Softech®
Gather to thrive together

Aptech™
COMPUTER EDUCATION
Unleash your potential

BÀI 09

Dictionaries Python
Kiểu dữ liệu từ điển

Tóm Tắt Nội Dung

Trong bài học này chúng ta sẽ đi tìm hiểu lần lượt các nội dung

1. Tổng quan về kiểu dữ liệu từ điển Dictionaries
2. Phương thức xử lý với từ điển Dictionaries

9.1 Tổng quan về kiểu dữ liệu từ điển Dictionaries

Khái niệm về Dictionaries

- Trong Python, Dictionaries (Từ điển) là một loại cấu trúc dữ liệu mạnh mẽ và linh hoạt, cho phép bạn lưu trữ và truy xuất dữ liệu theo cặp khóa-giá trị
- Dictionaries được biểu thị bằng dấu ngoặc nhọn "{}" và được sử dụng rộng rãi trong Python để lưu trữ và quản lý các cặp dữ liệu có tính chất "khóa-giá trị".
- **Mỗi cặp trong từ điển bao gồm hai phần:**

1. **Khóa (Key):** Đây là giá trị duy nhất dùng để xác định và truy xuất giá trị tương ứng. Khóa có thể là một đối tượng bất kỳ có thể băm (hashable) như chuỗi (string), số nguyên (integer), bộ (tuple), v.v

2. **Giá trị (Value):** Đây là giá trị được lưu trữ liên kết với khóa tương ứng. Giá trị có thể là bất kỳ đối tượng Python nào

9.1 Tổng quan về kiểu dữ liệu từ điển Dictionaries

Ví dụ

```
person = {
    "name": "John",
    "age": 30,
    "city": "New York"
}
```

Trong ví dụ này, từ điển "**person**" chứa ba cặp khóa - giá trị:

- Khóa **"name"** liên kết với giá trị **"John"**
- Khóa **"age"** liên kết với giá trị **30** (kiểu số nguyên).
- Khóa **"city"** liên kết với giá trị **"New York"**

9.1 Tổng quan về kiểu dữ liệu từ điển Dictionaries

Ví dụ

```
data = {
    "numbers": [1, 2, 3, 4, 5],
    "info": {
            "name": "Alice", "age": 25, "occupation": "Engineer"
            },
    "some_function": lambda x: x ** 2
}
```

- Qua ví dụ này và ví dụ trước bạn thấy **value** có thể có rất nhiều kiểu giá trị

  Kiểu số, chữ, set, function, ….

- Từ điền lồng vào nhau như value của key info ở trên

9.1 Tổng quan về kiểu dữ liệu từ điển Dictionaries

Đặc tính của dữ liệu từ điển

- Từ điển là một cấu trúc dữ liệu **không có thứ tự**, điều này có nghĩa là các phần tử không được sắp xếp theo vị trí thứ tự, mà được truy xuất thông qua khóa của chúng
- Bạn có thể thêm, sửa đổi và xóa các cặp khóa-giá trị trong từ điển một cách linh hoạt
- Không cho phép trùng lặp **key**

```
person = {
    "name": "John",
    "age": 30,
    "city": "New York“,
    "city": “WDC"
}
```

Lỗi vì trùng lặp key **city**

9.2 Phương thức xử lý với từ điển Dictionaries

Phương thức len()

Kiểm tra độ dài của từ điển

```
print(len(person)) # cho kết quả: 3
```

Phương thức type()

Kiểm tra kiểu dữ liệu của từ điển

```
print(type(person)) #cho kết quả <class 'dict'>
```

9.2 Phương thức xử lý với từ điển Dictionaries

Truy cập đến các phần tử của từ điển

- Bạn có thể lấy giá trị (value) của một phần tử dựa vào key của chúng

```
name = person["name"]
print(name)
```

- Ngoài ra bạn có thể sử dụng phương thức **get()** để lấy

```
name = person.get("name")
print(name)
```

- Lấy danh sách key của từ điển với phương thức **keys()**

```
keyList = person.keys()
print(keyList)
```

9.2 Phương thức xử lý với từ điển Dictionaries

Truy cập đến các phần tử của từ điển

- Lấy tất cả các value (giá trị) của từ điển với phương thức **values()**

```
vList = person.values()
print(vList)
```

- Lấy cả key và value của các phần tử trong từ điển với phương thức **items()**

```
itemList = person.items()
print(itemList)
#Kết quả in ra được:
[(‘name', 'Jonh'), (‘age', 30), (‘city', ‘New York’)]
```

Trả về một list các tuple

9.2 Phương thức xử lý với từ điển Dictionaries

Check sự tồn tại của một phần tử trong từ điển

- Ví dụ check city có tồn tại trong từ điển person không

```
if  "city" in person:
  print("Yes, city'  là một key trong từ điển person")
```

Thay đổi một phần tử trong từ điển

- Thay đổi giá trị (value) cho một phần tử (key) trong từ điển

```
person[“city”] = “Canada”
```

13.2 Phương thức xử lý với từ điển Dictionaries

Thay đổi một phần tử trong từ điển

- Hoặc bạn có thể sử dụng phương thức **update()**

```
person.update({“city”: “Canada”})
```

Thêm một phần tử vào từ điển

```
person[“email”] = “jonh@gmail.com”
```

Xóa một phần tử trong từ điển

```
person.pop(“email”)
```

Hoặc

```
del person[“email”]
```

9.2 Phương thức xử lý với từ điển Dictionaries

Thay đổi một phần tử trong từ điển

- Xóa phần tử ở cuối từ điển với phương thức **popitem()**

```
person.popitem()
```

Xoá luôn một từ điển

```
del person
```

Xóa tất cả phần tử trong từ điển, trả về từ điển rỗng

```
person.clear()
```

9.2 Phương thức xử lý với từ điển Dictionaries

Vòng lặp for với kiểu dữ liệu từ điển

- Lặp qua từ điển nó trả lại **key** sau mỗi lần lặp

```
for x in person:
   print(x) # Kết quả: name age city
```

- Hoặc bạn dùng phương thức keys()

```
for x in person.keys():
   print(x) # Kết quả: name age city
```

- Qua đó để lấy giá trị của các phần tử sau mỗi lần lặp bạn code như sau:

```
for x in person:
   print(person[x]) # Kết quả: Jonh 30 New york
```

9.2 Phương thức xử lý với từ điển Dictionaries

Vòng lặp for với kiểu dữ liệu từ điển

- Hoặc sử dụng phương thức **values()**

```
for x in person.values():
   print(x) # Kết quả: Jonh 30 New york
```

- Lặp qua lấy cả key và value với phương thức **items()**

```
for x, y in person.items():
  print(x, y)
```

9.2 Phương thức xử lý với từ điển Dictionaries

Sao chép (copy) một kiểu dữ liệu từ điển

- Sử dụng phương thức **copy()**

```
mydict = person.copy()
print(mydict)
```

- Hoặc một cách khác với phương thức dict()

```
mydict = dict(person)
print(mydict)
```

Tổng kết lại bài

1. Nắm được thế nào là kiểu dữ liệu từ điển Dictionaries

2. Nắm được các phương thức xử lý với từ điển Dictionaries