TIN HỌC CƠ SỞ

Phần 2: Ngôn ngữ lập trình Python

CHƯƠNG 11.

KIỂU DỮ LIỆU LIST, TUPLE, SET, DICTIONARY

Tin cơ sở (LT): 010100229802 - DHKL16A1HN, - DHKL16A2HN

GV. Cao Diệp Thắng

Mục tiêu chương

Nắm vững:

- Khái niệm, cú pháp khai báo, khởi tạo các kiểu dữ liệu List, Tuple, Set, Dictionary
- Cách viết code với các kiểu dữ liệu List, Tuple, Set, Dictionary

Nội dung

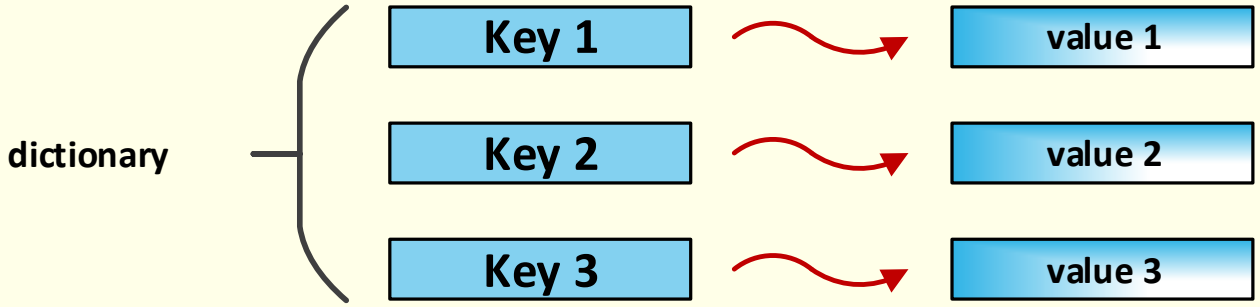
1. Kiểu dữ liệu tuần tự (sequential data type)
2. List (danh sách) trong Python
3. Tuple (hàng)
4. Set (tập hợp)
5. Dictionary (từ điển)

11.6. DICTIONARY

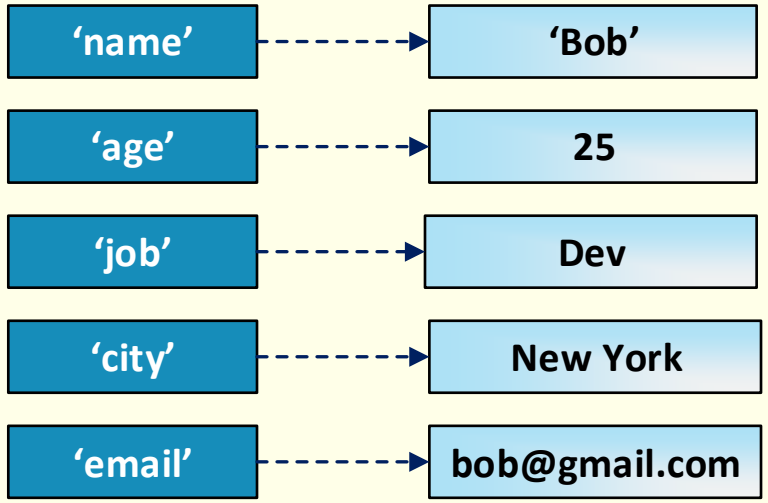
11.6.1. Khái niệm

- Từ điển (Dictionary) trong Python tương đương với khái niệm mục từ và ngữ nghĩa trong từ điển thông thường.
- Từ điển trong Python được định nghĩa gồm hai thành phần là: **khóa (key)** và **giá trị (value)** với các tính chất sau:
  - Một mục là một cặp (**pair**) khóa (**key**) và giá trị (**value**)
  - Các **khóa (key) là duy nhất** trong dictionary, còn **value thì có thể trùng**, như vậy có thể xem từ điển như một loại tập hợp (set)
  - Các khóa **không sắp thứ tự như từ điển thông thường**
  - Chỉ **dữ liệu bất biến (immutable) mới được dùng làm khóa**, kiểu dữ liệu của key có thể là **string**, **number** hoặc **tuple**
  - Giá trị được lưu có thể là một **danh sách** hoặc **danh sách trong danh sách**, một **chuỗi**, một **số**, một **đối tượng bất kỳ** trong python ...

