

# Data Structures

โครงสร้างข้อมูล

254251

1<sup>st</sup> semester 2017

# Info

- Instructor
  - Thanathorn Phoka (ธนธร พ่อคำ)
  - Thanathornp[!@#\$%^&]nu.ac.th
  - Room no. 417
- TA
- Programming language
  - Python
- Online resources
  - [https://github.com/fomapavlov/254251\\_1\\_2017](https://github.com/fomapavlov/254251_1_2017)
  - <http://interactivepython.org/runestone/static/pythonds/index.html>
  - <http://pythontutor.com/visualize.html>
  - Youtube 2110101 Computer Programming (Python) บทที่ 1 - 8

# Course description

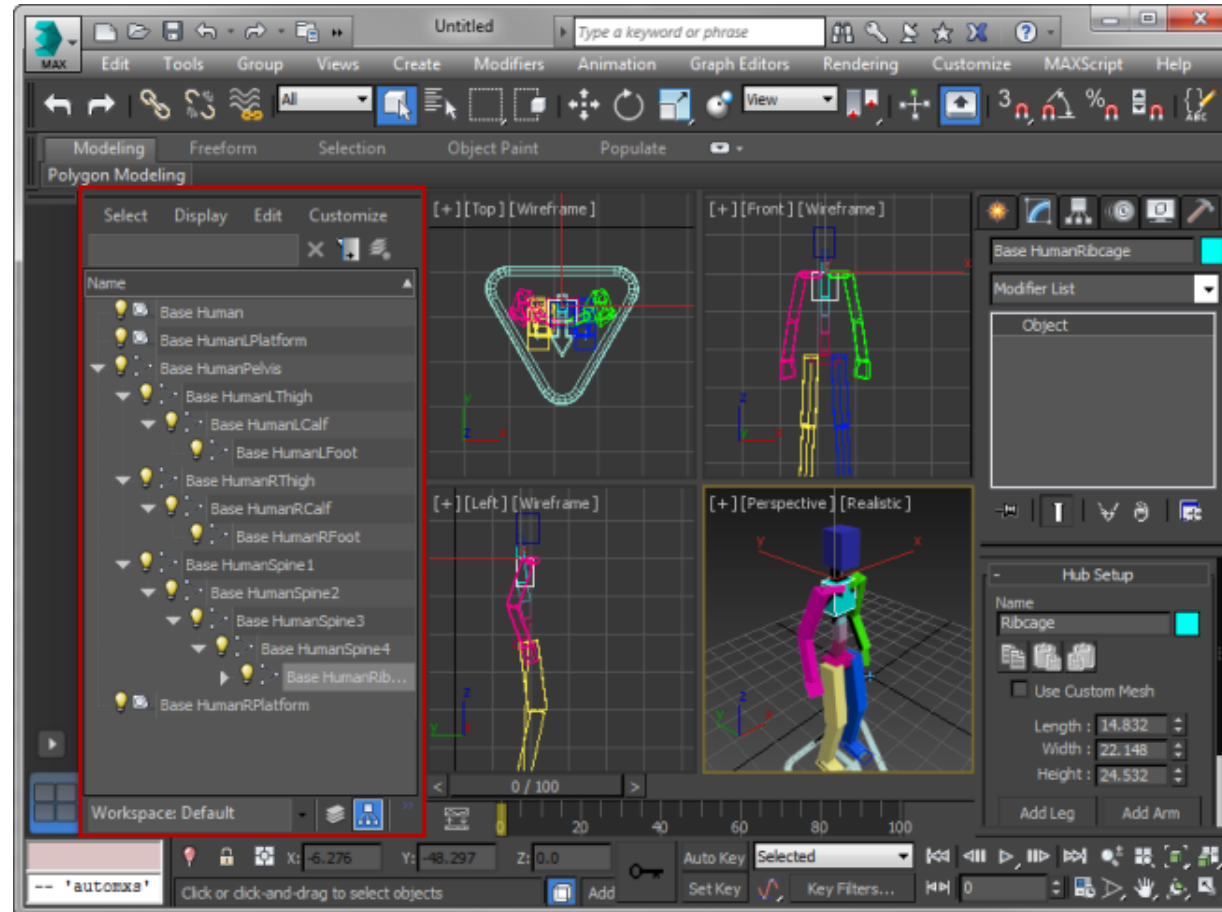
- โครงสร้างข้อมูลพื้นฐาน แอเรย์ สแตกและแถวคอย รายการโยง การเวียนบังเกิด ต้นไม้ค้นหาแบบทวิภาค ต้นไม้เอวีแอล ฮีป ตารางแฮช กราฟ การเรียงลำดับและการค้นหา
- Basic data structure, array, stacks, queues, linked lists, recursion, binary search trees, AVL trees, heaps, graphs, and hash tables, application to sorting and searching

# Why data structures?



<http://mathematica.stackexchange.com/questions/11673/how-to-play-with-facebook-data-inside-mathematica>

# Why data structures?



<https://knowledge.autodesk.com/support/3ds-max/learn-explore/caas/CloudHelp/cloudhelp/2017/ENU/3DSMax/files/GUID-2569461E-C859-4D54-BAFF-C8BD078B53AC-htm.html>

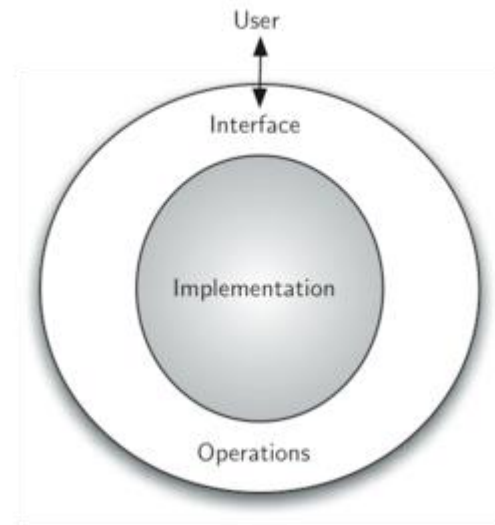
# Data structures and abstract data types

- Abstract Data Types (ADT)

