Dictionaries & Sets

Dictionaries

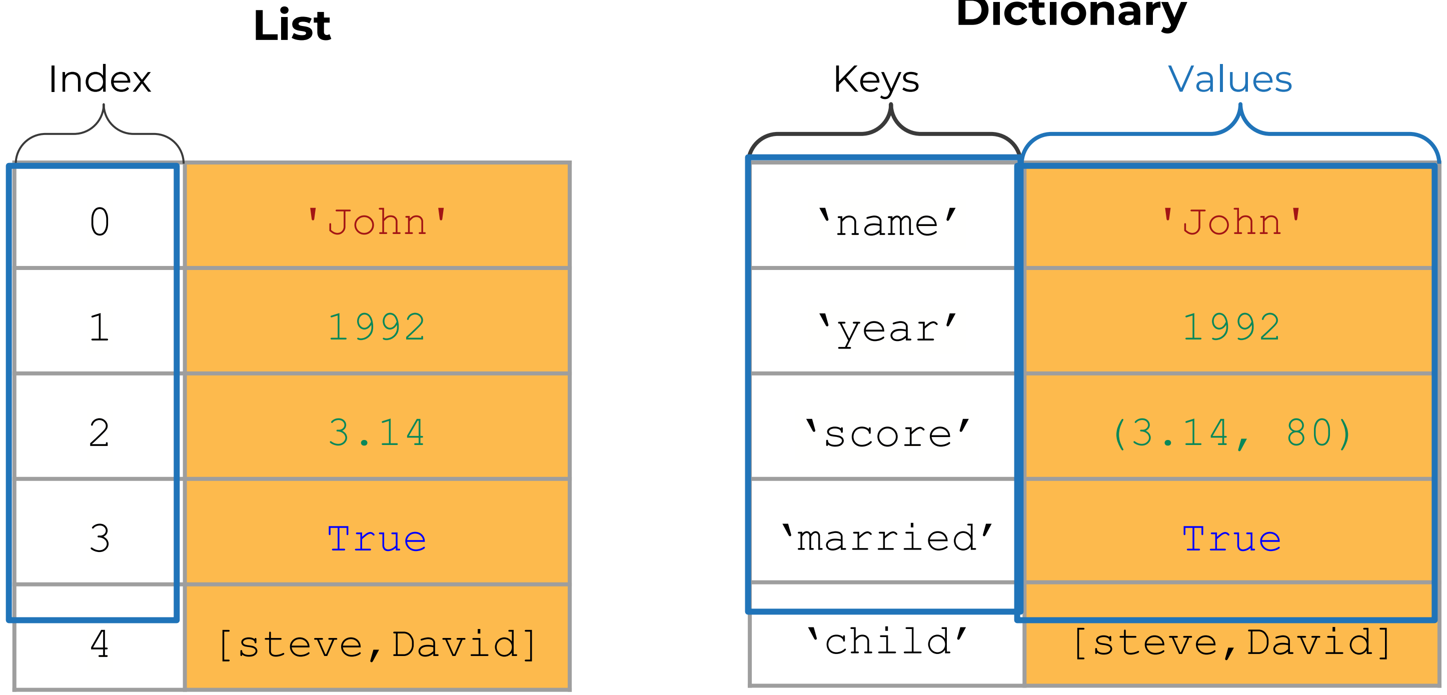
Defining Dictionaries

```
Dic = { <key1> : <value1>, <key2> : <value2> , <key3> : <value3> }
```

```
dictionary = dict([
    ('name', 'john'),
    ('year', 1995),
    ('score', [3.5, 80])
])
```

```
dictionary = dict(
    name = 'john',
    year = 1995,
    score = [3.5, 80])
```