Từ điển là ánh xạ giữa **key** và **value** (giữa **từ** và **giải nghĩa**)



Minh họa quan hệ giữa key và value



ví dụ về từ điển

Một cặp trong từ điển viết ở dạng <key>:<value>

11.6.1. Tạo dictionary

Cú pháp: `tên_dictionary = {key1:value1,key2:value2,…}`

Ví dụ 11.54

```
d1 = { }                              # từ điển rỗng
d2 = { 1: 'one', 2: 'two' }           # từ điển các cặp số-chuỗi
d3 = { 'one': 1, 'two': 2 }           # từ điển các cặp chuỗi-số
d4 = { 'tên': 'nam', 'sđt':0}         # từ điển hỗ hợp
```

Ví dụ 11.55

```
>>> dic1 = {1: "one", 2:"two", 3:"three", 4:"four"}
>>> print(dic1)
{1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

- Như vậy Python coi từ điển là dạng mở rộng của tập hợp
- Trường hợp lấy dữ liệu từ nguồn khác, cách thích hợp nhất là sử dụng hàm khởi tạo **dict()**

Ví dụ 11.56

```
d5 = dict(d4)                    # lấy dữ liệu từ d4
print(d5)
{'tên': 'nam', 'sđt': 0}         # {'tên': 'nam', 'sđt': 0}
d6 = dict()                      # tạo từ điển rỗng
print(d6)
{}                               #{}
```

*Python cũng cho phép tạo từ điển bằng **bộ suy diễn** từ điển, với cú pháp tương tự như bộ suy diễn danh sách*

Ví dụ 11.57 Tạo một từ điển gồm các bộ khóa là số tự nhiên nhỏ hơn N và giá trị tương ứng là lập phương của nó.

```
>>> n = int(input('N = '))
N = 4
>>> d = { i: i * i * i for i in range(n) }
>>> print(d)
{0: 0, 1: 1, 2: 8, 3: 27}
```

Ví dụ 11.58 .Tạo từ điển có khóa các ký tự xuất hiện trong từ S và giá trị là số lần xuất hiện của ký tự đó trong S

```
>>> S = input('N = ')
N = Hello
>>> d = {w:S.count(w) for w in S}   #các ký tự 'H', 'e', 'o' có số lần xuất hiện trong S là 1,
>>> print(d)                        #ký tự 'l' xuất hiện 2 lần
{'H': 1, 'e': 1, 'l': 2, 'o': 1}
```

***Lưu ý**: chỉ những loại dữ liệu immutable (không thể thay đổi) mới có thể dùng làm key của từ điển.*

Ví dụ 11.59

```
>>> dic1 = { (1,2,3):"abc", 3.1415:"abc"}
>>> dic2 = { [1,2,3]:"abc"}
Traceback (most recent call last):
File "<pyshell#16>", line 1, in <module>
dic = { [1,2,3]:"abc"}
TypeError: unhashable type: 'list'
```

#dic1 có hai phần tử với key là một tuple : (1,2,3) và một hằng (3.1415) là hợp lệ.

#dic2 sử dụng một list [1,2,3] làm key là không hợp lệ vì list là có thể thay đổi được

11.6.2. Truy xuất giá trị trong dictionary

Cú pháp: **tên_dictionar[key]**

Ví dụ 11.60

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck",
4:"bear", 5:"ant"}
>>> print(dic_animals)
{1: 'elephant', 2: 'dog', 3: 'duck', 4: 'bear', 5: 'ant'}
>>>
>>> print(dic_animals[1])
elephant
```