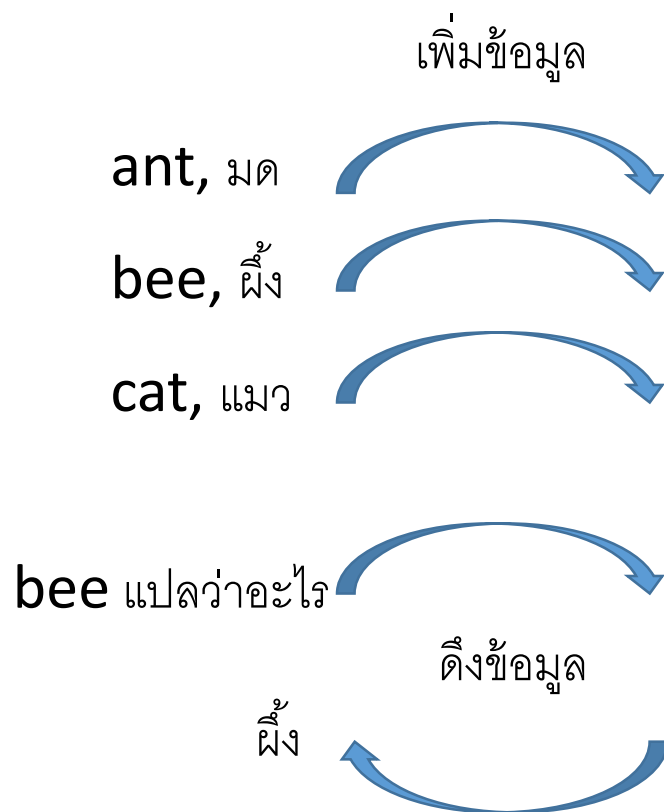
- เป็นมุมมองของผู้ใช้งาน Data structure ที่ไม่จำเป็นต้องรู้ว่า implementation เป็นอย่างไร

- Data structures

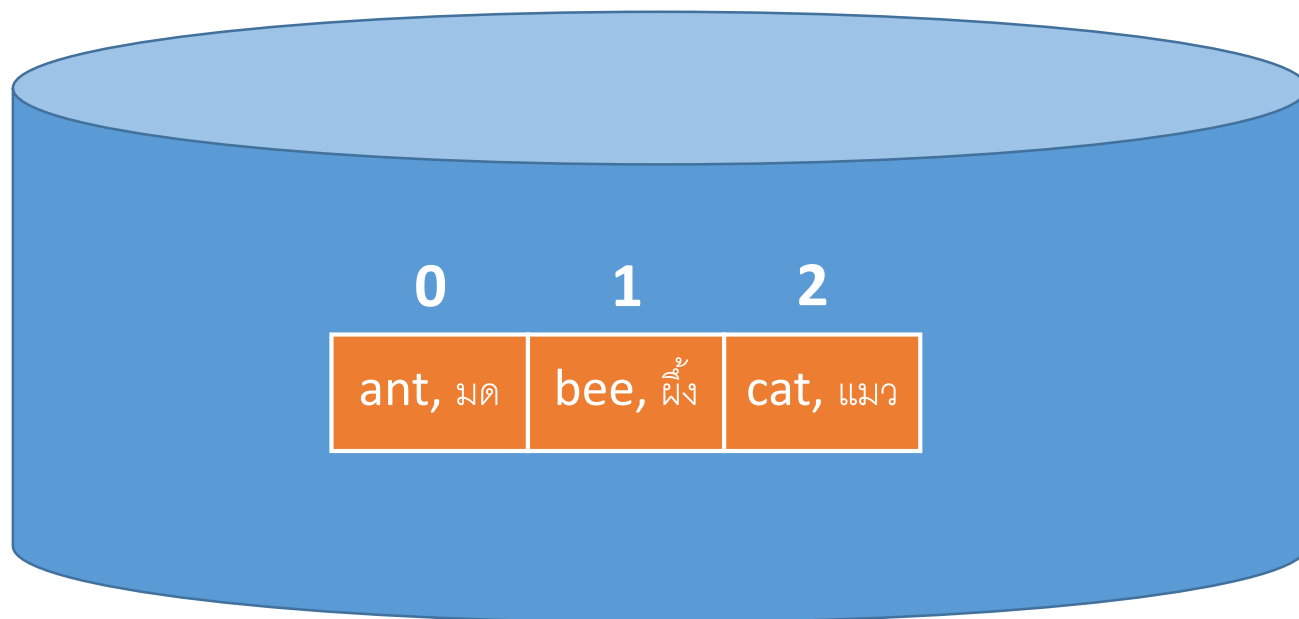
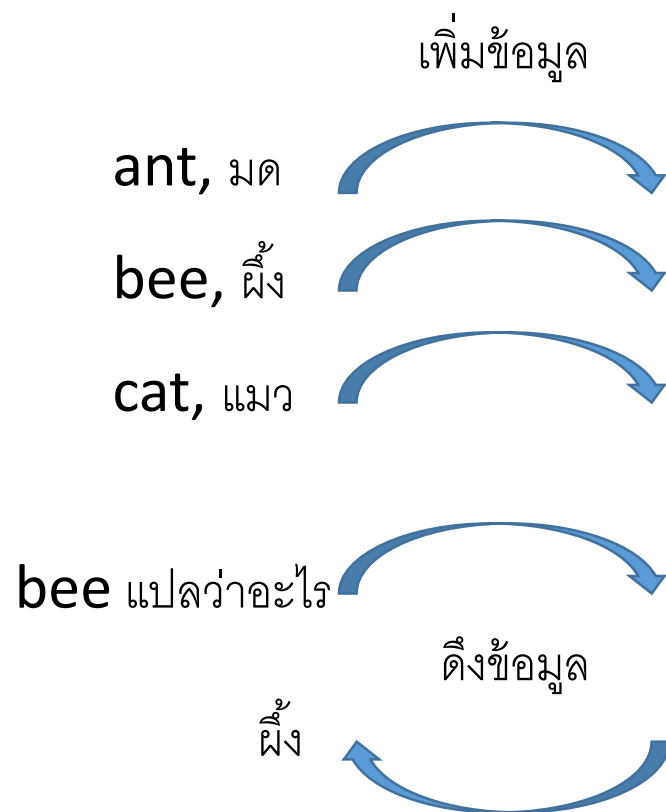
- เป็นส่วน implementation



