Day 6

Python for Data Science

Sets & Dictionary

Sets in Python

- Set is one of 4 built-in data types in Python used to store collections of data including List, Tuple, and Dictionary
- Sets are unordered, but you can remove items and add new items.
- Set elements are unique. Duplicate elements are not allowed.
- A set itself may be modified, but the elements contained in the set must be of an immutable type.
- Sets are used to store multiple items in a single variable.
- You can denote a set with a pair of curly brackets {}.

```
#two ways to define a set
# 1.set(<iter>) #<iter> can be list or set
# 2.{'obj1','obj2','obj3'}
```

```
x = set([1,2,3])
y = {1,2,3}
print(type(x))
print(type(y))
```

```
<class 'set'>
<class 'set'>
```

```
# To take a set without elements, use set() function without any items
y = set()
print(type(y))
```

```
<class 'set'>
```

```
# Take a set
set1 = {'Hello Python!', 3.14, 1.618, 'Hello World!', 3.14, 1.618,
2022}
set1
```

```
{1.618, 2022, 3.14, 'Hello Python!', 'Hello World!'}
```

```
# The empty set of curly braces denotes the empty dictionary, not
empty set
x = {}
print(type(x))
```

```
<class 'dict'>
```

Converting list to set

```
# A list can convert to a set
# Take a list
nlis = ['Hello Python!', 3.14, 1.618, 'Hello World!', 3.14, 1.618,
True, False, 2022]
# Convert the list to a set
set2 = set(nlis)
set2
```

{1.618, 2022, 3.14, False, 'Hello Python!', 'Hello World!', True}

Set Operations

```
# Take a set
set3 = set(['Hello Python!', 3.14, 1.618, 'Hello World!', 3.14,
1.618, True, False, 2022])
set3
```

{1.618, 2022, 3.14, False, 'Hello Python!', 'Hello World!', True}

add() function

To add an element into a set, we use the function add(). If the same element is added to the set, nothing will happen because the set accepts no duplicates.

```
# Addition of an element to a set
set3 = set(['Hello Python!', 3.14, 1.618, 'Hello World!', 3.14,
1.618, True, False, 2022])
set3.add('Hi, Python!')
set3
```

{1.618,
2022,
3.14,
False,
'Hello Python!',
'Hello World!',
'Hi, Python!',
True}

```
# Addition of the same element
set3.add('Hi, Python!')
set3
# As you see that there is only one from the added element 'Hi,
Python!'
```

{1.618,
2022,
3.14,
False,
'Hello Python!',
'Hello World!',
'Hi, Python!',
True}

update() function

To add multiple elements into the set

```
x_set = {6,7,8,9}
print(x_set)
x_set.update({3,4,5})
print(x_set)
```

{8, 9, 6, 7}
{3, 4, 5, 6, 7, 8, 9}

remove() function

To remove an element from the set

```
set3.remove('Hello Python!')
set3
```

{1.618, 2022, 3.14, False, 'Hello World!', 'Hi, Python!', True}

discard() function

It leaves the set unchanged if the element to be deleted is not available in the set.

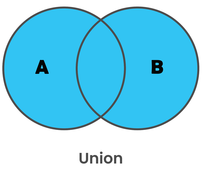
```
set3.discard(3.14)
set3
```

{1.618, 2022, False, 'Hello World!', 'Hi, Python!', True}

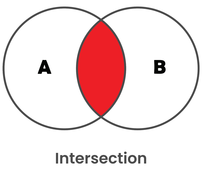
```
set3.discard(100)
set3
```

{1.618, 2022, False, 'Hello World!', 'Hi, Python!', True}

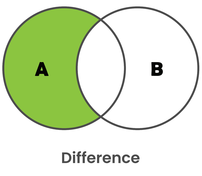
Logic operations in Sets



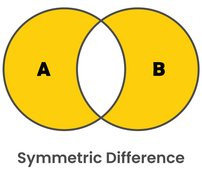
Union



Intersection



Difference



Symmetric Difference

```
# Take two sets
set4 = set(['Hello Python!', 3.14, 1.618, 'Hello World!'])
set5 = set([3.14, 1.618, True, False, 2022])
# Printing two sets
print(set4)
print(set5)
```

