HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts and Telecommunications Institute of Technology

CHƯƠNG 5: TỪ ĐIỂN

VŨ HOÀI THƯ
CNTT1. PTIT

NỘI DUNG

- Ví dụ đơn giản về từ điển
- Làm việc với từ điển
- Lặp qua toàn bộ từ điển
- Nesting

Ví dụ

- Một game với các nhân vật có đặc tính như màu sắc và điểm khác nhau khi chúng bị hạ gục.
- Dưới đây là cách đơn giản để lưu trữ thông tin của nhân vật bằng từ điển:

```
alien = {'color':'green', 'point':5}

print(alien)
```

```
{'color': 'green', 'point': 5}
```

ĐỊNH NGHĨA TỪ ĐIỂN

- Từ điển (dictionary) là một kiểu dữ liệu lưu trữ một cặp chứa khóa và giá trị (key-value)
- Các phần tử được đặt trong cặp dấu {} và được phân tách nhau bởi dấu phẩy (,)

dictionary_name = {key1: value1, key2: value2, ..., keyN: valueN}

```
population = {'Hà Nội': 900, 'TP Hồ Chí Minh': 1000, 'Ninh Bình': 370, 'Hà Nam': 250}

print(population)
```

```
{'Hà Nội': 900, 'TP Hồ Chí Minh': 1000, 'Ninh Bình': 370, 'Hà Nam': 250}
```

ĐỊNH NGHĨA TỪ ĐIỂN

- Lưu ý:
  - ✓ Mỗi cặp key-value được xem là một phần tử
  - ✓ Các phần tử đều phải có key và đã khai báo thì không đổi được tên
  - ✓ Key phải là giá trị không thể thay đổi: Số, Chuỗi, Tuple
  - ✓ Key phải là duy nhất, nếu không nó sẽ nhận giá trị của phần tử có key xuất hiện cuối cùng
  - ✓ Key có phân biệt chữ hoa thường
  - ✓ Value có thể là bất kỳ loại dữ liệu nào

LÀM VIỆC VỚI TỪ ĐIỂN

- Các phần tử trong dictionary được sắp xếp không theo một thứ tự nào cả, chính vì thế không thể sử dụng các cú pháp truy cập vào từng phần tử như đối với chuỗi và danh sách.
- Một số tác vụ làm việc với tử điển: thêm phần tử, xóa phần tử, thay đổi giá trị của phần tử, ...

TRUY CẬP VÀO CÁC PHẦN TỬ TRONG TỪ ĐIỂN

- Để truy cập vào các phần tử trong từ điển, sử dụng cú pháp sau:

dictionary_name[key]

- Trong đó:
  - ✓ *dictionary_name* là tên của dictionary
  - ✓ *key* là tên key của phần tử cần lấy ra trong dictionary

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20, 'address': 'Hà Nội'}

name = user['name']

age = user['age']

print(f'Xin chào khách hàng: {name} - {age} tuổi!')
```

```
Xin chào khách hàng: Nguyễn Văn A - 20 tuổi!
```

THÊM CẶP KEY-VALUE MỚI

- Để thêm một cặp new_key, new_value mới vào trong từ điển, sử dụng cú pháp sau:

dictionary_name[new_key] = new_value

- Trong đó:
  - ✓ *dictionary_name* là tên của dictionary
  - ✓ *new_key* là tên key của phần tử muốn thêm vào trong dictionary
  - ✓ *new_value*: giá trị của phần tử muốn thêm vào trong dictionary

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20, 'address': 'Hà Nội'}

print(user)

user['job'] = 'Sinh viên'

print(user)
```

```
{'name': 'Nguyễn Văn A', 'gender': 'Male', 'age': 20, 'address': 'Hà Nội'}
{'name': 'Nguyễn Văn A', 'gender': 'Male', 'age': 20, 'address': 'Hà Nội', 'job': 'Sinh viên'}
```

TỪ ĐIỂN RỖNG

- Thông thường, ta hay bắt đầu với từ điển rỗng, sau đó thêm các phần tử vào từ điển theo các bước dưới đây:

dictionary_name = {}

dictionary_name[key1] = value1

dictionary_name[key2] = value2

.....

dictionary_name[keyN] = valueN

```
user = {}
user['name'] = 'Nguyễn Văn A'
user['age'] = 20
user['gender'] = 'Male'
print(user)
```

```
{'name': 'Nguyễn Văn A', 'age': 20, 'gender': 'Male'}
```

THAY ĐỔI GIÁ TRỊ TRONG TỪ ĐIỂN

- Để thay đổi giá trị của phần tử trong dictionary, ta cần truy cập đến phần tử qua key và thay đổi giá trị (value) của nó.