# Dictionary

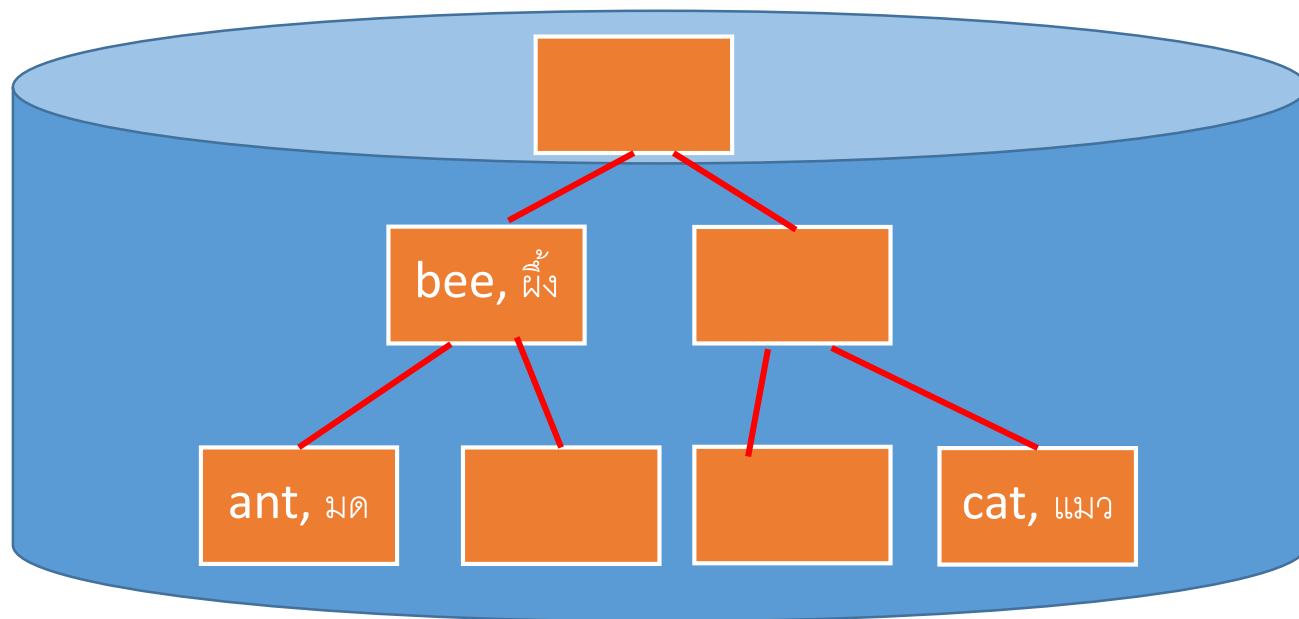
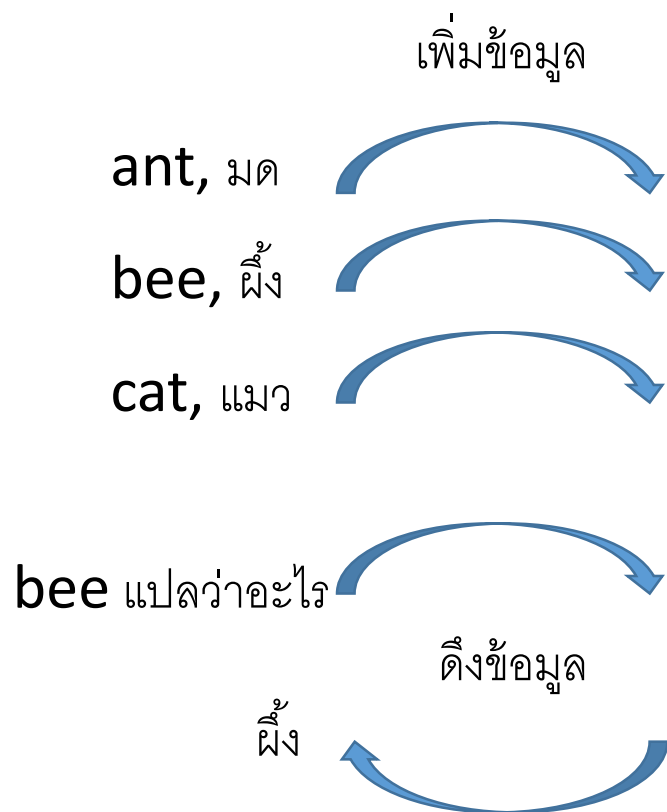