{1.618, 3.14, 'Hello World!', 'Hello Python!'}
{False, 1.618, True, 3.14, 2022}

To find the intersect of two sets using &

```
intersection = set4 & set5
intersection
```

{1.618, 3.14}

To find the intersect of two sets, use intersection() function

```
set4.intersection(set5) # The output is the same as that of above
```

{1.618, 3.14}

difference() function

To find the difference between two sets

```
print(set4.difference(set5))
print(set5.difference(set4))
# The same process can make using subtraction operator as follows:
print(set4-set5)
print(set5-set4)
```

{'Hello World!', 'Hello Python!'}
{False, True, 2022}
{'Hello World!', 'Hello Python!'}
{False, True, 2022}

Set comparison

```
print(set4>set5)
print(set5>set4)
print(set4==set5)
```

False
False
False

union() function

it corresponds to all the elements in both sets

```
set4.union(set5)
```

{1.618, 2022, 3.14, False, 'Hello Python!', 'Hello World!', True}

issuperset() and issubset() functions

To control if a set is a superset or a subset of another set

```
set(set4).issuperset(set5)
```

False

```
set(set4).issubset(set5)
```

False

```
print(set([3.14, 1.618]).issubset(set5))
print(set([3.14, 1.618]).issubset(set4))
print(set4.issuperset([3.14, 1.618]))
print(set5.issuperset([3.14, 1.618]))
```

True
True
True
True

min(), max() and sum() functions

```
A = [1,1,2,2,3,3,4,4,5,5] # Take a list
B = {1,1,2,2,3,3,4,4,5,5} # Take a set
print('The minimum number of A is', min(A))
print('The minimum number of B is', min(B))
print('The maximum number of A is', max(A))
print('The maximum number of B is', max(B))
print('The sum of A is', sum(A))
print('The sum of B is', sum(B))
# As you see that the sum of A and B is different. Because the
set takes no duplicate.
```

The minimum number of A is 1
The minimum number of B is 1
The maximum number of A is 5
The maximum number of B is 5
The sum of A is 30
The sum of B is 15

No mutable sequence in a set

A set can not have mutable elements such as list or dictionary in it. If any, it returns error as follows:

```
set6 = {'Python', 1,2,3, [1,2,3]}
set6
```

TypeError Traceback (most recent call last)
Input In [44], in <cell line: 1>()
----> 1 set6 = {'Python', 1,2,3, [1,2,3]}
2 set6

TypeError: unhashable type: 'list'

```
set6 = {'Python', 1,2,3, (1,2,3)}
set6
```

{(1, 2, 3), 1, 2, 3, 'Python'}

index() function

This function does not work in set since the set is unordered collection

```
set7 = {1,2,3,4}
set7[1]
```

TypeError Traceback (most recent call last)
Input In [46], in <cell line: 2>()
1 set7 = {1,2,3,4}
----> 2 set7[1]

TypeError: 'set' object is not subscriptable

clear() function

it removes all elements in the set and then do the set empty.

```
x = {0, 1,1,2,3,5,8,13, 21,34}
print(x)
x.clear()
print(x)
```

{0, 1, 2, 3, 34, 5, 8, 13, 21}
set()

pop() function

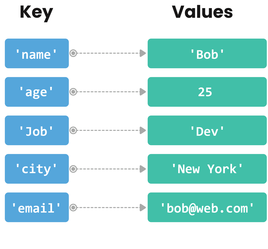
It removes and returns an arbitrary set element.

```
x = {0, 1,1,2,3,5,8,13,21,34}
print(x)
x.pop()
print(x)
```

{0, 1, 2, 3, 34, 5, 8, 13, 21}
{1, 2, 3, 34, 5, 8, 13, 21}

Dictionaries in Python Dictionaries are used to store data values in key:value pairs.