11.6.3. Cập nhật giá trị/thêm mới phần tử vào dictionary

Cú pháp: **ten_dictionary**[key] = **value**

Ví dụ 11.61

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck",
4:"bear", 5:"ant"}
>>> print(dic_animals[1])
elephant
>>>
>>> dic_animals[2] = "lion"
>>> print("After updated: ", dic_animals)
After updated:  {1: 'elephant', 2: 'lion', 3: 'duck', 4:
'bear', 5: 'ant'}
>>>
>>> dic_animals[6] = "python"
>>> print("After inserted: ", dic_animals)
After inserted:  {1: 'elephant', 2: 'lion', 3: 'duck', 4:
'bear', 5: 'ant', 6: 'python'}
```

11.6.4. Xóa phần tử trong dictionary

a. Xóa phần tử theo khóa Cú pháp: **del tên_dictionary[key]**

b. Xóa tất cả các phần tử trong dictionary Cú pháp: **tên_dictionary.clear()**

c. Xóa luôn dictionary Cú pháp: **del tên_dictionary**

Ví dụ 11.62

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck", 4:"bear", 5:"ant", 6:'bird'}
>>> print("Original dic:", dic_animals)
Original dic: {1: 'elephant', 2: 'dog', 3: 'duck', 4: 'bear', 5: 'ant', 6: 'bird'}
>>> del dic_animals[1]
>>> print("After removed key 1:", dic_animals)
After removed key 1: {2: 'dog', 3: 'duck', 4: 'bear', 5: 'ant', 6: 'bird'}
>>> dic_animals.clear()
>>> print("After removed all:", dic_animals)
After removed all: {}
>>> del dic_animals
>>> print("After deleted dic:", dic_animals)
Traceback (most recent call last):
File "<pyshell#24>", line 1, in <module>
print("After deleted dic:", dic_animals)
NameError: name 'dic_animals' is not defined
```

Lưu ý: Đối với key trong dictionary

- *Key là duy nhất trong dictionary, không có key trùng nhau*
- *Key có giá trị không thay đổi, do đó chỉ có thể cập nhật value, không thể cập nhật key.*

11.6.5. Các phương thức cơ bản trong dictionary

a. Chiều dài của dictionary

Cú pháp: `len(ten_dictionary)`

Ví dụ 11.63

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck", 4:"bear", 5:"ant"}
>>> print("Length:", len(dic_animals))
Length: 5
```

b. In dictionary dưới dạng chuỗi

Cú pháp: `str(ten_dictionary)`

Ví dụ 11.64

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck", 4:"bear", 5:"ant"}
>>> print(type(dic_animals))
<class 'dict'>
>>> convert_to_str = str(dic_animals)
>>> print(convert_to_str)
{1: 'elephant', 2: 'dog', 3: 'duck', 4: 'bear', 5: 'ant'}
>>> print(type(convert_to_str))
<class 'str'>
```

c.Tạo bản sao của dictionary

Cú pháp: **ten_dictionary_đích = ten_dictionary_nguồn.copy**

Ví dụ 11.65

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck", 4:"bear", 5:"ant"}
>>> print("Source:", dic_animals)
Source: {1: 'elephant', 2: 'dog', 3: 'duck', 4: 'bear', 5: 'ant'}
>>> dic_animals_copy = dic_animals.copy()
>>> print("Dest:", dic_animals)
Dest: {1: 'elephant', 2: 'dog', 3: 'duck', 4: 'bear', 5: 'ant'}
```

Ví dụ 11.66

```
>>> Tu_dien_thu_cung = {1:'Chim Hoa mi', 2:'Ca Ba duoi', 3: 'Ran', 4: 'Tran',
5:'Ky Nhong', 6:'Ca Ong tien'}
>>> Ban_sao = Tu_dien_thu_cung.copy()
>>> print("\nBan sao:")
>>> for i in Ban_sao:
        Ban_sao[i] = Ban_sao[i].upper()
        print(i, ':', Ban_sao[i])
>>> print("\nBan goc:")
>>> for i in Tu_dien_thu_cung:
        print(i, ':', Tu_dien_thu_cung[i])
```

```
Ban sao:
1 : CHIM HOA MI
2 : CA BA DUOI
3 : RAN
4 : TRAN
5 : KY NHONG
6 : CA ONG TIEN
Ban goc:
1 : Chim Hoa mi
2 : Ca Ba duoi
3 : Ran
4 : Tran
5 : Ky Nhong
6 : Ca Ong tien
```

d.Tạo dictionary với danh sách các key từ sequence

Cú pháp: **ten_dictionary=dict.fromkeys(seq[,value])**

Ví dụ 11.67

```
>>> list_1 = ["one", "two", "three", "four"]
>>> dict1 = dict.fromkeys(list_1)
>>> print(dict1)
{'one': None, 'two': None, 'three': None, 'four': None}
>>> dict1 = dict.fromkeys(list_1, 15)
>>> print(dict1)
{'one': 15, 'two': 15, 'three': 15, 'four': 15}
```

e. Lấy giá trị (value) dựa trên key trong dictionary

Cú pháp: **ten_dictionary=dict.get(key)**

Ví dụ 11.68

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck", 4:"bear", 5:"ant"}
>>> print(dic_animals.get(3))
duck
```