# Dictionary





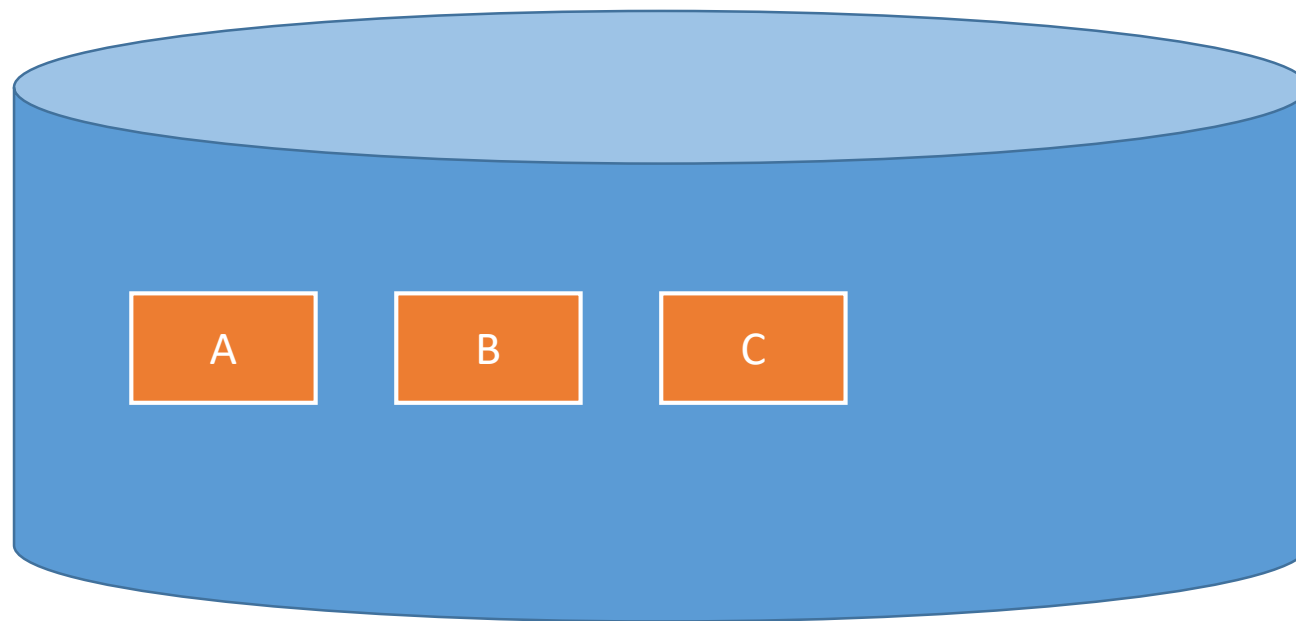
# Dictionary



# ที่เก็บข้อมูลพื้นฐาน

- Set

- ไม่มีอันดับ ซ้ำไม่ได้



# ที่เก็บข้อมูลพื้นฐาน

- List
  - มีอันดับ เข้าได้



# ที่เก็บข้อมูลพื้นฐาน

- List
  - มีอันดับ เข้าได้



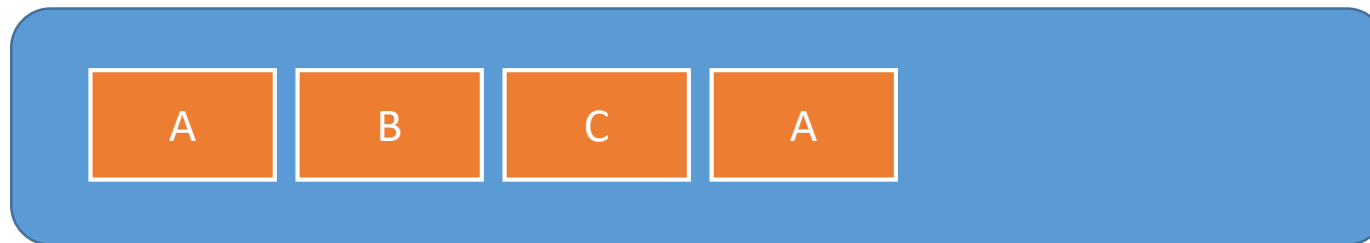
# ที่เก็บข้อมูลพื้นฐาน

- List
  - มีอันดับ เข้าได้



# ที่เก็บข้อมูลพื้นฐาน

- List
  - มีอันดับ เข้าได้



# ที่เก็บข้อมูลพื้นฐาน

- Queue

- มีอันดับ เข้าก่อน ออกก่อน



# ที่เก็บข้อมูลพื้นฐาน

- Queue

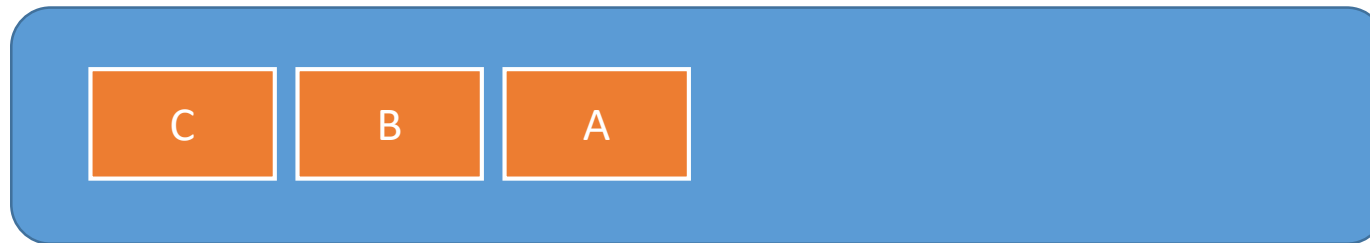
- มีอันดับ เข้าก่อน ออกก่อน





# ที่เก็บข้อมูลพื้นฐาน

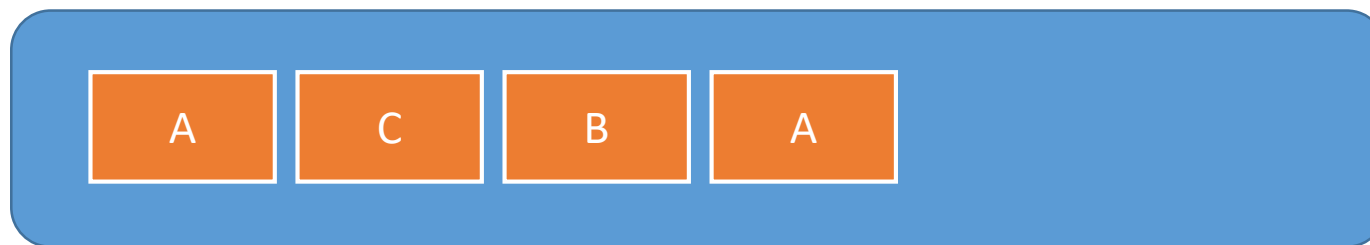
- Queue
  - มีอันดับ เข้าก่อน ออกก่อน



# ที่เก็บข้อมูลพื้นฐาน

- Queue

- มีอันดับ เข้าก่อน ออกก่อน



# ที่เก็บข้อมูลพื้นฐาน

- Queue

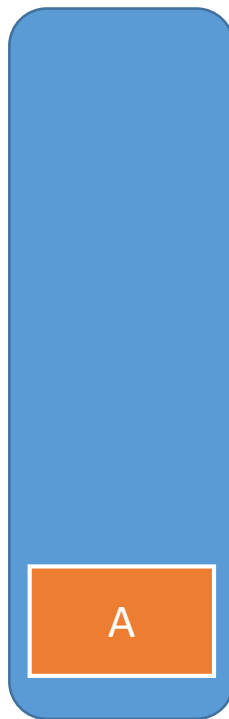
- มีอันดับ เข้าก่อน ออกก่อน



# ที่เก็บข้อมูลพื้นฐาน

- Stack

- มีอันดับ เข้าหลัง ออกก่อน



# ที่เก็บข้อมูลพื้นฐาน

- Stack

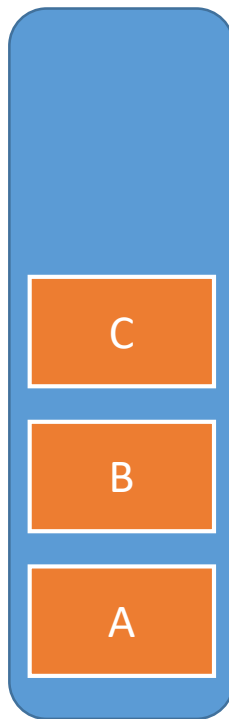
- มีอันดับ เข้าหลัง ออกก่อน



# ที่เก็บข้อมูลพื้นฐาน

- Stack

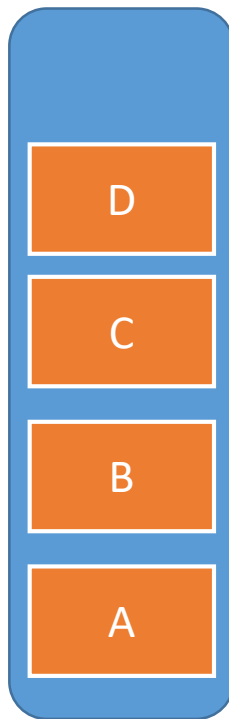
- มีอันดับ เข้าหลัง ออกก่อน



# ที่เก็บข้อมูลพื้นฐาน

- Stack