dictionary[key] = new_value

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

print(f"User's age is {user['age']}")

user['age'] = 21

print(f"User's age is now {user['age']}")
```

```
User's age is 20
User's age is now 21
```

XÓA PHẦN TỬ TRONG TỪ ĐIỂN BẰNG HÀM pop() VÀ popitem()

- Hàm pop() xóa phần tử của dictionary với key được chỉ định

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

popped_value = user.pop('name')

print(user)

print(f'Popped value is: {popped_value}')
```

```
{'gender': 'Male', 'age': 20}
Popped value is: Nguyễn Văn A
```

- Hàm popitem() xóa phần tử cuối cùng của dictionary

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

popped_value = user.popitem()

print(user)

print(popped_value)
```

```
{'name': 'Nguyễn Văn A', 'gender': 'Male'}
('age', 20)
```

XÓA PHẦN TỬ TRONG TỪ ĐIỂN BẰNG LỆNH del VÀ HÀM clear()

- Xóa một phần tử trong từ điển sử dụng hàm del: *del dictionary_name[key]*

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

del user['name']

print(user)
```

```
{'gender': 'Male', 'age': 20}
```

- Xóa một dictionary sử dụng hàm del: *del dictionary_name*

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

del user
```

- Xóa tất cả các phần tử của dictionary sử dụng hàm clear(): *dictionary_name.clear()*

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

user.clear()

print(user)
```

→ {}

PHƯƠNG THỨC GET()

- Sử dụng key trong dấu ngoặc vuông ([ ]) để truy xuất giá trị value của phần tử trong từ điển sẽ xảy ra lỗi khi key không tồn tại.

```
1 user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
2
3 user['address']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_7740/1886389112.py in <module>
      1 user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
      2
----> 3 user['address']

KeyError: 'address'
```

- Phương thức get(): trả về giá trị một phần tử trong từ điển với một khóa cho trước, khóa không tồn tại sẽ trả về giá trị mặc định.
- Phương thức key() và value(): Trả về danh sách khóa và danh sách các giá trị của từ điển

PHƯƠNG THỨC GET()

❑ Phương thức get()

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

name = user.get('name', 'No address value assign')

print(name)
```

→ 'Nguyễn Văn A'

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

address = user.get('address')

print(address)
```

→ None

❑ Phương thức key() và value()

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
list_keys = user.keys()
list_values = user.values()
print(list_keys)
print(list_values)
```

→

```
dict_keys(['name', 'gender', 'age'])
dict_values(['Nguyễn Văn A', 'Male', 20])
```

BÀI TẬP

- Bài tập 1: Viết chương trình thực hiện các yêu cầu sau:
  - ✓ Tạo một từ điển với key – value tương ứng là tên, MSSV và lớp tương ứng.
  - ✓ Thêm vào từ điển vừa tạo key “Môn học” với giá trị tương ứng là “Lập trình Python”
- Bài tập 2: Cho một danh sách gồm các phần tử bất kỳ. Sử dụng dictionary, đếm số lần xuất hiện của các phần tử trong danh sách.

Ví dụ: list = [‘a’, 1, 1, ‘b’, 4, 4, ‘a’, 2] => result = {‘a’ : 2, 1 : 2, ‘b’ : 1, 4 : 2, 2 : 1}

VÒNG LẶP VỚI TỪ ĐIỂN

- Vòng lặp qua tất cả các cặp key-value:

```
for key, value in dictionar_name.items():

    do something with key-value
```

```
user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}

for key, value in user.items():
    print(f'Key : {key} -  Value: {value}')
```

```
Key : name -  Value: Nguyễn Văn A
Key : gender -  Value: Male
Key : age -  Value: 20
```

VÒNG LẶP VỚI TỪ ĐIỂN

❑ Vòng lặp qua tất cả các key:

```
for key in dictionary_name.keys():

    do something with key
```

```
1  user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
2
3  for key in user.keys():
4      print(key)
```

```
name
gender
age
```

VÒNG LẶP VỚI TỪ ĐIỂN

- Vòng lặp qua tất cả các key được sắp xếp giá trị:

```
for key in sorted(dictionary_name.keys()):

    do something with key
```

```
1  user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
2
3  for key in sorted(user.keys()):
4      print(key)
```

```
age
gender
name
```

VÒNG LẶP VỚI TỪ ĐIỂN

- Vòng lặp qua tất cả các value:

