Mahidol University
Wisdom of the Land

Mahidol University

Lecture 10: Dictionary, Tuple, Set

Lecturers:

Aj. Jidapa Kraisangka
Aj. Akara Supratak
Aj. Tipajin Thaipisutikul

Recap: List

- โครงสร้างข้อมูลชนิดหนึ่ง (built-in data type) ที่ใช้เก็บข้อมูลแบบลำดับ (sequence)
- สามารถใช้ List เพื่อเก็บข้อมูลจำนวนมาก และหลากหลายประเภท (เช่น integer, string, object) ในเวลาเดียวกัน
- ข้อมูลใน list สามารถถูกแก้ไขเปลี่ยนแปลงได้ (mutable) และ list สามารถเก็บข้อมูลซ้ำ (duplicate value) ได้
- ความยาว (length) ของ list สามารถเปลี่ยนแปลง ได้โดยไม่ต้องมีการประกาศล่วงหน้า

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Recap: Creating a 1D List

- การประกาศ list นั้นข้อมูลของ list จะอยู่ภายในเครื่องหมาย “[ ]”
- คั่นสมาชิก (item) แต่ละตัวด้วยเครื่องหมาย “ , ”

การสร้าง list เปล่าที่ยังไม่มีข้อมูล

```
# Create an empty list
list1 = []
#or
list1 = list()
```

```
# Create a list of integers
list1 = [1, 2, 3, 4, 5]

# Create a list of characters
list2 = ['a', 'b', 'c', 'd', 'a']

# Create a list with mixed type of item
list3 = [1, 2, 'b', 'c', list1]
```

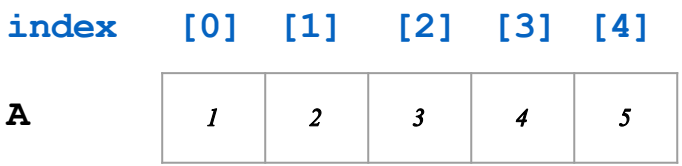
Item ใน list สามารถซ้ำกันได้

Item ใน list สามารถมีได้หลายประเภท

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

RECAP: 1D List

- List: โครงสร้างข้อมูลชนิดหนึ่ง (built-in data type) ที่ใช้เก็บข้อมูลแบบลำดับ (sequence)
- การประกาศ list: A = [] หรือ A = [1, 2, 3, 4, 5]
- List ใช้ index สำหรับการเข้าถึงข้อมูล แต่ละตัว (ข้อมูลแรก index **0**)




```
A[0]    # 1
A[1]    # 2
A[4]    # 5
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Recap: Accessing Range of List Item

•การเข้าถึง range ของข้อมูลใน list สามารถทำได้ โดยการระบุ index เริ่มต้นและ index สุดท้าย (ไม่รวม) ของ range

```
list[start:end]
```

```
list1 = ['a', 'b', 'c', 'd', 'e', 1, 2, 3]
```

```
print(list1[0:2])   → ['a', 'b']
print(list1[3:6])   → ['d', 'e', 1]
print(list1[5:8])   → [1, 2, 3]
```