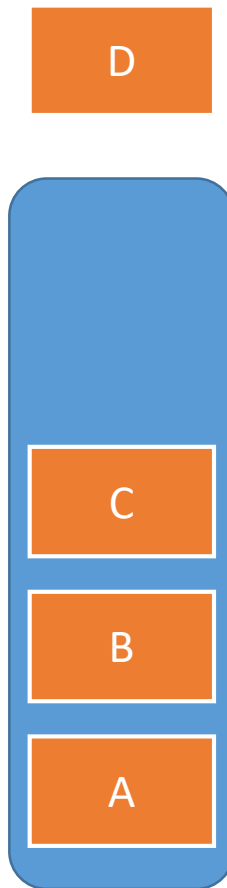
- มีอันดับ เข้าหลัง ออกก่อน



# ที่เก็บข้อมูลพื้นฐาน

- Stack

- มีอันดับ เข้าหลัง ออกก่อน



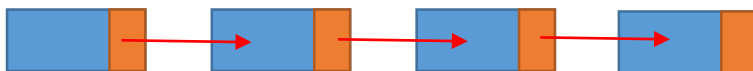


# วิธีสร้างที่เก็บข้อมูล

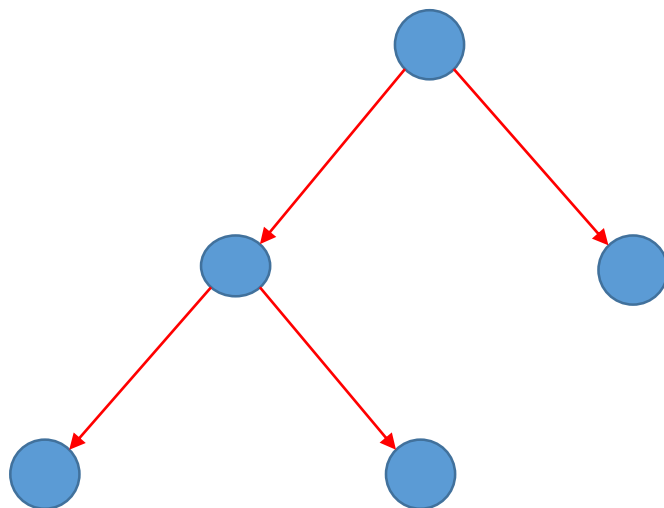
- ใช้ array



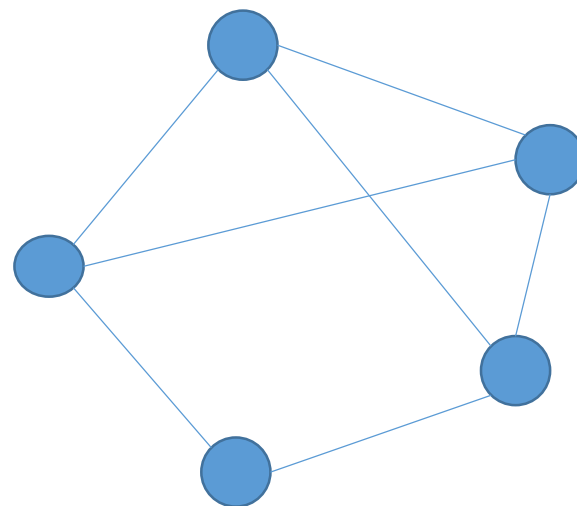
- ใช้การโยง



- ใช้ต้นไม้



- ใช้กราฟ



- ใช้ตาราง

# วัตถุประสงค์

- เลือกใช้ให้เหมาะสมกับปัญหา
- รู้ว่าจะใช้งานแต่ละ **data structure** อย่างไร
- สามารถ **implement** ได้

# Basic Python

# Our editors & shell

- <http://pythontutor.com/visualize.html>
- <https://repl.it/languages/python3>

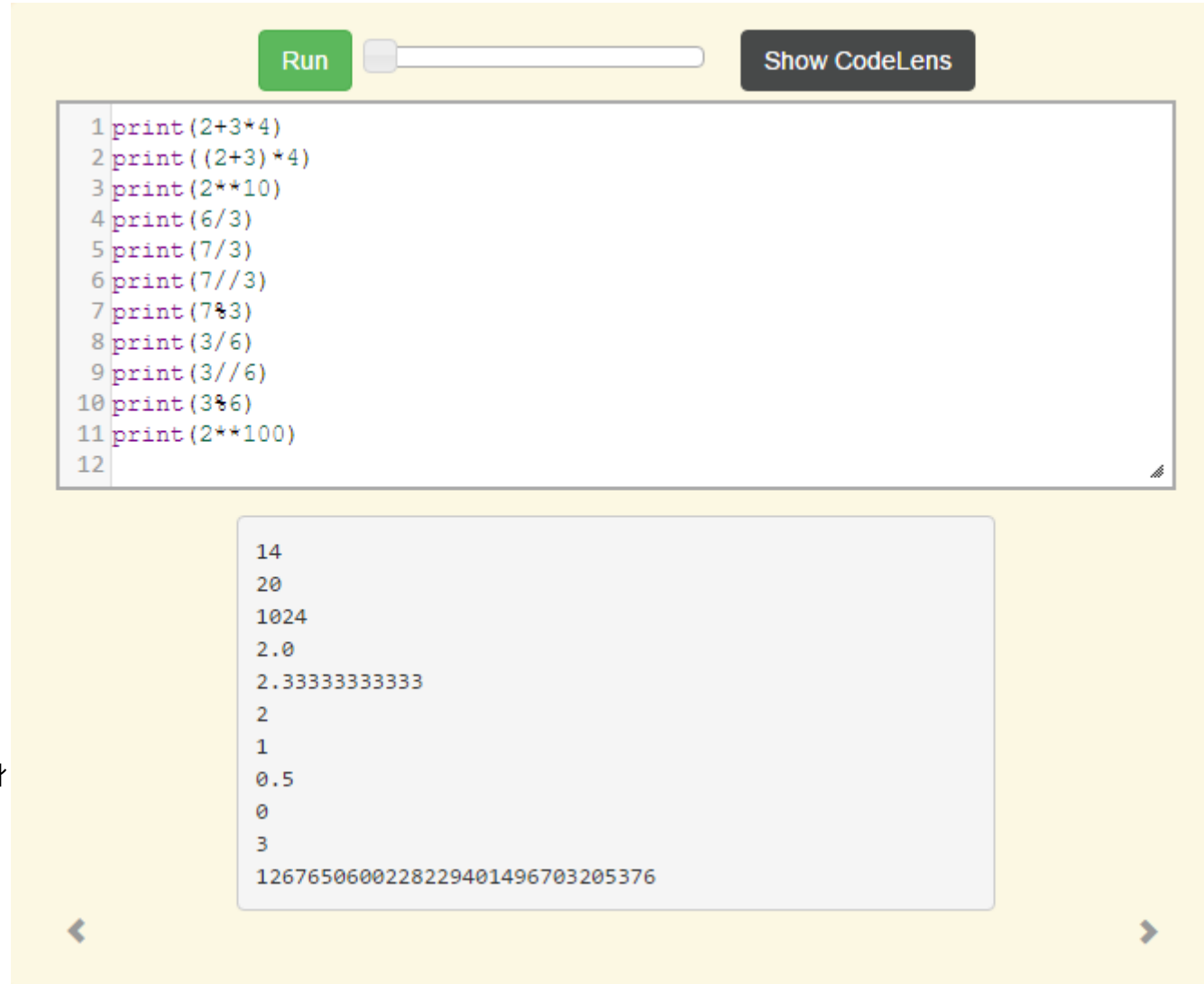
# Indentation based

- ไม่มี `begin end` หรือ `{ }` เป็นตัวบอก scope
- ใช้ `space` หรือ `tab` เข้าไปแทน
  - ระวังเรื่องการใช้ `space` และ `tab` (บาง editor ไม่แปลง `tab` เป็น `space` ให้)

```
def fib(n):  
    ↑—— print 'n =', n  
    ↑—— if n > 1:  
        ↑—— return n * fib(n - 1)  
    else:  
        ↑—— print 'end of the line'  
        ↑—— return 1
```

# Numeric Data types

- int – จำนวนเต็ม
- float – จำนวนทศนิยม
- Operation
  - +
  - -
  - \*
  - /
  - \*\* - ยกกำลัง
  - % - หารจำนวนเต็มเอาเศษ
  - // - หารจำนวนเต็ม



