Dictionary

Get All Keys

```
>>> a = {"1":1, "2":2, "3":3}
>>> b = {"2":2, "3":3, "4":4}
>>> a.keys()
['1', '3', '2']
```

Get Key and Value

```
>>> a = {"1":1, "2":2, "3":3}
>>> a.items()
```

Find Same Keys

```
>>> a = {"1":1, "2":2, "3":3}
>>> b = {"2":2, "3":3, "4":4}
>>> [_ for _ in a.keys() if _ in b.keys()]
['3', '2']
>>> # better way
>>> c = set(a).intersection(set(b))
>>> list(c)
['3', '2']
>>> # or
>>> [_ for _ in a if _ in b]
['3', '2']
[('1', 1), ('3', 3), ('2', 2)]
```

Set a Default Value

```
>>> # intuitive but not recommend
>>> d = {}
>>> key = "foo"
>>> if key not in d:
...     d[key] = []
...

# using d.setdefault(key[, default])
>>> d = {}
>>> key = "foo"
>>> d.setdefault(key, [])
[]
>>> d[key] = 'bar'
>>> d
{'foo': 'bar'}

# using collections.defaultdict
>>> from collections import defaultdict
>>> d = defaultdict(list)
>>> d["key"]
[]
>>> d["foo"]
[]
>>> d["foo"].append("bar")
>>> d
defaultdict(<class 'list'>, {'key': [], 'foo': ['bar']})
```

`dict.setdefault(key[, default])` returns its default value if *key* is not in the dictionary. However, if the key exists in the dictionary, the function will return its value.

```
>>> d = {}
>>> d.setdefault("key", [])
[]
```

```
>>> d["key"] = "bar"
>>> d.setdefault("key", [])
'bar'
```

Update Dictionary

```
>>> a = {"1":1, "2":2, "3":3}
>>> b = {"2":2, "3":3, "4":4}
>>> a.update(b)
>>> a
{'1': 1, '3': 3, '2': 2, '4': 4}
```

Merge Two Dictionaries

Python 3.4 or lower

```
>>> a = {"x": 55, "y": 66}
>>> b = {"a": "foo", "b": "bar"}
>>> c = a.copy()
>>> c.update(b)
>>> c
{'y': 66, 'x': 55, 'b': 'bar', 'a': 'foo'}
```

Python 3.5 or above

```
>>> a = {"x": 55, "y": 66}
>>> b = {"a": "foo", "b": "bar"}
>>> c = {**a, **b}
>>> c
{'x': 55, 'y': 66, 'a': 'foo', 'b': 'bar'}
```

Emulating a Dictionary

```
>>> class EmuDict(object):
...   def __init__(self, dict_):
...     self._dict = dict_
...   def __repr__(self):
...     return "EmuDict: " + repr(self._dict)
...   def __getitem__(self, key):
...     return self._dict[key]
...   def __setitem__(self, key, val):
...     self._dict[key] = val
...   def __delitem__(self, key):
...     del self._dict[key]
...   def __contains__(self, key):
...     return key in self._dict
...   def __iter__(self):
...     return iter(self._dict.keys())
...
>>> _ = {"1":1, "2":2, "3":3}
>>> emud = EmuDict(_)
>>> emud  # __repr__
EmuDict: {'1': 1, '2': 2, '3': 3}
>>> emud['1']  # __getitem__
1
>>> emud['5'] = 5  # __setitem__
>>> emud
EmuDict: {'1': 1, '2': 2, '3': 3, '5': 5}
>>> del emud['2']  # __delitem__
>>> emud
EmuDict: {'1': 1, '3': 3, '5': 5}
>>> for _ in emud:
...     print(emud[_], end=' ')  # __iter__
... else:
...     print()
...
1 3 5
```

```
>>> '1' in emud  # __contains__
True
```