```
print(list1[:2])    → ['a', 'b']
print(list1[5:])    → [1, 2, 3]
```

Dictionary

- โครงสร้างข้อมูลชนิดหนึ่ง (built-in data type) ที่ใช้เก็บข้อมูลแบบคู่ของ key-value
- ข้อมูลใน dictionary สามารถเข้าถึงได้โดยใช้ key ซึ่งต่างกับข้อมูลใน list ที่ใช้ index ในการระบุถึงค่าใน list
- dictionary สามารถถูกแก้ไขเปลี่ยนแปลงได้ (mutable)
- ข้อมูลใน dictionary สามารถเปลี่ยนแปลง โดยการเพิ่ม ลด เปลี่ยนค่า จาก dictionary methods

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

ทำไมต้อง Dictionary?

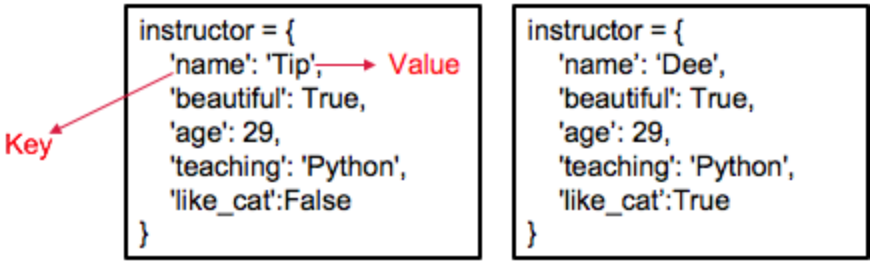
List ที่เก็บข้อมูลอาจารย์: ['Tip', True, 29, 'Python', False]

- จะเห็นว่าข้อมูลใน List จะสามารถเข้าถึงได้โดยใช้ index ซึ่งไม่ได้บอกความหมายของข้อมูลแต่ละตัวอย่างชัดเจน
- ผู้ใช้ต้องจำเอาเองว่า ข้อมูลในแต่ละตำแหน่งหมายถึงอะไร เช่น ข้อมูลตำแหน่งที่แรก index=0 จะเก็บชื่อ
- หากผู้ใช้ต้องการ เก็บข้อมูลอาจารย์มากกว่า 1 คน ผู้ใช้อาจใช้ nested list ได้เช่น

[['Tip', True, 29, 'Python', False], ['Dee', True, 29, 'Python', True]]

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Dictionary ที่เก็บข้อมูลอาจารย์:



- โครงสร้างข้อมูลที่ประกอบด้วย key และ value pairs ซึ่งผู้ใช้สามารถตั้งชื่อ key ตามความเหมาะสมได้เอง
- เราสามารถใช้ keys ในการอธิบายความหมายของข้อมูลได้ และ ค่าของข้อมูลนั้นจะถูกเก็บอยู่ใน values
- เราอ้างอิงข้อมูลใน Dictionary โดยใช้ key (ต่างกับ list ที่ใช้ index)

การสร้าง dictionary (1)

- การประกาศ dict นั้นข้อมูลของ dict จะอยู่ภายในเครื่องหมาย “**{ }**”
- คั่นสมาชิก (item) แต่ละตัวด้วยเครื่องหมาย “ **,** ”
- คั่น key และ value ด้วย “:”

การสร้าง dict เปล่าที่ยังไม่มีข้อมูล

```
# Create an empty dict
dict1 = {}
#or
dict1 = dict()
```

```
#1. declear + initilize
instructor = {
    'name': 'Tip',
    'beautiful': True,
    'age': 29,
    'teaching': 'Python',
    'like_cat':False
}
```

→ Item ใน dict สามารถมีได้หลายประเภท (int, float, boolean, string)

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

การสร้าง dictionary (2)

- การประกาศ dict นั้นข้อมูลของ dict จะอยู่ภายในเครื่องหมาย “**dict()**”
- คั่นสมาชิก (item) แต่ละตัวด้วยเครื่องหมาย “ , ”
- คั่น key และ value ด้วย “=”

การสร้าง dict เปล่าที่ยังไม่มีข้อมูล

```
# Create an empty dict
dict1 = {}
#or
dict1 = dict()
```

Item ใน dict สามารถมีได้หลายประเภท (int, float, boolean, string)

```
#2. declear + initilize
instructor2 = dict(name = 'Dee',
beautiful= True,
age =28,
teaching = 'Python',
like_cat = True)
```

การสร้าง dictionary (3)

การสร้าง dict เปล่าที่ยังไม่มีข้อมูล

```
# Create an empty dict
dict1 = {}
#or
dict1 = dict()
```

POSITION: INSTRUCTOR

Lect. Dr. Akara Supratak

E-mail: akara.sup@mahidol.edu

CV Scopus Google

```
#3. declear + initialize later
instructor3 = dict()
instructor3['position'] = 'instructor'
instructor3['name'] = 'akara'
instructor3['surname'] = 'supratak'
instructor3['email'] = 'akara.sup'
print(instructor3)
```

```
{'position': 'instructor', 'name': 'akara',
'surname': 'supratak', 'email': 'akara.sup'}
```

Key ควรจะไม่ซ้ำกัน

```
#4. Loop to create dict
instructors = dict() #or instructor = {}
for i in range(n): #n is the total number of key-value pairs
        #ask a user to input key & value
        instructors[key] = value
```

การเข้าถึงข้อมูลใน Dictionary

- Dictionary ใช้ key สำหรับการเข้าถึงข้อมูล แต่ละตัว

Syntax ชื่อdict[‘ชื่อkey’]

