counts with dictionaries
- Unique vocabulary with sets
- Fast membership checks

Demo: Word Frequency and Unique Words

- Split a sentence into words

Script: `demo/word_frequency.py`

Demo: Word Frequency and Unique Words

- Split a sentence into words
- Build a frequency dictionary

Script: `demo/word_frequency.py`

Demo: Word Frequency and Unique Words

- Split a sentence into words
- Build a frequency dictionary
- Build a set of unique words

Script: `demo/word_frequency.py`

Demo: Word Frequency and Unique Words

- Split a sentence into words
- Build a frequency dictionary
- Build a set of unique words
- Extension: print top 3 most frequent words

Script: `demo/word_frequency.py`

Checkpoint 1

What is the result of `set([1,1,2,2])`?

Checkpoint 2

What is the difference between `d[key]` and `d.get(key)`?

Think-Pair-Share

Choose set vs list for membership checks and justify.

Summary

- Sets store unique elements and support set algebra

Summary

- Sets store unique elements and support set algebra
- Dictionaries map keys to values for fast lookup/counting

Summary

- Sets store unique elements and support set algebra
- Dictionaries map keys to values for fast lookup/counting
- Sorting dict items enables ranking and reporting

Exit Question

Write one line to sort dictionary items by value in descending order.