- Từ điển cho phép lấy giá trị tương ứng với **khóa k** bằng phương thứ **get(k),** Nhưng có thể dùng chỉ mục thuận tiện hơn nhiều

Ví dụ 11.69

```
>>> d = { 1: 'one', 2: 'two', 3: 'three' }
>>> print('d.get(3) = ', d.get(3))
d.get(3) =  three
>>> print('d[3] = ', d[3])
d[3] =  three
>>> print('d.get(9) = ', d.get(9)) #9 không có trong danh sách khóa
d.get(9) =  None
>>> print('d[9] = ', d[9])
Traceback (most recent call last):
  File "<pyshell#95>", line 1, in <module>
    print('d[9] = ', d[9])
KeyError: 9
>>> d[3] = 'ba' #Cap nhat cap 3: 'three' thanh 3: 'ba'
>>> d[4] = 'bon' #them moi cap 4: 'bon'
>>> print(d)
{1: 'one', 2: 'two', 3: 'ba', 4: 'bon'}
```

Lưu ý: *Khi dữ liệu không có trong từ điển, get(k) trả về None còn chỉ mục (index) phát sinh ngoại lệ KeyError*

g. Trả về danh sách các bộ tuple (key,value) của dictionary

Cú pháp: **tên_dictionary=dict.items()**

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck", 4:"bear", 5:"ant"}
>>> print(dic_animals.items())
dict_items([(1, 'elephant'), (2, 'dog'), (3, 'duck'), (4, 'bear'), (5, 'ant')])
```

h. Trả về danh sách các value của dictionary

Cú pháp: **tên_dictionary=dict.values()**

Ví dụ 11.70

```
>>> dic_animals = {1:"elephant", 2:"dog", 3:"duck", 4:"bear", 5:"ant"}
>>> print(dic_animals.values())
dict_values(['elephant', 'dog', 'duck', 'bear', 'ant'])
```

i. Cập nhật dictionary bằng cách thêm dictionary khác vào

Cú pháp: **tên_dictionary=dict.update(tên_dictionary_added)**

Ví dụ 11.71

```
>>> dic_nums = {1:"one", 2:"two", 3:"three"}
>>> dic_added = {4:"four", 5:"five"}
>>> dic_nums.update(dic_added)
>>> print(dic_nums)
{1: 'one', 2: 'two', 3: 'three', 4: 'four', 5: 'five'}
```

j. Duyệt dictionary

Python cung cấp nhiều phương thức lấy về các nhóm dữ liệu trong từ điển, ta có thể dễ dàng duyệt chúng với vòng lặp **for**

Ví dụ 11.72

```
>>> Tu_dien_thu_cung = {1:'Chim Hoa mi', 2:'Ca Ba duoi', 3:
'Ran', 4: 'Tran', 5:'Ky Nhong', 6:'Ca Ong tien'}
>>> for i in Tu_dien_thu_cung:
      print(i, ':', Tu_dien_thu_cung[i])
```

Kết quả:

```
key 1 : Chim Hoa mi
key 2 : Ca Ba duoi
key 3 : Ran
key 4 : Tran
key 5 : Ky Nhong
key 6 : Ca Ong tien
```

Ví dụ 11.73

```
d = { 1: 'one', 3: 'three', 2: 'two', 0: 'zero' }
# duyệt từ điển theo khóa: 1, 3, 2, 0
>>> for i in d: print(i, end = ' ')
1 3 2 0
# duyệt từ điển theo khóa: 1, 3, 2, 0
for i in d.keys(): print(i, end = ' ')
1 3 2 0
# duyệt theo cặp khóa-giá trị: (1, 'one'), (3,
'three'),..
for i in d.items(): print(i, end = ' ')
(1, 'one') (3, 'three') (2, 'two') (0, 'zero')
# duyệt theo giá trị: 'one', 'three', 'two', 'zero'
for i in d.values(): print(i, end = ' ')
one three two zero
```

Câu hỏi thảo luận

1. Nêu khái niệm List? Tuple? Set? Dictionary?
2. Sự khác nhau giữa List và Tuple?
3. Nêu ít nhất 05 thao tác cơ bản với list? Cho ví dụ?
4. Từ khóa key trong dictionary chỉ có thể là những kiểu dữ liệu nào?