```
instructor3 = dict()
instructor3['position'] = 'instructor'
instructor3['name'] = 'akara'
instructor3['surname'] = 'supratak'
instructor3['email'] = 'akara.sup'
```

```
Instructor3['name']     # 'akara'
Instructor3['surname']  # 'supratak'
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

- เราสามารถใช้ for...in...loop ในการเข้าถึงข้อมูลใน dictionary ได้ร่วมกับ 3 methods (.values(), .keys(), .items()) ดังนี้

```
instructor = {
    'name': 'Tip',
    'beautiful': True,
    'age': 29,
    'teaching': 'Python',
    'like_cat':False
}
#if we want to print all the values
print(instructor['name'])
print(instructor['beautiful'])
print(instructor['age'])
print(instructor['teaching'])
print(instructor['like_cat'])
```

```
for value in instructor.values():
    print(value)

for key in instructor.keys():
    print(key)

for key, value in instructor.items():
    print(key, value)
```

```
Tip
True
29
Python
False
```

```
name
beautiful
age
teaching
like_cat
```

```
name Tip
beautiful True
age 29
teaching Python
like_cat False
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

- จงหาจำนวนยอดบริจาครวมจากข้อมูล dictionary ต่อไปนี้ โดยใช้ loop ในการคำนวณ
- จงหารายชื่อผู้บริจาคจากข้อมูล dictionary ต่อไปนี้ โดยใช้ loop ในการคำนวณ
- จงหาชื่อของผู้บริจาคเงินมากที่สุดจากข้อมูล dictionary ต่อไปนี้ โดยใช้ loop ในการคำนวณ
- จงหาชื่อของผู้บริจาคเงินน้อยที่สุดจากข้อมูล dictionary ต่อไปนี้ โดยใช้ loop ในการคำนวณ

```
donations = dict(sam=25.0,
lena=88.99,
chuck=13.0,
linus=99.5,
stan=150.0,
lisa=50.25,
harrison=10.0)
```

การเข้าถึงข้อมูลใน Dictionary โดยใช้ลูป


```
donations = dict(sam=25.0,
lena=88.99,
chuck=13.0,
linus=99.5,
stan=150.0,
lisa=50.25,
harrison=10.0)
```

จงหารายชื่อผู้บริจาคจากข้อมูล dictionary ต่อไปนี้ โดยใช้ loop ในการคำนวณ:

```
total_donations = 0
for donation in donations.values():
        total_donations += donation
print(total_donations)
```

```
436.74
```

จงหารายชื่อผู้บริจาคจากข้อมูล dictionary ต่อไปนี้ โดยใช้ loop ในการคำนวณ:

```
for key in donations.keys():
        print(key, end=' ')
```

```
sam lena chuck linus stan lisa harrison
```

การเข้าถึงข้อมูลใน Dictionary โดยใช้ลูป


```
donations = dict(sam=25.0,
lena=88.99,
chuck=13.0,
linus=99.5,
stan=150.0,
lisa=50.25,
harrison=10.0)
```

จงหาชื่อของผู้บริจาคเงินมาก/น้อยที่สุดจากข้อมูล dictionary โดยใช้ loop ในการคำนวณ

```
min_value = min(donations.values())
max_value = max(donations.values())
print("minvalue: "+str(min_value))
print("maxvalue: "+str(max_value))
for key,value in donations.items():
    if value == min_value:
            print('name (min): '+ key)
    if value == max_value:
            print('name (max): '+ key)
```

```
minvalue: 10.0
maxvalue: 150.0

name (max): stan
name (min): harrison
```


Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

การใช้ In กับ Dictionaries

Syntax ‘คำที่ต้องการหา’ in ชื่อdict

โดยปกติ คำที่ต้องการหา จะไปค้นหาใน Key
ถ้ามีคำนั้นๆอยู่ จะ return true ถ้าไม่มี จะ return false

```
instructor = {
    'name': 'Tip',
    'beautiful': True,
    'age': 29,
    'teaching': 'Python',
    'like_cat':False
}
```

