dictionaries

November 11, 2025

A dictionary is a collection of unordered, modifiable(mutable) paired (key: value) data type.

Creating a Dictionary

To create a dictionary we use curly brackets, {} or the dict() built-in function.

[9]:
```
# syntax
empty_dict = {}
# Dictionary with data values
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
```

Example:

[14]:
```
person = {
    'first_name':'Abc',
    'last_name':'def',
    'age':250,
    'country':'Finland',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
```

The dictionary above shows that a value could be any data types:string, boolean, list, tuple, set or a dictionary.

Dictionary Length

It checks the number of 'key: value' pairs in the dictionary.

[21]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print(len(dct)) # 4
```

```
4
```

Example:

[26]:
```
person = {
    'first_name':'Abc',
    'last_name':'def',
    'age':250,
    'country':'Finland',
    'is_married':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
print(len(person)) # 7
```

```
7
```

Accessing Dictionary Items

We can access Dictionary items by referring to its key name.

[34]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print(dct['key1']) # value1
print(dct['key4']) # value4
```

```
value1
value4
```

Example:

[50]:
```
person = {
    'first_name':'Ali',
    'last_name':'def',
    'age':250,
    'country':'Finland',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
print(person['first_name']) # Ali
print(person['country'])    # def
print(person['skills'])     # ['JavaScript', 'React', 'Node', 'MongoDB',␣
 ↪'Python']
print(person['skills'][0])  # JavaScript
print(person['address']['street']) # Space street
print(person['city'])       # Error
```

```
Ali
Finland
['JavaScript', 'React', 'Node', 'MongoDB', 'Python']
JavaScript
Space street
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[50], line 18
     16 print(person['skills'][0])  # JavaScript
     17 print(person['address']['street']) # Space street
---> 18 print(person['city'])

KeyError: 'city'
```

Accessing an item by key name raises an error if the key does not exist. To avoid this error first we have to check if a key exist or we can use the **get method**. The get method returns None, which is a NoneType object data type, if the key does not exist.

[52]:
```
person = {
    'first_name':'Ali',
    'last_name':'def',
    'age':250,
    'country':'Finland',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
print(person.get('first_name')) # Ali
print(person.get('country'))    # def
print(person.get('skills')) #['HTML','CSS','JavaScript', 'React', 'Node',␣
 ↪'MongoDB', 'Python']
print(person.get('city'))   # None
```

```
Ali
Finland
['JavaScript', 'React', 'Node', 'MongoDB', 'Python']
None
```

Adding Items to a Dictionary

We can add new key and value pairs to a dictionary

[59]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
```

```
dct['key5'] = 'value5'
```

Example:

[64]:
```
person = {
    'first_name':'Ali',
    'last_name':'def',
    'age':250,
    'country':'Finland',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
        }
}
person['job_title'] = 'Instructor'
person['skills'].append('HTML')
print(person)
```

```
{'first_name': 'Ali', 'last_name': 'def', 'age': 250, 'country': 'Finland',
'is_marred': True, 'skills': ['JavaScript', 'React', 'Node', 'MongoDB',
'Python', 'HTML'], 'address': {'street': 'Space street', 'zipcode': '02210'},
'job_title': 'Instructor'}
```

Modifying Items in a Dictionary

We can modify items in a dictionary

[70]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
dct['key1'] = 'value-one'
```

Example:

[79]:
```
person = {
    'first_name':'ali',
    'last_name':'def',
    'age':250,
    'country':'Finland',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
person['first_name'] = 'Eyob'
person['age'] = 252
print(person)
```

```
{'first_name': 'Eyob', 'last_name': 'def', 'age': 252, 'country': 'Finland',
'is_marred': True, 'skills': ['JavaScript', 'React', 'Node', 'MongoDB',
'Python'], 'address': {'street': 'Space street', 'zipcode': '02210'}}
```

Checking Keys in a Dictionary

We use the in operator to check if a key exist in a dictionary

[85]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print('key2' in dct) # True
print('key5' in dct) # False
```

```
True
False
```

Removing Key and Value Pairs from a Dictionary

- **pop(key): removes the item with the specified key name:**
- **popitem(): removes the last item**
- **del: removes an item with specified key name**

[95]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
dct.pop('key1') # removes key1 item
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
dct.popitem() # removes the last item
del dct['key2'] # removes key2 item
```

Example:

[100]:
```
person = {
    'first_name':'Asabeneh',
    'last_name':'Yetayeh',
    'age':250,
    'country':'Finland',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
person.pop('first_name')        # Removes the firstname item
person.popitem()                # Removes the address item
del person['is_marred']        # Removes the is_married item\
```

[102]:
```
print(person)
```

```
{'last_name': 'Yetayeh', 'age': 250, 'country': 'Finland', 'skills':
['JavaScript', 'React', 'Node', 'MongoDB', 'Python']}
```

Changing Dictionary to a List of Items

The items() method changes dictionary to a list of tuples.

[108]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print(dct.items()) # dict_items([('key1', 'value1'), ('key2', 'value2'),␣
  ↪('key3', 'value3'), ('key4', 'value4')])
```

```
dict_items([('key1', 'value1'), ('key2', 'value2'), ('key3', 'value3'), ('key4',
'value4')])
```

Clearing a Dictionary

If we don't want the items in a dictionary we can clear them using clear() method

[114]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print(dct.clear()) # None
```

```
None
```

Deleting a Dictionary

If we do not use the dictionary we can delete it completely

[120]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
del dct
```

Getting Dictionary Keys as a List

The keys() method gives us all the keys of a a dictionary as a list.

[126]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
keys = dct.keys()
print(keys)     # dict_keys(['key1', 'key2', 'key3', 'key4'])
```

```
dict_keys(['key1', 'key2', 'key3', 'key4'])
```

Getting Dictionary Values as a List

The values method gives us all the values of a a dictionary as a list.

[130]:
```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
values = dct.values()
print(values)     # dict_values(['value1', 'value2', 'value3', 'value4'])
```

```
dict_values(['value1', 'value2', 'value3', 'value4'])
```

Exercise

- **Create an empty dictionary called dog**

- **Add name, color, breed, legs, age to the dog dictionary**
- **Create a student dictionary and add first_name, last_name, gender, age, marital status, skills, country, city and address as keys for the dictionary**
- **Get the length of the student dictionary**
- **Get the length of the student dictionary**
- **Get the length of the student dictionaryGet the length of the student dictionary**
- **Get the value of skills and check the data type, it should be a list**
- **Modify the skills values by adding one or two skills**
- **Get the dictionary keys as a list**
- **Get the dictionary values as a list**
- **Change the dictionary to a list of tuples using items() method**
- **Delete one of the items in the dictionary**
- **Delete one of the dictionaries**

```
[ ]:
```