The screenshot shows a Python code editor with a 'Run' button and a 'Show CodeLens' button. The code consists of 12 lines of print statements. Below the code, the output is displayed in a separate window.

```
1 print(2+3*4)
2 print((2+3)*4)
3 print(2**10)
4 print(6/3)
5 print(7/3)
6 print(7//3)
7 print(7%3)
8 print(3/6)
9 print(3//6)
10 print(3%6)
11 print(2**100)
12
```

Output:

```
14
20
1024
2.0
2.333333333333333
2
1
0.5
0
3
1267650600228229401496703205376
```

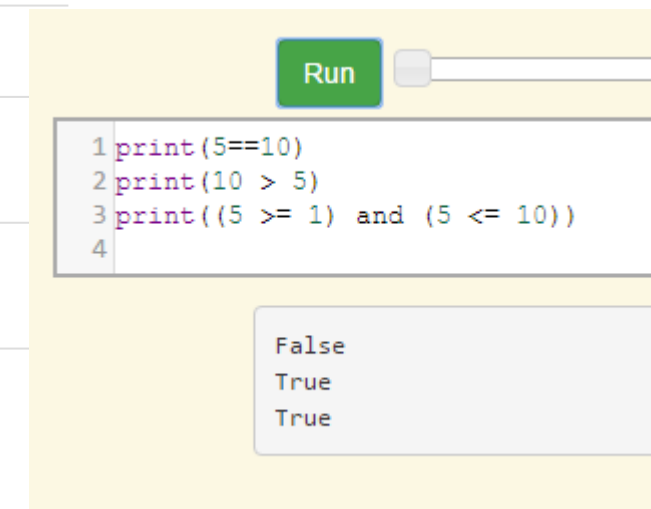
# Boolean

- True
- False
- Operation
  - and
  - Or
  - not

```
>>> True
True
>>> False
False
>>> False or True
True
>>> not (False or True)
False
>>> True and True
True
```

# Logical operators

Operation Name	Operator	Explanation
less than	<	น้อยกว่า
greater than	>	มากกว่า
less than or equal	<=	น้อยกว่าเท่ากับ
greater than or equal	>=	มากกว่าเท่ากับ
equal	==	เท่ากันหรือไม่
not equal	!=	ไม่เท่ากันหรือไม่
logical and	and	True and True = True
logical or	or	True ตัวใดตัวหนึ่ง = True
logical not	not	นิเสธ



```
1 print(5==10)
2 print(10 > 5)
3 print((5 >= 1) and (5 <= 10))
4
```

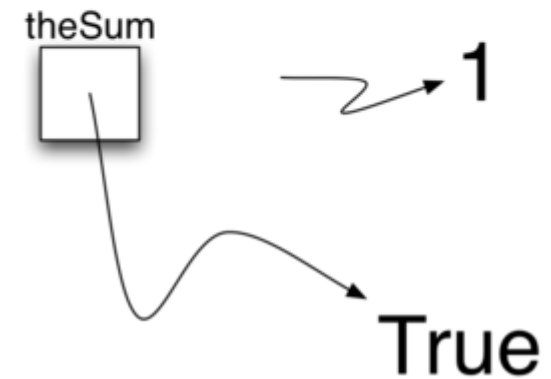
False  
True  
True



# การกำหนดค่าแบบอ้างอิง

- ไม่ยึดติดกับ data type

```
int → >>> theSum = 0
      >>> theSum
      0
      >>> theSum = theSum + 1
      >>> theSum
      1
boolean → >>> theSum = True
           >>> theSum
           True
```



# Data structures พื้นฐานใน python

- มีอันดับ
  - List (หรือ array), string, tuple
- ไม่มีอันดับ
  - Set, dictionary

# List

- สร้างด้วย [ ] คั่นด้วย ,
- ไม่จำเป็นต้องเป็น **data type** แบบเดียวกัน
- [ ] คือ **list** ว่าง

```
>>> [1,3,True,6.5]  
[1, 3, True, 6.5]  
>>> myList = [1,3,True,6.5]  
>>> myList  
[1, 3, True, 6.5]
```

- เริ่มต้น **list** ด้วยค่าเดียวกัน

```
>>> myList = [0] * 6  
>>> myList  
[0, 0, 0, 0, 0, 0]
```

# List's operators

Operation Name	Operator	Explanation
indexing	[ ]	ดึงค่าจากตำแหน่งที่กำหนดใน [ ]
concatenation	+	รวมลำดับ
repetition	*	รวมลำดับแบบทำซ้ำเป็นจำนวนครั้ง
membership	in	ถามว่ามี item ใน list หรือไม่
length	len	จำนวน item ใน list
slicing	[ : ]	ดึงลำดับมาบางส่วน

Run

Show CodeLens

```
1 myList = [1,2,3,4]
2 A = [myList]*3
3 print(A)
4 myList[2]=45
5 print(A)
6
7
```

[[1, 2, 3, 4], [1, 2, 3, 4], [1, 2, 3, 4]]  
[[1, 2, 45, 4], [1, 2, 45, 4], [1, 2, 45, 4]]

# List's methods

Method Name	Use	Explanation
append	<code>alist.append(item)</code>	เพิ่ม <b>item</b> ท้าย <b>list</b>
insert	<code>alist.insert(i,item)</code>	เพิ่ม <b>item</b> ที่ตำแหน่ง <b>i</b>
pop	<code>alist.pop()</code>	คืนค่า <b>item</b> ที่ตำแหน่งสุดท้ายแล้วเอาออกด้วย
pop	<code>alist.pop(i)</code>	คืนค่า <b>item</b> ที่ตำแหน่ง <b>i</b> แล้วเอาออกด้วย
sort	<code>alist.sort()</code>	เรียงลำดับใน <b>list</b>
reverse	<code>alist.reverse()</code>	ย้อนลำดับใน <b>list</b>
del	<code>del alist[i]</code>	ลบ <b>item</b> ที่ตำแหน่ง <b>i</b>
index	<code>alist.index(item)</code>	ค้นหา <b>item</b> แล้วคืนค่าตำแหน่งที่เจอมาให้
count	<code>alist.count(item)</code>	นับจำนวนที่มีค่าเดียวกับ <b>item</b> ใน <b>list</b>
remove	<code>alist.remove(item)</code>	ลบตัวที่มีค่าเดียวกับ <b>item</b> ตัวแรกที่เจอ

# ตัวอย่าง

Run

Show CodeLens

```
1 myList = [1024, 3, True, 6.5]
2 myList.append(False)
3 print(myList)
4 myList.insert(2, 4.5)
5 print(myList)
6 print(myList.pop())
7 print(myList)
8 print(myList.pop(1))
9 print(myList)
10 myList.pop(2)
11 print(myList)
12 myList.sort()
13 print(myList)
14 myList.reverse()
15 print(myList)
16 print(myList.count(6.5))
17 print(myList.index(4.5))
18 myList.remove(6.5)
19 print(myList)
20 del myList[0]
21 print(myList)
22
```

```
[1024, 3, True, 6.5, False]
[1024, 3, 4.5, True, 6.5, False]
False
[1024, 3, 4.5, True, 6.5]
3
[1024, 4.5, True, 6.5]
[1024, 4.5, 6.5]
[4.5, 6.5, 1024]
[1024, 6.5, 4.5]
1
2
[1024, 4.5]
[4.5]
```