```
print(instructor['address'])
#Does a dictionary have a key?
print('name' in instructor)
print('status' in instructor)
#Does a dictionary have a value?
print(instructor.values())
print('Tip' in instructor.values())
print('Tip' in instructor.keys())
print(5 in instructor.values())
print('name' in instructor.values())
print('name' in instructor.keys())
print('name' in instructor)
```

```
#KeyError: 'address'
#Does a dictionary have a key?
True
False
#Does a dictionary have a value?
dict_values(['Tip', True, 29, 'Python', False])
True
False
False
False
True
True
```

Dictionary Methods: `clear()`

- **`clear()`**: ลบข้อมูลทั้งหมดจาก dict

Syntax `dict.clear()`

```
d = dict(a=1, b=2, c=3)
print(d)
d.clear()
print(d)
```

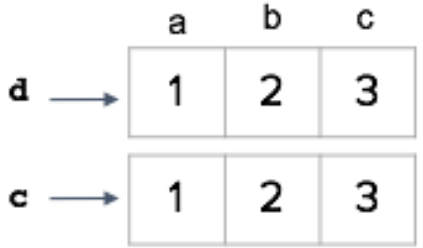
```
{'a': 1, 'b': 2, 'c': 3}
{}
```

Dictionary Methods: copy()

- ใช้ในการคัดลอก (shallow copy) dict ทั้งหมด
- การแก้ไข dict ใหม่จะไม่กระทบกับ dict ต้นแบบที่ถูกคัดลอก

Syntax `dict.copy()`

```
d = dict(a=1, b=2, c=3)
c =d.copy()
print(c)
c['a'] = 5
print(c)
print(d)
```




```
c=  {'a': 1, 'b': 2, 'c': 3}
c=  {'a': 5, 'b': 2, 'c': 3}
d=  {'a': 1, 'b': 2, 'c': 3}
```

Dictionary Methods : fromkeys()

fromkeys() : สร้างข้อมูล dict จาก key-value pairs โดยมี ‘,’ ใช้คั่น

Syntax `dict.fromkeys('key','value')`

```
{}.fromkeys("a","b")
```
→ `{'a':'b'}`

```
dict.fromkeys('a','b')
```
→ `{'a':'b'}`

Dictionary Methods : fromkeys()

fromkeys() : สร้างข้อมูล dict จาก key-value pairs โดยมี ‘,’ ใช้คั่น

Syntax `dict.fromkeys(['key1','key2'],'value')`

```
{}.fromkeys(['email'],'unknown')
print({}.fromkeys('a',[1,2,3,4,5]))

print({}.fromkeys(['email1','email2'],'unknown'))
print(dict.fromkeys(['email1','email2'],'unknown'))
```

```
{'email':'unknown'}
{'a': [1, 2, 3, 4, 5]}
{'email1': 'unknown', 'email2': 'unknown'}
```

Dictionary Methods : fromkeys()

fromkeys() : สร้างข้อมูล dict จาก key-value pairs โดยมี ‘,’ ใช้คั่น

Syntax `dict.fromkeys(['key1','key2'],'value')`

```
print({}.fromkeys('phone','unknown'))
```

```
{'p': 'unknown',
'h': 'unknown',
'o': 'unknown',
'n': 'unknown',
'e': 'unknown'}
```

```
print({}.fromkeys(['phone'],'unknown'))
```

```
{'phone': 'unknown'}
```

```
print({}.fromkeys(range(1,10),'unknown'))
```

```
{1: 'unknown', 2: 'unknown', 3: 'unknown', 4: 'unknown', 5:
'unknown', 6: 'unknown', 7: 'unknown', 8: 'unknown', 9: 'unknown'}
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Dictionary Methods : fromkeys()

หา output ของ code ต่อไปนี้

```
new_user = {}.fromkeys(['name', 'score', 'email',
'gender','phone'],'Unknown')
print(new_user)
```

```
{'name': 'Unknown', 'score': 'Unknown', 'email':
'Unknown', 'gender': 'Unknown', 'phone': 'Unknown'}
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Dictionary Methods : get()

get() : การเข้าถึงค่า value จาก key

Syntax `dict.get(key)`

หา output ของ code ต่อไปนี้