Accessing Elements from Dictionary

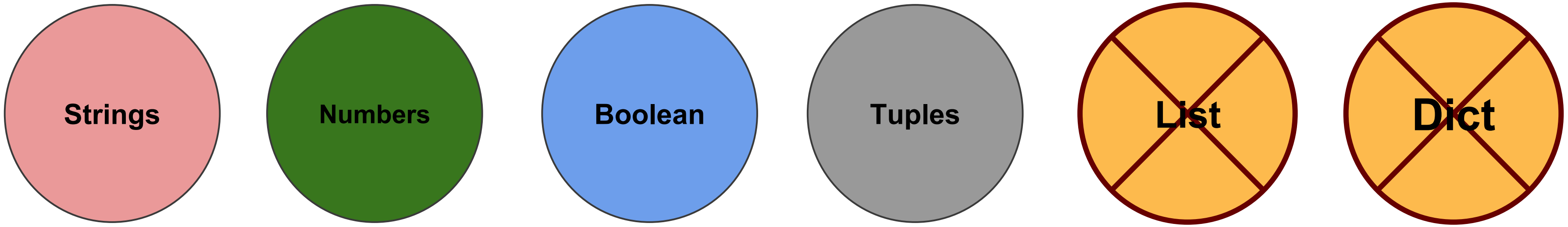


```
>>> Dictionary['name']
'John'
```

Accessing Elements, Keys & Values

```
dic = {'a': [0, 1], 1: 'b', True: 'c', (1, (‘x’, False)): ‘d’}

    name_of_dictionary[key]              →    dic[‘a’]
               Or                                 Or
    name_of_dictionary.get(key)          →    dic.get(‘a’)


    Name_of_dictionary.keys()            →    dic.keys()

    name_of_dictionary.values()          →    dic.values()
```

Keys of dictionary are Immutable



```
dic = {'a': [0, 1], 1: 'b', True: 'c', (1, (‘x’, False)): d'}
```

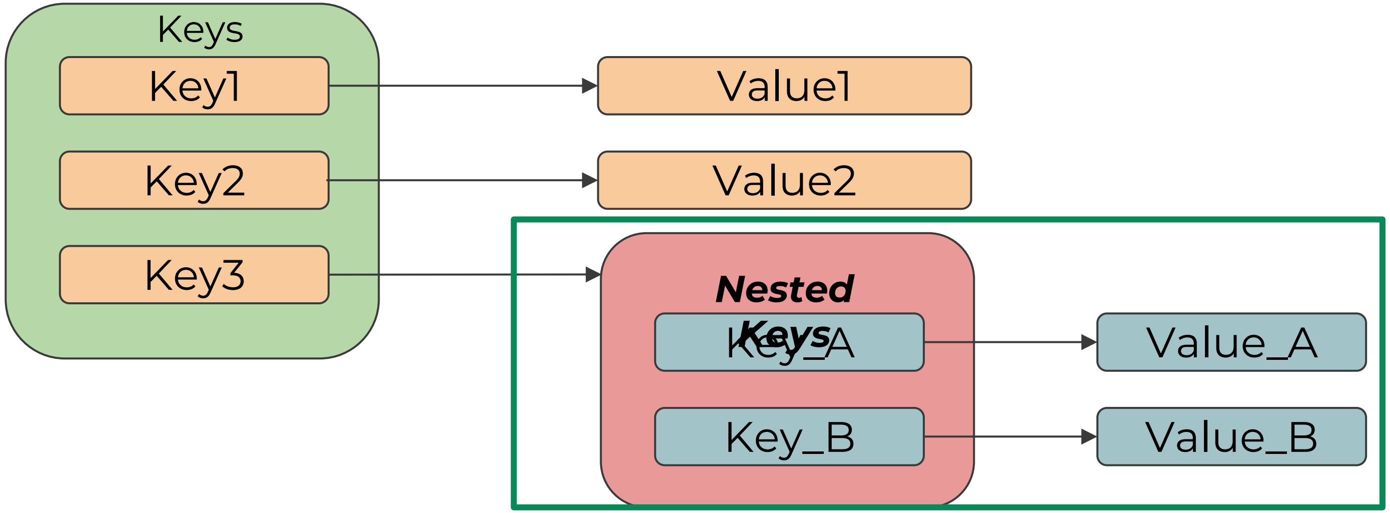
Modifying values

```
info = {'name': 'John', 'age': 24}

info[      ]

info['score'] = 80

info: {'name': 'John', 'age': 24, 'score': 80}

info.pop('age')

info.popitem()
```