# range

- เป็น **function** ที่ใช้สร้างลำดับของตัวเลข
- ใช้บ่อยใน **for loop**
- การทำงานต่างกันตามจำนวน **parameter**
  - **range(n)** สร้าง 0 ถึง n-1
  - **range(n, m)** สร้าง n ถึง m-1
  - **range(n, m, o)** สร้าง n ด้วย step o และไม่เกิน m

```
>>> range(10)
range(0, 10)
>>> list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> range(5,10)
range(5, 10)
>>> list(range(5,10))
[5, 6, 7, 8, 9]
>>> list(range(5,10,2))
[5, 7, 9]
>>> list(range(10,1,-1))
[10, 9, 8, 7, 6, 5, 4, 3, 2]
>>>
```

# String

Method Name	Use	Explanation
center	asString.center(w)	สร้าง string ความยาว w โดยมีค่าเดิมอยู่ตรงกลาง
count	asString.count(item)	จำนวนที่ค้นหาค่าเดียวกับ item เจอ
ljust	asString.ljust(w)	สร้าง string ความยาว w โดยมีค่าเดิมอยู่ด้านซ้าย
lower	asString.lower()	สร้าง string ที่เป็น lower case
rjust	asString.rjust(w)	สร้าง string ความยาว w โดยมีค่าเดิมอยู่ด้านขวา
find	asString.find(item)	คืนค่า index แรกที่เจอตัวที่มีค่าเดียวกับ item
split	asString.split(schar)	แบ่ง string ด้วยตัวคั่น schar (ใช้บ่อย)

```
>>> "David"
'David'
>>> myName = "David"
>>> myName[3]
'i'
>>> myName*2
'DavidDavid'
>>> len(myName)
5
>>>
```

```
>>> myName
'David'
>>> myName.upper()
'DAVID'
>>> myName.center(10)
'  David  '
>>> myName.find('v')
2
>>> myName.split('v')
['Da', 'id']
```



# Mutability

- ความสามารถในการเปลี่ยนแปลงค่า
- List ทำได้
- String ทำไม่ได้

```
>>> myList
[1, 3, True, 6.5]
>>> myList[0]=2**10
>>> myList
[1024, 3, True, 6.5]
>>>
>>> myName
'David'
>>> myName[0]='X'

Traceback (most recent call last):
  File "<pyshell#84>", line 1, in -toplevel-
    myName[0]='X'
TypeError: object doesn't support item assignment
>>>
```

# Tuple

- คล้าย **list** แต่ **immutable**
- สร้างโดยอยู่ใน ( ) คั่นด้วย ,

```
>>> myTuple = (2,True,4.96)
>>> myTuple
(2, True, 4.96)
>>> len(myTuple)
3
>>> myTuple[0]
2
>>> myTuple * 3
(2, True, 4.96, 2, True, 4.96, 2, True, 4.96)
>>> myTuple[0:2]
(2, True)
>>>
```

```
>>> myTuple[1]=False
```

```
Traceback (most recent call last):
```

```
  File "<pysHELL#137>", line 1, in -toplevel-
```

```
    myTuple[1]=False
```

```
TypeError: object doesn't support item assignment
```

```
>>>
```

# Set

- Immutable, ไม่มีอันดับ
- สร้างโดยอยู่ใน { } คั่นด้วย ,

```
>>> {3,6,"cat",4.5,False}
{False, 4.5, 3, 6, 'cat'}
>>> mySet = {3,6,"cat",4.5,False}
>>> mySet
{False, 4.5, 3, 6, 'cat'}
>>>
```

# Set's operators

Operation Name	Operator	Explanation
membership	in	ถามว่ามีค่านั้นๆ ใน set หรือไม่
length	len	ขนาดของ set
	aset   otherset	union ของ 2 sets
&	aset & otherset	Intersection ของ 2 sets
-	aset - otherset	Difference ของ 2 sets คือ ตัวที่มีใน set แรกแต่ไม่มีใน set ที่สอง
<=	aset <= otherset	ถามว่าสมาชิกใน set แรกน้อยกว่าเท่ากับสมาชิกใน set ที่สอง ทั้งหมดหรือไม่

```
>>> mySet
{False, 4.5, 3, 6, 'cat'}
>>> len(mySet)
5
>>> False in mySet
True
>>> "dog" in mySet
False
>>>
```

# Set's methods

Method Name	Use	Explanation
union	aset.union(otherset)	union ของ 2 sets
intersection	aset.intersection(otherset)	Intersection ของ 2 sets
difference	aset.difference(otherset)	Difference ของ 2 sets คือ ตัวที่มีใน set แรกแต่ไม่มีใน set ที่สอง
issubset	aset.issubset(otherset)	ถามว่าเป็น subset ของ otherset หรือไม่
add	aset.add(item)	เพิ่ม item เข้าใน set
remove	aset.remove(item)	ลบ item ออกจาก set
pop	aset.pop()	คืนค่าและลบทิ้งสมาชิก 1 ตัว
clear	aset.clear()	ลบ item ทั้งหมด

```
>>> mySet
{False, 4.5, 3, 6, 'cat'}
>>> yourSet = {99,3,100}
>>> mySet.union(yourSet)
{False, 4.5, 3, 100, 6, 'cat', 99}
>>> mySet | yourSet
{False, 4.5, 3, 100, 6, 'cat', 99}
>>> mySet.intersection(yourSet)
{3}
>>> mySet & yourSet
{3}
>>> mySet.difference(yourSet)
{False, 4.5, 6, 'cat'}
>>> mySet - yourSet
{False, 4.5, 6, 'cat'}
>>> {3,100}.issubset(yourSet)
True
>>> {3,100}<=yourSet
True
>>> mySet.add("house")
>>> mySet
{False, 4.5, 3, 6, 'house', 'cat'}
>>> mySet.remove(4.5)
>>> mySet
{False, 3, 6, 'house', 'cat'}
>>> mySet.pop()
False
>>> mySet
{3, 6, 'house', 'cat'}
>>> mySet.clear()
>>> mySet
set()
>>>
```

# Dictionary

- Mutable, ไม่มีอันดับ
- คู่ลำดับของข้อมูล
  - Key
  - Value
- สร้างโดยอยู่ใน { } มี : คั่นระหว่าง key:value มี , คั่นระหว่างคู่ลำดับ

```
>>> capitals = {'Iowa': 'DesMoines', 'Wisconsin': 'Madison'}
>>> capitals
{'Wisconsin': 'Madison', 'Iowa': 'DesMoines'}
>>>
```

# Dictionary's operators

Operator	Use	Explanation
[]	myDict[key]	คืนค่าที่สัมพันธ์กับ key
in	key in dict	มี key อยู่ใน dictionary หรือไม่
del	del dict[key]	ลบคู่ลำดับที่สัมพันธ์กับ key