```
d = dict(a=1, b=2, c=3)
print(d['a'])
print(d.get('a'))
print(d['no_key'])
print(d.get('no_key'))
d['a'] = 5  #update values
print(d)
```

Dictionary Methods : pop ()

pop () : เอาข้อมูลออกจาก dict โดยใช้ key ในการค้นหาข้อมูลเพื่อลบออก

Syntax `dict.pop(key)`

```
d = dict(a=1,b=2,c=3)
d.pop()
d.pop('a')
print(d)

d.pop('e')
```

```
TypeError: pop expected at least 1
argument, got 0.
```

```
{'b': 2, 'c': 3}
```

```
KeyError
```

Dictionary Methods : Update()

update() : update ค่า keys and values in a dictionary ด้วยชุดข้อมูลของคู่ key value pairs

Syntax `dict.update(another_dict)`

```
first = dict(a=1, b=2, c=3, d=4,
e=5)
second = {} #empty dict
second.update(first)
print(second)
```

```
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5}
```

Dictionary Methods : Update()


```
instructor = {
    'name': Akara',
    'age': 30,
    'teaching': 'Python'
}
person = {'city': 'Thailand'}
person.update(instructor)
print(person)
```

```
{'city': 'Thailand', 'name': Akara',
'age': 30, 'teaching': 'Python'}
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

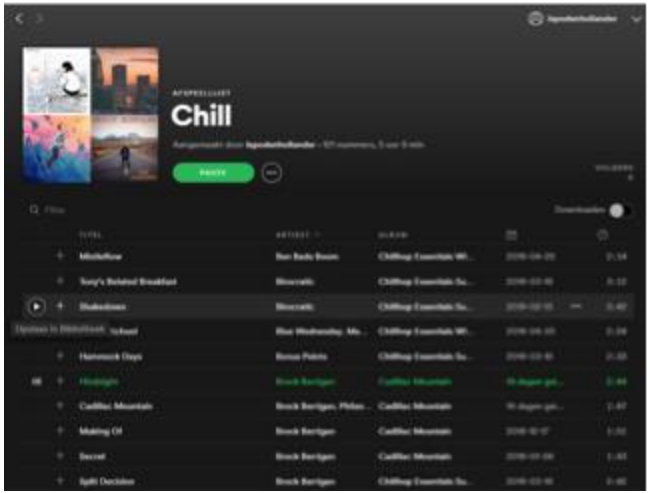
Summary of Dict Methods

Methods	Description
`dict1.clear()`	ลบข้อมูลทั้งหมดจาก dict
`dict1.copy()`	ใช้ในการคัดลอก (shallow copy) dict ทั้งหมด
`dict1.fromkeys([key list], value)`	สร้างข้อมูล dict จาก key-value pairs โดยมี ',' ใช้คั่น
`dict1.get(key)`	การเข้าถึงค่า value จาก key
`dict1.pop(key)`	เอาข้อมูลออกจาก dict โดยใช้key ในการค้นหาข้อมูลเพื่อลบออก
`dict1.update(dict2)`	update ค่า keys and values in a dictionary ด้วย another set of key value pairs

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Nested List to model Dictionary

Spotify Playlist Example




```
playlist  = {'title': 'chill',
             'author':'somename',
             'songs':[
                 {'title':'song1',
                 'artist': ['blue'],
                 'duration':2.5},
                 {'title':'song2',
                 'artist': ['hj', 'dj'],
                 'duration':5},
                 {'title':'song3',
                 'artist': ['tt', 'dd'],
                 'duration':10}
               ]
               }
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Nested List to model Dictionary


```
playlist  = {'title': 'chill',
             'author':'somename',
             'songs':[
             {'title':'song1',
             'artist': ['blue'],
             'duration':2.5},
             {'title':'song2',
             'artist': ['hj', 'dj'],
             'duration':5},
             {'title':'song3',
             'artist': ['tt', 'dd'],
             'duration':10}
             ]
             }
```

Output ของ code นี้คืออะไร

```
print(playlist['title'])
print(playlist['author'])
print(playlist['songs'])
print(playlist['songs'][0])
print(playlist['songs'][0]['artist'])
print(playlist['songs'][0]['duration'])
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Tuple

