มหาวิทยาลัยราชภัฏนครปฐม
Nakhon Pathom Rajabhat University

บทที่ 8 ข้อมูลประเภทเซ็ท (Set) และ ดิกชันนารี (Dictionary) ของภาษา python

รายวิชา 3602801 การเขียนโปรแกรมคอมพิวเตอร์ทางธุรกิจ

ผศ.ดร.เดช ธรรมศิริ

Set (เซ็ต)

- เป็นออบเจ็คที่ใช้สำหรับเก็บข้อมูลที่ไม่ซ้ำกัน ในการเขียนโปรแกรมเราสามารถใช้ Set สำหรับหาค่าที่ไม่ซ้ำกันภายในลิสต์ ตรวจสอบข้อมูลที่มีอยู่แล้ว หรือการดำเนินการทางคณิตศาสตร์เกี่ยวกับ Set ได้ เช่น Intersection, Union, Difference และ Symmetric difference

- ตัวอย่างของ เซ็ต ได้แก่

- `myset = {"apple", "banana", "cherry"}`

การสร้างเซ็ต

- ```
  thisset = {"apple", "banana", "cherry"}
  print(thisset)
  ```

```
{'cherry', 'apple', 'banana'}
```
```

คุณสมบัติของข้อมูลประเภทเซ็ต

- จะไม่เรียงลำดับ
- จะไม่สามารถเปลี่ยนค่าได้
- ไม่สามารถมีค่าที่ซ้ำกันได้

ตัวอย่าง เก็บค่าซ้ำในเซ็ต ผลลัพธ์จะปรับให้เหลือเก็บแค่ค่าเดียวที่ไม่ซ้ำกัน

- ```
  thisset = {"apple", "banana", "cherry", "apple"}

  print(thisset)
  ```

```
{'cherry', 'apple', 'banana'}
```
```

การเข้าถึงข้อมูลในเซ็ต

- thisset = {"apple", "banana", "cherry"}

```
for x in thisset:
  print(x)
```

```
cherry
apple
banana
```

- หรืออาจใช้ รูปแบบนี้เพื่อเช็คข้อมูลที่มีอยู่ในเซ็ต
- thisset = {"apple", "banana", "cherry"}

```
print("banana" in thisset)
```

```
True
```

การเปลี่ยนแปลงข้อมูลในเซ็ต

- เซ็ตจะไม่สามารถเปลี่ยนแปลงข้อมูลได้ แต่สามารถเพิ่มข้อมูลใหม่เข้าไปได้

การเพิ่มข้อมูลในเซ็ตด้วย add() method

- ```
  thisset = {"apple", "banana", "cherry"}

  thisset.add("orange")

  print(thisset)
  ```

```
{'cherry', 'orange', 'apple', 'banana'}
```
```

ทำการ add item จากอีกเซ็ต ในข้อมูลเซ็ตปัจจุบัน ด้วย update() method

- ```
  thisset = {"apple", "banana", "cherry"}
  tropical = {"pineapple", "mango", "papaya"}

  thisset.update(tropical)

  print(thisset)
  ```

```
{'cherry', 'mango', 'pineapple', 'papaya', 'banana', 'apple'}
```
```

การเพิ่มข้อมูลเซ็ตไปยังข้อมูลประเภทอื่นด้วย update() method

- ```
  thisset = {"apple", "banana", "cherry"}
  mylist = ["kiwi", "orange"]

  thisset.update(mylist)

  print(thisset)
  ```

```
{'orange', 'kiwi', 'cherry', 'banana', 'apple'}
```
```

การนำข้อมูลออกจากเซ็ต โดยใช้ remove() หรือ discard()