- Dictionary items are presented in key:value pairs, and can be referred to by using the key name.
- Dictionaries are changeable, meaning that we can change, add or remove items after the dictionary has been created.
- Dictionaries cannot have two items with the same key.
- A dictionary can nested and can contain another dictionary.



```
# Take a sample dictionary
sample_dict = {'key_1': 3.14, 'key_2': 1.618,
 'key_3': True, 'key_4': [3.14, 1.618],
 'key_5': (3.14, 1.618), 'key_6': 2022, (3.14, 1.618): 'pi and
golden ratio'}
sample_dict
```

{'key_1': 3.14,
'key_2': 1.618,
'key_3': True,
'key_4': [3.14, 1.618],
'key_5': (3.14, 1.618),
'key_6': 2022,
(3.14, 1.618): 'pi and golden ratio'}

Note: As you see that the whole dictionary is enclosed in curly braces, each key is separated from its value by a column ":", and commas are used to separate the items in the dictionary.

```
# Accessing to the value using the key
print(sample_dict['key_1'])
print(sample_dict['key_2'])
print(sample_dict['key_3'])
print(sample_dict['key_4'])
print(sample_dict['key_5'])
print(sample_dict['key_6'])
print(sample_dict[(3.14, 1.618)]) # Keys can be any immutable
object like tuple
```

3.14
1.618
True
[3.14, 1.618]
(3.14, 1.618)
2022
pi and golden ratio

Keys

```
# Take a sample dictionary
product = {'Aspergillus niger': 'inulinase', 'Saccharomyces cerevi-
siae': 'ethanol',
 'Scheffersomyces stipitis': 'ethanol', 'Aspergillus sojae_1': 'man-
nanase',
 'Streptococcus zooepidemicus': 'hyaluronic acid', 'Lactobacillus
casei': 'lacti c acid',
 'Aspergillus sojae_2': 'polygalacturonase'}
product
```

{'Aspergillus niger': 'inulinase',
'Saccharomyces cerevisiae': 'ethanol',
'Scheffersomyces stipitis': 'ethanol',
'Aspergillus sojae_1': 'mannanase',
'Streptococcus zooepidemicus': 'hyaluronic acid',
'Lactobacillus casei': 'lacti c acid',
'Aspergillus sojae_2': 'polygalacturonase'}

```
# Retrieving the value by keys
print(product['Aspergillus niger'])
print(product['Saccharomyces cerevisiae'])
print(product['Scheffersomyces stipitis'])
```

inulinase
ethanol
ethanol

keys() function to get the keys in the dictionary

```
# What are the keys in the dictionary?
product.keys()
```

dict_keys(['Aspergillus niger', 'Saccharomyces cerevisiae', 'Scheffersomyces stipitis',
'Aspergillus sojae_1', 'Streptococcus zooepidemicus', 'Lactobacillus casei',
'Aspergillus sojae_2'])

values() function to get the values in the dictionary

```
# What are the values in the dictionary?
product.values()
```

dict_values(['inulinase', 'ethanol', 'ethanol', 'mannanase', 'hyaluronic acid',
'lacti c acid', 'polygalacturonase'])

Addition of a new key:value pair in the dictionary

```
product['Yarrovia lipolytica'] = 'microbial oil'
product
```

{'Aspergillus niger': 'inulinase',
'Saccharomyces cerevisiae': 'ethanol',
'Scheffersomyces stipitis': 'ethanol',
'Aspergillus sojae_1': 'mannanase',
'Streptococcus zooepidemicus': 'hyaluronic acid',
'Lactobacillus casei': 'lacti c acid',
'Aspergillus sojae_2': 'polygalacturonase',
'Yarrovia lipolytica': 'microbial oil'}

Delete an item using del() function in the dictionary by key

```
del(product['Aspergillus niger'])
del(product['Aspergillus sojae_1'])
product
```