Syntax numbers = (1,2,3,4) #comma separated values

- การประกาศ tuple นั้นข้อมูลจะอยู่ภายในเครื่องหมาย **“( )”**
- คั่นสมาชิก (item) แต่ละตัวด้วยเครื่องหมาย **“ , ”**
- ข้อมูลใน tuple ไม่สามารถเปลี่ยนแปลงได้ (immutable) ซึ่งต่างกับ list
- ข้อมูลใน tuple มี index คล้าย list นั่นหมายถึง ข้อมูลใน tuple มีลำดับก่อนหลัง
- ส่วนใหญ่นำไปเก็บข้อมูลที่ไม่มีการเปลี่ยนแปลง เช่น **Valid keys in a dictionary**

บาง method เช่น **.items()** ให้ค่า กลับมาเป็น tuple

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

การสร้าง Tuple

- การประกาศ tuple นั้นข้อมูลจะอยู่ภายในเครื่องหมาย “( )”
- คั่นสมาชิก (item) แต่ละตัวด้วยเครื่องหมาย “ , ”

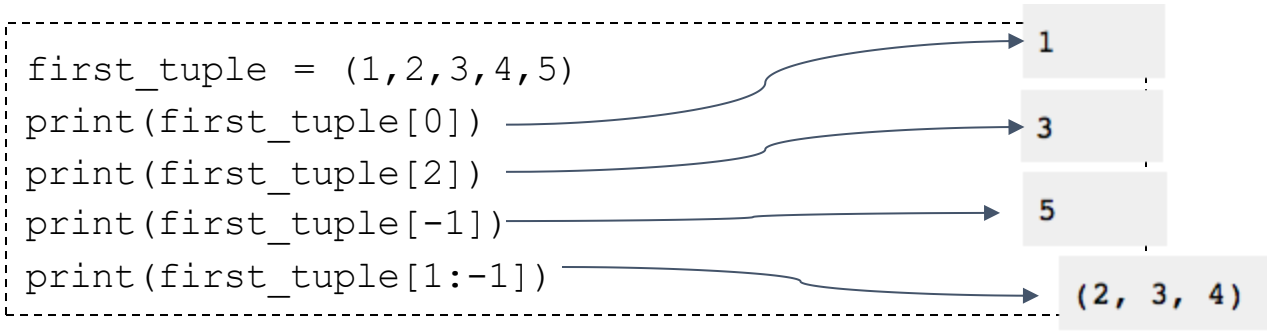
```
# Create a tuple of integers
x = (1,2,3)
3 in x #true
x[0] = 'change me' #typeError
alphabet = ('a','b','c','d')
alphabet.append('e')
alphabet[0] = 'A'
```

```
TypeError: 'tuple' object does not support item assignment
```

```
AttributeError: 'tuple' object has no attribute 'append'
```

```
TypeError: 'tuple' object does not support item assignment
```

•การเข้าถึงข้อมูลโดยใช้ index คล้าย list



Tuple สามารถใช้เป็น key ใน Dictionary

- Tuples สามารถใช้เป็น keys in dictionaries

```
locations = {
    (35.6895, 39.6917): 'Tokyo Office',
    (40.7128, 74.0060): 'NewYork Office',
    (37.7749, 122.4194): 'San Fran Office'
}
print(locations[(35.6895, 39.6917)])
```

```
Tokyo Office
```

Iterate a Tuple (คล้ายกับ List)

- for, while ในการเข้าถึงข้อมูลใน tuple

```
months = ('Jan', 'Feb', 'Mar', 'Apr')
for month in months:
    print(month)

i = len(months)-1
while i>=0:
    print(months[i])
    i-=1
```

```
Jan
Feb
Mar
Apr
```

```
Apr
Mar
Feb
Jan
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Tuple Methods

count(item) : นับจำนวนข้อมูล

index(item) : returns the index ที่เจอข้อมูลนั้น

```
x = (1,2,3,3,3)
print(x.count(3))
print(x.index(1))
print(x.index(5))
print(x.index(3))
#only the first matching index is returned
```

3

0

ValueError

2

หา output ของ code ต่อไปนี้