- ```
  thisset = {"apple", "banana", "cherry"}

  thisset.remove("banana")

  print(thisset)
  ```

```
{'cherry', 'apple'}
```
```

ตัวอย่างทำการลบ banana ออกจากเซ็ตด้วย discard() method:

- ```
  thisset = {"apple", "banana", "cherry"}

  thisset.discard("banana")

  print(thisset)
  ```

```
{'cherry', 'apple'}
```
```

เอาข้อมูลลำดับท้ายสุดออกจากเซ็ตโดยใช้ pop() method

- ```
  thisset = {"apple", "banana", "cherry"}

  x = thisset.pop()

  print(x)

  print(thisset)
  ```

```
cherry
{'apple', 'banana'}
```
```

เอาข้อมูลทั้งหมดออกจากเซ็ต ด้วย clear() method

- ```
  thisset = {"apple", "banana", "cherry"}

  thisset.clear()

  print(thisset)
  ```

```
set()
```
```

การใช้ del keyword ในการลบเซ็ต

- thisset = {"apple", "banana", "cherry"}

```
del thisset

print(thisset)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-12-01aebabb8c34> in <module>()
      1 thisset = {"apple","banana","cherry"}
      2 del thisset
----> 3 print(thisset)

NameError: name 'thisset' is not defined

SEARCH STACK OVERFLOW
```

การดึงข้อมูลจากเซ็ตออกมาใช้งานด้วย for loop

- thisset = {"apple", "banana", "cherry"}

```
for x in thisset:
  print(x)
```

```
apple
cherry
banana
```

การเชื่อม 2 เซ็ตเข้าด้วยกัน ด้วยคำสั่ง union() method

- ```
  set1 = {"a", "b" , "c"}
  set2 = {1, 2, 3}

  set3 = set1.union(set2)
  print(set3)
  ```

```
{1, 'b', 2, 'a', 3, 'c'}
```
```

การเชื่อม 2 เซ็ตเข้าด้วยกัน ด้วยคำสั่ง update() method

- ```
  set1 = {"a", "b" , "c"}
  set2 = {1, 2, 3}

  set1.update(set2)
  print(set1)
  ```

```
{1, 'b', 2, 'a', 3, 'c'}
```
```

ข้อมูลประเภทดิชันนารี (Dictionaries)

- คือประเภทข้อมูลที่เก็บข้อมูลในรูปแบบคู่ของ Key และ Value โดยที่ Key ใช้สำหรับเป็น Index ในการเข้าถึงข้อมูลและ Value เป็นค่าข้อมูลที่สอดคล้องกับ Key ของมัน การเข้าถึงข้อมูลใน Dictionary นั้นรวดเร็ว เพราะว่าข้อมูลได้ถูกทำ Index ไว้อัตโนมัติโดยใช้ Key นอกจากนี้ Dictionary ยังมีเมธอดและฟังก์ชันอำนวยความสะดวกสำหรับการทำงานทั่วไป
- ตัวอย่างข้อมูล
- ```
  thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
  }
  ```
  ```

การสร้างข้อมูลประเภทดิกชันนารี

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(thisdict)
```

Key

Values

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

ข้อมูลในดิชันนารี

- ข้อมูลจะเรียงลำดับ เปลี่ยนแปลงได้ แต่ไม่อนุญาตให้มีค่าที่ซ้ำกัน ตัวอย่างเช่นต้องการดึงข้อมูล brand ซึ่งเป็นค่าที่เก็บอยู่ในตัวแปรดิชันนารี
-
```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(thisdict["brand"])
```

```
Ford
```

การเรียงลำดับในข้อมูลประเภทดิชันนารี

- ในภาษา python ตั้งแต่ version 3.7 ขึ้นไป ข้อมูลจะถูกเรียงลำดับ
- แต่หากในภาษา python ตั้งแต่ version 3.6 ลงมา ข้อมูลจะไม่มีการเรียงลำดับ

ในดิกชันนารีจะไม่อนุญาตให้มีข้อมูลที่เป็น key ซ้ำ ดังตัวอย่าง

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964,
  "year": 2020
}
print(thisdict)
```

```
thisdict ={"brand":"Ford","model":"Mustang","year":1964,"year":2021}
print(thisdict)
```