Update existing value

Add new key-value pair

Nested Dictionary



```
dic = {'a': 0, 1: 'b', True: {1: 'd', (‘x’, False): ‘d'}}
```

```
dic[True][1]
        or
dic.get(True).get(1)
```

Dictionaries: Code Demo

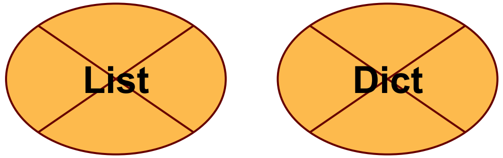
Sets

Python Sets

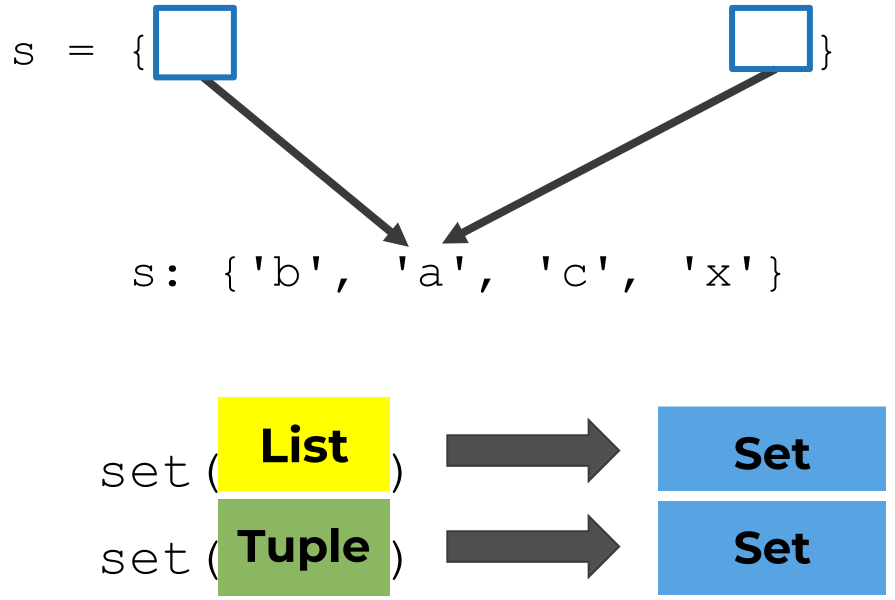
Collections of well-defined distinct items

Items are unordered.

Items are immutable.

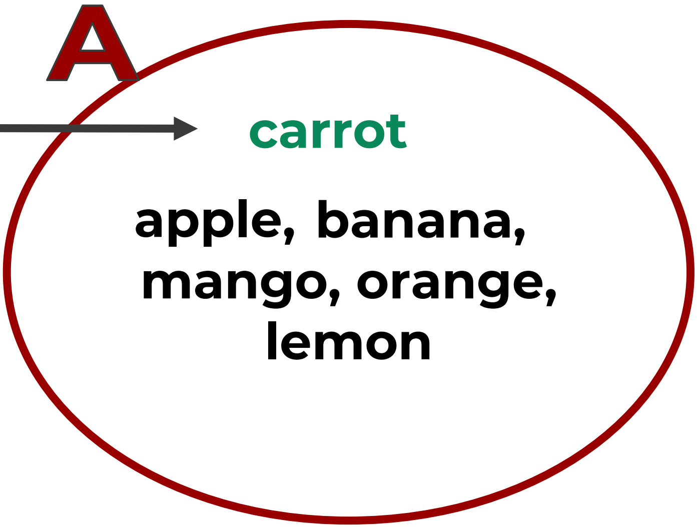


Defining Sets



Modifying a set

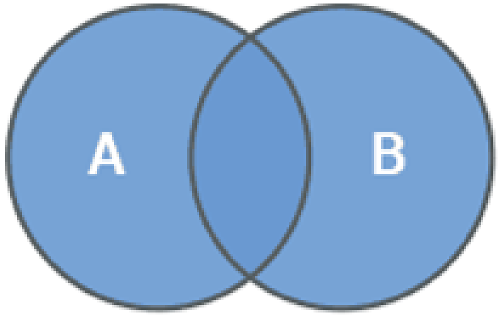
```
A = { “apple”, “banana”, “mango”, “orange”, “lemon” }


A.add(           )


A.remove(‘          )
```