{'Saccharomyces cerevisiae': 'ethanol',
'Scheffersomyces stipitis': 'ethanol',
'Streptococcus zooepidemicus': 'hyaluronic acid',
'Lactobacillus casei': 'lacti c acid',
'Aspergillus sojae_2': 'polygalacturonase',
'Yarrovia lipolytica': 'microbial oil'}

Verification using in or not in

```
print('Saccharomyces cerevisiae' in product)
print('Saccharomyces cerevisiae' not in product)
```

True
False

dict() function

This function is used to create a dictionary

```
dict_sample = dict(family = 'music', type='pop', year='2022' ,
name='happy new year')
dict_sample
```

{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'happy new year'}

```
# Numerical index is not used to take the dictionary values. It
gives a KeyError
dict_sample[1]
```

KeyError Traceback (most recent call last)
Input In [59], in <cell line: 2>()
1 # Numerical index is not used to take the dictionary values. It gives a KeyError
----> 2 dict_sample[1]

KeyError: 1

copy() function

It returns a shallow copy of the main dictionary

```
sample_original = dict(family = 'music', type='pop', year='2022' ,
name='happy new year')
sample_copy = sample_original.copy()
print(sample_original)
print(sample_copy)
```

{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'happy new year'}
{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'happy new year'}

```
sample_copy['name'] ='HPY'
print(sample_original)
print(sample_copy)
```

{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'happy new year'}
{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'HPY'}

```
# This method can be made using '=' sign
sample_copy = sample_original
print(sample_copy)
print(sample_original)
```

{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'happy new year'}
{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'happy new year'}

```
sample_copy['name'] ='HPY'
print(sample_original)
print(sample_copy)
```

{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'HPY'}
{'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'HPY'}

pop() function

This function is used to remove a specific item from the dictionary

```
sample_original = dict(family = 'music', type='pop', year='2022' ,
name='happy new year')
print(sample_original.pop('type'))
print(sample_original)
```

pop
{'family': 'music', 'year': '2022', 'name': 'happy new year'}

get() function

This method returns the value for the specified key if it is available in the dictionary. If the key is not available, it returns None.

```
sample_original = dict(family = 'music', type='pop', year='2022' , name='happy new year')
print(sample_original.get('family'))
print(sample_original.get(3))
```

music
None

from keys() function

It returns a new dictionary with the certain sequence of the items as the keys of the dictionary and the values are assigned with None

```
keys = {'A', 'T', 'C', 'G'}
sequence = dict.fromkeys(keys)
print(sequence)
```

{'C': None, 'G': None, 'T': None, 'A': None}

update() function

It integrates a dictionary with another dictionary or with an iterable of key:value pairs.

```
product = {'Aspergillus niger': 'inulinase', 'Saccharomyces cerevisiae': 'ethanol',
 'Scheffersomyces sti piti s': 'ethanol', 'Aspergillus sojae_1': 'mannanase',
 'Streptococcus zooepidemicus': 'hyaluronic acid', 'Lactobacillus casei': 'lacti c acid',
 'Aspergillus sojae_2': 'polygalacturonase'}
sample_original = dict(family = 'music', type='pop', year='2022' , name='happy new year')
product.update(sample_original)
print(product)
```

{'Aspergillus niger': 'inulinase', 'Saccharomyces cerevisiae': 'ethanol', 'Scheffersomyces sti piti s': 'ethanol', 'Aspergillus sojae_1': 'mannanase', 'Streptococcus zooepidemicus': 'hyaluronic acid', 'Lactobacillus casei': 'lacti c acid', 'Aspergillus sojae_2': 'polygalacturonase', 'family': 'music', 'type': 'pop', 'year': '2022', 'name': 'happy new year'}

items() function

It returns a list of key:value pairs in a dictionary. The elements in the lists are tuples.

```
product = {'Aspergillus niger': 'inulinase', 'Saccharomyces cerevisiae': 'ethanol',
 'Scheffersomyces sti piti s': 'ethanol', 'Aspergillus sojae_1': 'mannanase',
 'Streptococcus zooepidemicus': 'hyaluronic acid', 'Lactobacillus casei': 'lacti c acid',
 'Aspergillus sojae_2': 'polygalacturonase'}
product.items()
```

dict_items([('Aspergillus niger', 'inulinase'), ('Saccharomyces cerevisiae', 'ethanol'), ('Scheffersomyces sti piti s', 'ethanol'), ('Aspergillus sojae_1', 'mannanase'), ('Streptococcus zooepidemicus', 'hyaluronic acid'), ('Lactobacillus casei', 'lacti c acid'), ('Aspergillus sojae_2', 'polygalacturonase')])

