Sets

October 7, 2024

1 Sets

A set is a collection of unique data. That is, elements of a set cannot be duplicate. For example,

Suppose we want to store information about student IDs. Since student IDs cannot be duplicate, we can use a set.

1.0.1 Creating a Set

In Python, we create sets by placing all the elements inside curly braces {}, separated by comma.

[ ]:

```
# create a set of integer type
student_id = {112, 114, 116, 118, 115}
print('Student ID:', student_id)
```

[ ]:

```
# create a set of string type
vowel_letters = {'a', 'e', 'i', 'o', 'u'}
print('Vowel Letters:', vowel_letters)
```

[6]:

```
# create a set of mixed data types
mixed_set = {'Hello', 101, -2, 'Bye'}
print('Set of mixed data types:', mixed_set)

for i in mixed_set:
    print(type(i))
```

```
Set of mixed data types: {'Bye', 'Hello', 101, -2}
<class 'str'>
<class 'str'>
<class 'int'>
<class 'int'>
```

[7]:

```
print(mixed_set)
```

```
{'Bye', 'Hello', 101, -2}
```

Create an Empty Set in Python Creating an empty set is a bit tricky. Empty curly braces {} will make an empty dictionary in Python. To make a set without any elements, we use the set() function without any argument. For example

[1]:

```python
# create an empty set
empty_set = set()
print('Data type of empty_set:', type(empty_set))
```

```
Data type of empty_set: <class 'set'>
```

1.0.2 Duplicate Items in a Set

Let's see what will happen if we try to include duplicate items in a set.

[8]:

```python
numbers = {2, 4, 6, 6, 2, 8}
print(numbers)
```

```
{8, 2, 4, 6}
```

1.0.3 Add and Update Set Items in Python

Sets are mutable. However, since they are unordered, indexing has no meaning.

We cannot access or change an element of a set using indexing or slicing. Set data type does not support it.

1.0.4 Add Items

[9]:

```python
numbers = {21, 34, 54, 12}

print('Initial Set:',numbers)

# using add() method
numbers.add(32)

print('Updated Set:', numbers)
```

```
Initial Set: {34, 12, 21, 54}
Updated Set: {32, 34, 12, 21, 54}
```

1.0.5 Update Sets

The update() method is used to update the set with items other collection types (lists, tuples, sets, etc). For example,

[10]:

```python
companies = {'Lacoste', 'Ralph Lauren'}
tech_companies = ['apple', 'google', 'apple']

companies.update(tech_companies)

print(companies)
```

```
{'Lacoste', 'Ralph Lauren', 'google', 'apple'}
```

1.0.6 Remove an Element from a Set

We use the discard() method to remove the specified element from a set. For example,

[ ]:

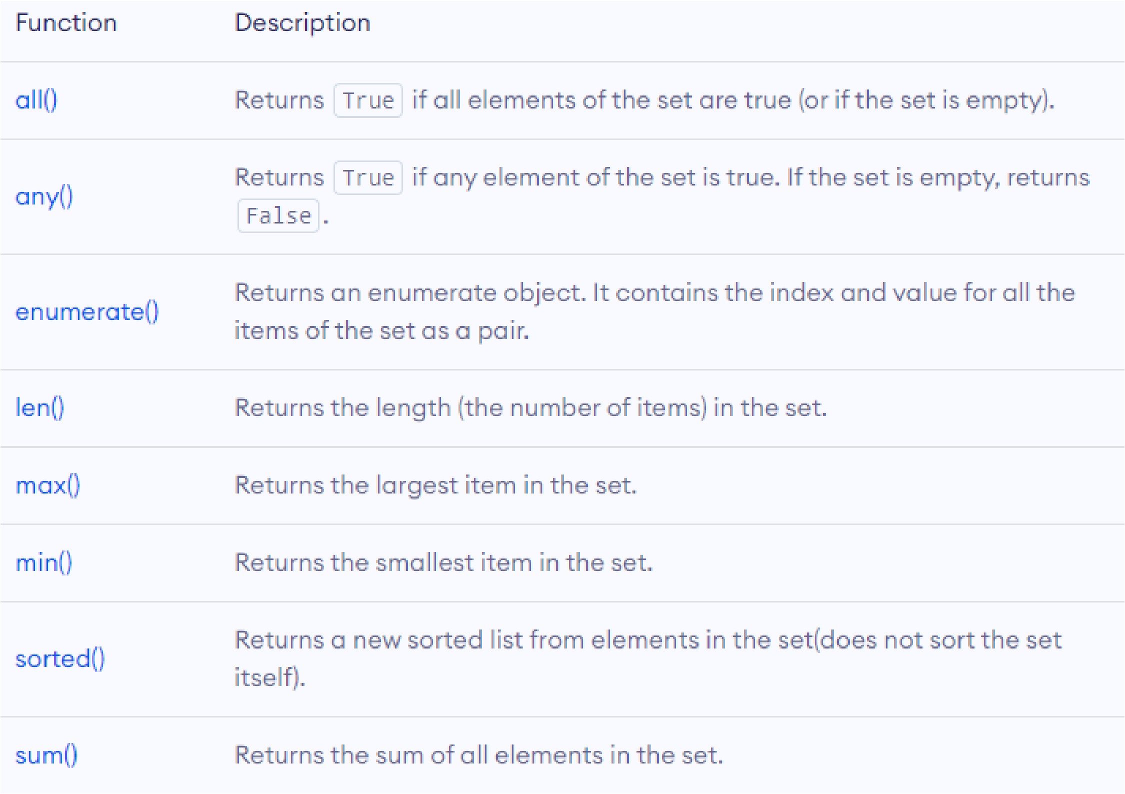
```
languages = {'Swift', 'Java', 'Python'}

print('Initial Set:',languages)

# remove 'Java' from a set
removedValue = languages.discard('Java')

print('Set after remove():', languages)
```

1.0.7 Built-in Functions with Set



Function	Description
all()	Returns `True` if all elements of the set are true (or if the set is empty).
any()	Returns `True` if any element of the set is true. If the set is empty, returns `False`.
enumerate()	Returns an enumerate object. It contains the index and value for all the items of the set as a pair.
len()	Returns the length (the number of items) in the set.
max()	Returns the largest item in the set.
min()	Returns the smallest item in the set.
sorted()	Returns a new sorted list from elements in the set(does not sort the set itself).
sum()	Returns the sum of all elements in the set.

1.0.8 Iterate Over a Set in Python

[ ]:

```
fruits = {"Apple", "Peach", "Mango"}

# for loop to access each fruits
for fruit in fruits:
    print(fruit)
```

Find Number of Set Elements

We can use the len() method to find the number of elements present in a Set. For example,

[ ]:

```
even_numbers = {2,4,6,8}
print('Set:',even_numbers)

# find number of elements
print('Total Elements:', len(even_numbers))
```

1.0.9 Summary

Sets can be made using curly braces or using the set() function given another sequence type:

- 1 food = {‘burgers’, ‘tacos’, ‘burritos’}
- 2 food = set([‘burgers’, ‘tacos’, ‘burritos’])
- 3 food = set((‘burgers’, ‘tacos’, ‘burritos’))

Note that empty curly braces create an empty dictionary. To create an empty set, use the set() function:

No element can appear twice.