Some Sets Operations

Some Sets Operations: Union

```
A = { “apple”, “banana”, “mango”, “orange”, “lemon” }

B = { “orange”, “pineapple”, “watermelon”, “grapes”, “lemon” }
```



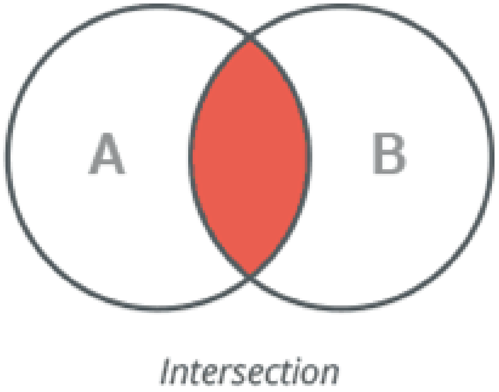
Union

```
A.union(B)
    OR
  A | B
```

```
{'mango', 'banana', 'pineapple', 'apple',
'lemon', 'watermelon', 'grapes', 'orange'}
```

Some Sets Operations: Intersection

```
A = { “apple”, “banana”, “mango”, “orange”, “lemon” }

B = { “orange”, “pineapple”, “watermelon”, “grapes”, “lemon” }
```

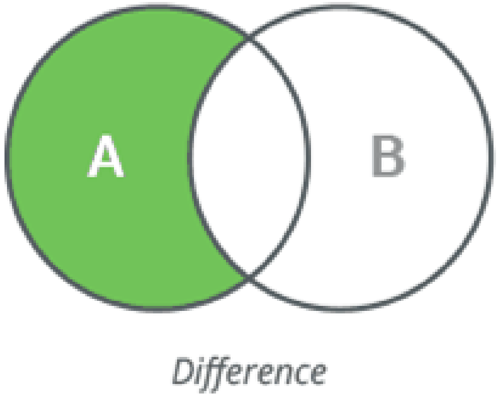


```
A.intersection(B)
       OR
     A & B
```

```
{'orange', 'lemon'}
```

Some Sets Operations: Difference

```
A = { “apple”, “banana”, “mango”, “orange”, “lemon” }

B = { “orange”, “pineapple”, “watermelon”, “grapes”, “lemon” }
```

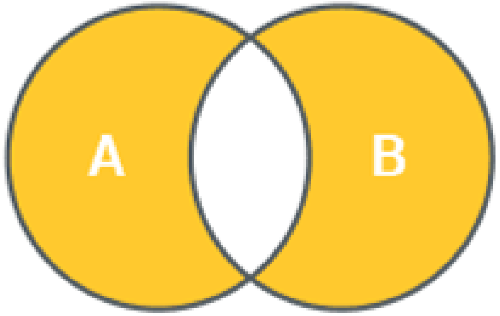


```
A.difference(B)
OR
A - B
```

```
{'banana', 'apple', 'mango'}
```

Some Sets Operations: Symmetric Difference

```
A = { "apple", "banana", "mango", "orange", "lemon" }

B = { "orange", "pineapple", "watermelon", "grapes", "lemon" }
```



Symmetric Difference

```
A.symmetric_difference(B)
OR
A ^ B
```

```
{'watermelon', 'grapes', 'pineapple',
'banana', 'mango', 'apple'}
```

Frozen sets

Frozen sets

```
frozenset(iterable_object_name)
```

Return Type: an equivalent **frozenset object.**

```
A = frozenset([1, 2, 3, 4])
```

~~A.add(8)~~
~~A.remove(1)~~

Modifying methods are not allowed

Non-modifying methods are allowed

copy(), difference(), intersection(), isdisjoint(), issubset(), issuperset(), symmetric_difference() and union()

Sets: Code Demo