Iterating dictionary

A dictionary can be iterated using the for loop

```
# 'for' loop print all the keys in the dictionary
product = {'Aspergillus niger': 'inulinase', 'Saccharomyces cerevisiae': 'ethanol',
 'Scheffersomyces sti piti s': 'ethanol', 'Aspergillus sojae_1': 'mannanase',
 'Streptococcus zooepidemicus': 'hyaluronic acid', 'Lactobacillus casei': 'lacti c acid',
 'Aspergillus sojae_2': 'polygalacturonase'}
for k in product:
  print(k)
```

Aspergillus niger
Saccharomyces cerevisiae
Scheffersomyces sti piti s
Aspergillus sojae_1
Streptococcus zooepidemicus
Lactobacillus casei
Aspergillus sojae_2

```
# 'for' loop to print the values of the dicti onary by using
values() and other method
product = {'Aspergillus niger': 'inulinase', 'Saccharomyces cerevi-
siae': 'ethanol',
 'Scheffersomyces sti piti s': 'ethanol', 'Aspergillus sojae_1':
'mannanase',
 'Streptococcus zooepidemicus': 'hyaluronic acid', 'Lactobacillus
casei': 'lactic acid',
 'Aspergillus sojae_2': 'polygalacturonase'}
for x in product.values():
  print(x)
print()
# 'for' loop to print the values of the dicti onary by using
values() and other method
for x in product:
  print(product[x])
```

```
inulinase
ethanol
ethanol
mannanase
hyaluronic acid
lactic acid
polygalacturonase

inulinase
ethanol
ethanol
mannanase
hyaluronic acid
lactic acid
polygalacturonase
```

```
# 'for' loop to print the items of the dictionary by using items()
method
product = {'Aspergillus niger': 'inulinase', 'Saccharomyces cerevi-
siae': 'ethanol',
 'Scheffersomyces stipitis': 'ethanol', 'Aspergillus sojae_1': 'man-
nanase',
 'Streptococcus zooepidemicus': 'hyaluronic acid', 'Lactobacillus
casei': 'lactic acid',
 'Aspergillus sojae_2': 'polygalacturonase'}
for x in product.items():
  print(x)
```

```
('Aspergillus niger', 'inulinase')
('Saccharomyces cerevisiae', 'ethanol')
('Scheffersomyces stipitis', 'ethanol')
('Aspergillus sojae_1', 'mannanase')
('Streptococcus zooepidemicus', 'hyaluronic acid')
('Lactobacillus casei', 'lactic acid')
('Aspergillus sojae_2', 'polygalacturonase')
```

```
product = {'Aspergillus niger': 'inulinase', 'Saccharomyces cerevi-
siae': 'ethanol',
 'Scheffersomyces sti piti s': 'ethanol', 'Aspergillus sojae_1':
'mannanase',
 'Streptococcus zooepidemicus': 'hyaluronic acid', 'Lactobacillus
casei': 'lacti c acid',
 'Aspergillus sojae_2': 'polygalacturonase'}
for x, y in product.items():
  print(x, y)
```

```
Aspergillus niger inulinase
Saccharomyces cerevisiae ethanol
Scheffersomyces sti piti s ethanol
Aspergillus sojae_1 mannanase
Streptococcus zooepidemicus hyaluronic acid
Lactobacillus casei lacti c acid
Aspergillus sojae_2 polygalacturonase
```

```
years={1995:'Java', 1972:'C', 1994:'Python'}
list(years)
```

```
[1995, 1972, 1994]
```