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 2021}
```

การเข้าถึงข้อมูลของดิกชันนารี จะเข้าถึงโดยผ่าน key

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
x = thisdict["model"]
```

```
Mustang
```

สามารถเข้าถึงโดยใช้ get()

- ```
  x = thisdict.get("model")
  print(x)
  ```

```
Mustang
```
```

แสดงข้อมูล key ทั้งหมด ที่มีอยู่ในดิชันนารี ด้วย keys() method

- ```
  x = thisdict.keys()
  print(x)
  ```

```
dict_keys(['brand', 'model', 'year'])
```
```

เพิ่ม key และ values ใหม่ สำหรับค่าใน ดิชันนารี

```
car = {
"brand": "Ford",
"model": "Mustang",
"year": 1964
}

x = car.keys()

print(x) #before the change

car["color"] = "white"

print(x) #after the change
```

```
dict_keys(['brand', 'model', 'year'])
dict_keys(['brand', 'model', 'year', 'color'])
```

การเปลี่ยนแปลงค่าใน ดิกชันนารี

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict["year"] = 2018
print(thisdict["year"])
```

```
2018
```

การเพิ่มkey และข้อมูลเข้าสู่ ดิกชันนารี

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict["color"] = "red"
print(thisdict)
```

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964, 'color': 'red'}
```

การแก้ข้อมูลในดิกชันนารี ด้วย update() method:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.update({"color": "red"})
```

```
print(thisdict)
```

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964, 'color': 'red'}
```

การนำข้อมูลออกจากดิชันนารี ด้วย pop() method

- โดยสามารถระบุ key ที่ต้องการนำข้อมูลออก ดังตัวอย่าง

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.pop("model")
print(thisdict)
```

```
{'brand': 'Ford', 'year': 1964}
```

การใช้ ดึงค่า key ออกมา ด้วย for loop

- ```
  for x in thisdict:
      print(x)
  ```

```
brand
year
```
```

การดึง ค่า ใน ดิกชันนารี ออกมา ด้วย for loop

- ```
  for x in thisdict.values():
    print(x)
  ```

```
Ford
1964
```
```

สามารถใช้ keys() method ในการแสดงค่า key ใน ดิกชันนารี

- ```
  for x in thisdict.keys():
    print(x)
  ```

```
brand
year
```
```

สามารถดึงทั้ง key และ ค่าที่เก็บไว้ ออกมาพร้อมกันด้วย items() method:

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

```
brand Ford
year 1964
```
```

การทำสำเนาข้อมูลในดิกชันนารี ด้วย copy() method:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
mydict = thisdict.copy()
print(mydict)
```

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

สามารถเก็บข้อมูลดิกชันนารี หลาย ๆข้อมูลรวมกัน

```
myfamily = {
  "child1" : {
    "name" : "Emil",
    "year" : 2004
  },
  "child2" : {
    "name" : "Tobias",
    "year" : 2007
  },
  "child3" : {
    "name" : "Linus",
    "year" : 2011
  }
}
```

```
{'child1': {'name': 'Emil', 'year': 2004}, 'child2': {'name': 'Tobias', 'year': 2007}, 'child3'
```

หรือใช้วิธีนี้เก็บข้อมูลร่วมกันในดิกชันนารีได้เช่นกัน

```
child1 = {
  "name" : "Emil",
  "year" : 2004
}
child2 = {
  "name" : "Tobias",
  "year" : 2007
}
child3 = {
  "name" : "Linus",
  "year" : 2011
}

myfamily = {
  "child1" : child1,
  "child2" : child2,
  "child3" : child3
}
```

```
{'child1': {'name': 'Emil', 'year': 2004}, 'child2': {'name': 'Tobias', 'year': 2007}, 'child3'
```

ขอบคุณครับ