```
#nested tuple, just like a
list
nums = (1,2,3, (4,5),6)
print(nums[3][1])
print(nums[0:])
print(nums[:4])
```

Set

Syntax `set_data = {1,2,3,4}  #comma separated values`

- การประกาศ set นั้นข้อมูลจะอยู่ภายในเครื่องหมาย **“{ }”**
- คั่นสมาชิก (item) แต่ละตัวด้วยเครื่องหมาย **“ , ”**
- การประกาศ set นั้นคล้ายกันกับ dictionary แต่ข้อมูลใน set ไม่มี key-value pairs.
- ข้อมูลใน set จะไม่ซ้ำกัน (value is unique)
- ข้อมูลใน set สามารถทำ formal mathematical sets (union, intersect) ได้
- ข้อมูลใน set ไม่มีลำดับ (Elements in sets aren't ordered)
- ข้อมูลใน set ไม่สามารถเข้าถึงโดยใช้ index เพราะข้อมูลใน set ไม่มีลำดับ
- ข้อมูลที่เก็บใน set เหมาะกับข้อมูลที่ไม่มีลำดับ ไม่มีข้อมูลซ้ำซ้อน ไม่มี key-value pairs

การสร้าง Set

การสร้าง set เปล่าที่ยังไม่มีข้อมูล

```
# Create a set
set1 = {data}
#or
set1 = set({data})
```

```
# Create a set
s = {1,4,5,'a','b',25.5542}
#notice that the order is changed (cannot
use order to access)
print(s)
s = set({1,4,5})
#or
s = {1,4,5}
#sets cannot have duplicates
s = set({1,2,3,4,5,5,5}) #{1,2,3,4,5}
print(s)
print(4 in s)
print(8 in s)
```

```
{1, 'b', 4, 5, 'a', 25.5542}
{1, 2, 3, 4, 5}
True
False
```

Access a Set


```
# Access all values in a  using loop
 s = set({1,2,3,4,5,5,5})
for x in s:
    print(x, end=' ')
cities = ['A','A','B','C','D']
print(set(cities))   #get unique value
print(len(set(cities)))   #get unique value
print(list(set(cities)))
```

```
{1, 2, 3, 4, 5}
{'D', 'A', 'C', 'B'}
4
['D', 'A', 'C', 'B']
```


Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Set Methods: add(), remove(), discard()

- ใช้ในการเพิ่มหรือลบข้อมูลจาก set

Syntax `set.add(data)`

```
s = set({1,2,3})
s.add(4)      ⟶  {1, 2, 3, 4}
print(s)
s.add(4)
print(s)      ⟶  {1, 2, 3, 4}
```

Syntax `set.remove(data)`

Syntax `set.discard(data)`

ถ้าข้อมูลใน `set.discard(data)` ไม่มี เราจะได้ค่าปัจจุบันของ set กลับมาโดยไม่มี Error Message

```
s = set({1,2,3})
s.remove(1)
print(s)      ⟶  {2, 3, 4}
s.remove(8)
print(s)      ⟶  KeyError, element not in the set
s.discard(8)
print(s)      ⟶  {2, 3, 4}
```


Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

Set Methods: Copy(), clear()

- ใช้ในการคัดลอก (shallow copy) set ทั้งหมด
- ใช้ในการลบข้อมูลออกจาก set ทั้งหมด

Syntax `set.copy()`

```
s = set({1,2,3})
another_s = s.copy()
print(another_s)
print(type(another_s))
```

```
{1, 2, 3, 4}
<class 'set'>
```

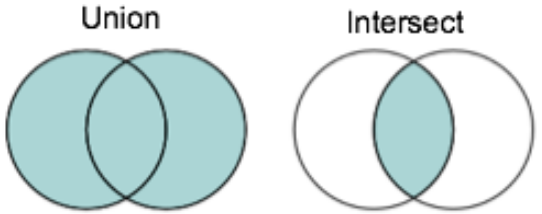
Syntax `set.clear()`

```
s = set({1,2,3})
s.clear()
print(s)
```

```
set()
```

Set Methods: Union, Intersection

- union set สามารถใช้ | ใน expression
- intersect set สามารถใช้ & ใน expression




```
math_students = {"Mathew", "Helen", "Jane"}
programming_students = {"Jane", "Mesut", "Helen"}
#Union to get all students (unique)
print(math_students | programming_students)
#Intersection
print(math_students & programming_students)
```

→ `{'Helen', 'Jane', 'Mathew', 'Mesut'}`

→ `{'Jane', 'Helen'}`