```
for value in dictionary_name.values():

    do something with value
```

```
1  user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
2
3  for value in user.values():
4      print(value)
```

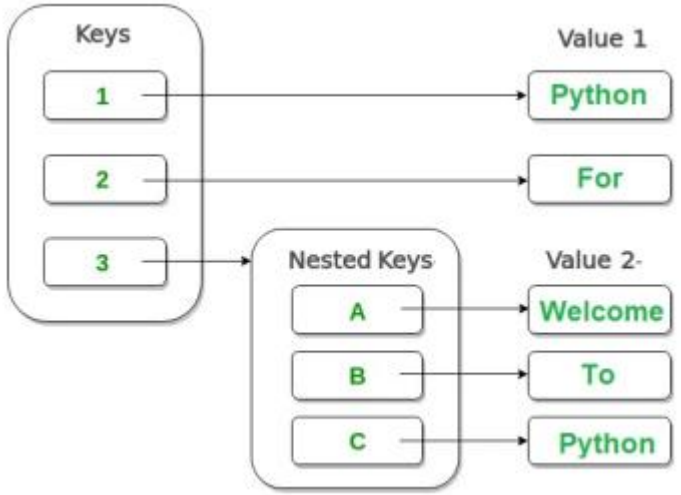
```
Nguyễn Văn A
Male
20
```

NESTING – DANH SÁCH CÁC TỪ ĐIỂN

- Lưu trữ nhiều từ điển trong một danh sách hoặc lưu trữ một từ điển trong một từ điển khác được gọi là nesting (hay từ điển lồng nhau)

```
1 Dict = {1: 'Python', 2: 'For', 3: {'A': 'Welcome', 'B':'To', 'C':'Python'}}
2
3 print(Dict)
```

```
{1: 'Python', 2: 'For', 3: {'A': 'Welcome', 'B': 'To', 'C': 'Python'}}
```



CÁC HÀM VÀ PHƯƠNG THỨC DỰNG SẴN CHO TỪ ĐIỂN

- Hàm tìm độ dài (số lượng item) của một dictionary:

len(dictionary_name)

```
1  user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
2
3  len(user)
```

```
3
```

- Hàm trả về bản sao của một dictionary:

dictionary_name.copy()

```
1  user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
2
3  user_copy = user.copy()
4
5  print(user_copy)
```

```
{'name': 'Nguyễn Văn A', 'gender': 'Male', 'age': 20}
```

CÁC HÀM VÀ PHƯƠNG THỨC DỰNG SẴN CHO TỪ ĐIỂN

- Hàm cập nhật từ điển update():

dictionary_name.update({key:value})

```
1  user = {'name': 'Nguyễn Văn A', 'gender':'Male', 'age': 20}
2
3  user.update({'address' : 'Hà Nội'})
4
5  print(user)
```

```
{'name': 'Nguyễn Văn A', 'gender': 'Male', 'age': 20, 'address': 'Hà Nội'}
```

```
1  user = {'name': 'Nguyễn Văn A'}
2
3  info = {'gender':'Male', 'age': 20, 'address':'Hà Nội'}
4
5  user.update(info)
6
7  print(user)
```

```
{'name': 'Nguyễn Văn A', 'gender': 'Male', 'age': 20, 'address': 'Hà Nội'}
```

CÁC HÀM VÀ PHƯƠNG THỨC DỰNG SẴN CHO TỪ ĐIỂN

- Phương thức fromkeys(): Trả về một từ điển với các key và value được chỉ định.

```
1  keys = {'a', 'e', 'i', 'o', 'u' }
2  value = 'vowel'
3
4  vowels = dict.fromkeys(keys, value)
5  print(vowels)
```

```
{'e': 'vowel', 'u': 'vowel', 'o': 'vowel', 'a': 'vowel', 'i': 'vowel'}
```

```
1  keys = {'a', 'e', 'i', 'o', 'u' }
2  # value = 'vowel'
3
4  vowels = dict.fromkeys(keys)
5  print(vowels)
```

```
{'e': None, 'u': None, 'o': None, 'a': None, 'i': None}
```

BÀI TẬP

- Bài tập 3: Viết chương trình Python in ra một từ điển trong đó key là các số từ 1 đến 15 và value là bình phương của các khóa.

- Bài tập 4: Viết chương trình tính tổng của các phần tử có value là số trong một từ điển.

BÀI TẬP

- Bài tập 5: Viết chương trình Python kết hợp hai từ điển bằng cách tính tổng giá trị của các phần tử có cùng key.

  Ví dụ: d1 = {'a': 100, 'b': 200, 'c':300}

  d2 = {'a': 300, 'b': 200, 'd':400}

  => Output: {'a': 400, 'b': 400, 'c': 300 , 'd': 400}

- Bài tập 6: Viết chương trình đếm số lượng các phần tử có giá trị là một danh sách trong một từ điển bất kỳ.

KẾT CHƯƠNG

- Dictionary là kiểu dữ liệu lưu trữ một cặp giá trị key-value
- Truy cập giá trị, thêm, sửa, xóa các phần tử trong từ điển
- Vòng lặp trong từ điển
- Khái niệm nesting trong từ điển