Run

Hide Codelens

```
1 capitals = {'Iowa': 'DesMoines', 'Wisconsin': 'Madison'}
2 print(capitals['Iowa'])
3 capitals['Utah'] = 'SaltLakeCity'
4 print(capitals)
5 capitals['California'] = 'Sacramento'
6 print(len(capitals))
7 for k in capitals:
8     print(capitals[k], " is the capital of ", k)
9
```

DesMoines  
{'Iowa': 'DesMoines', 'Wisconsin': 'Madison', 'Utah': 'SaltLakeCity'}  
4  
DesMoines is the capital of Iowa  
Madison is the capital of Wisconsin  
SaltLakeCity is the capital of Utah  
Sacramento is the capital of California

# Dictionary's methods

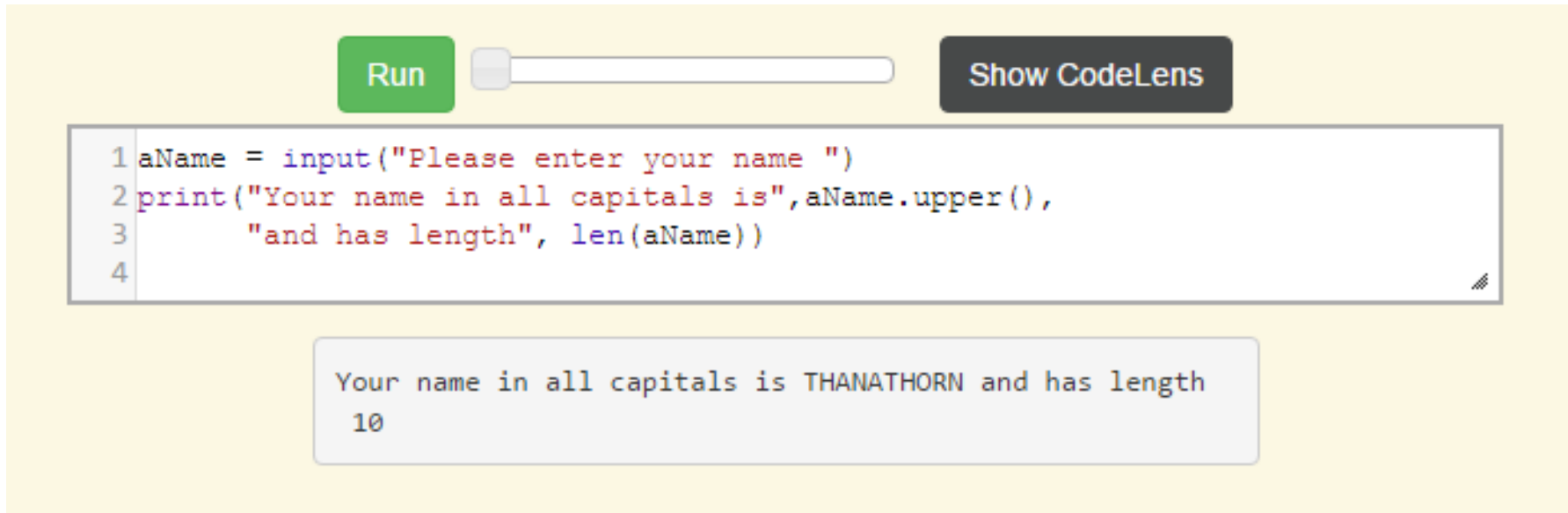
Method Name	Use	Explanation
keys	dict.keys()	คืนค่า <b>key</b> ทั้งหมด
values	dict.values()	คืนค่า <b>value</b> ทั้งหมด
items	dict.items()	คืนค่าคู่ลำดับทั้งหมด
get	dict.get(k)	คืนค่าที่สัมพันธ์กับ <b>key k</b> ถ้าไม่เจอจะคืนค่า <b>None</b>
get	dict.get(k,alt)	คืนค่าที่สัมพันธ์กับ <b>key k</b> ถ้าไม่เจอจะคืนค่า <b>alt</b>

```
>>> phoneext={'david':1410,'brad':1137}
>>> phoneext
{'brad': 1137, 'david': 1410}
>>> phoneext.keys()
dict_keys(['brad', 'david'])
>>> list(phoneext.keys())
['brad', 'david']
>>> phoneext.values()
dict_values([1137, 1410])
>>> list(phoneext.values())
[1137, 1410]
>>> phoneext.items()
dict_items([('brad', 1137), ('david', 1410)])
>>> list(phoneext.items())
[('brad', 1137), ('david', 1410)]
>>> phoneext.get("kent")
>>> phoneext.get("kent","NO ENTRY")
'NO ENTRY'
>>>
```



# Input และ output

- in = input(string)
- Output ใช้ print(string)



The screenshot shows a Python code editor interface. At the top, there is a green 'Run' button, a progress slider, and a dark grey 'Show CodeLens' button. Below these is a code editor with the following Python code:


```
1 aName = input("Please enter your name ")
2 print("Your name in all capitals is", aName.upper(),
3       "and has length", len(aName))
4
```

Below the code editor, the output of the program is displayed in a light grey box:

```
Your name in all capitals is THANATHORN and has length
10
```

# Input ข้อควรระวัง

- `in = input(string)`
  - สิ่งที่คืนค่าให้กับ `in` คือ `string`
  - ก่อนนำไปใช้ต้องรู้ว่าเราจะเอาไปใช้งานเป็น `data type` แบบไหน
  - ใช้เป็น `float` ก็ต้องแปลงเป็น `float` ก่อน



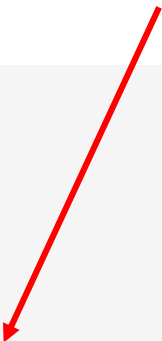
```
sradius = input("Please enter the radius of the circle ")  
radius = float(sradius)  
diameter = 2 * radius
```

# String format


- Python 3 เท่านั้น !!!!

```
>>> print("Hello")
Hello
>>> print("Hello","World")
Hello World
>>> print("Hello","World", sep="***")
Hello***World
>>> print("Hello","World", end="***")
Hello World***>>>
```

ตัวคั่น



ลงท้าย



# String format

- % ที่อยู่ใน “ ” บอกเราว่าจะพิมพ์ออกเป็น data type ไหน
  - %d จำนวนเต็ม
  - %f จำนวนทศนิยม
  - %s string
  - อื่นๆ ลองไปศึกษาดู
- % ที่อยู่ข้างนอกบอกว่าตัวแปรข้างหลังจะนำมาแทนที่ % ที่อยู่ใน “ ” ตามลำดับ

```
> aName = "Eiei"
=> None
> age = 16
=> None
> print("%s is %d years old." % (aName, age))
Eiei is 16 years old.
```

# String format

Modifier	Example	Description
number	%20d	พิมพ์ค่าโดยจองที่ให้ 20 ตัว ชิดขวา
-	%-20d	พิมพ์ค่าโดยจองที่ให้ 20 ตัว ชิดซ้าย
+	%+20d	พิมพ์ค่าโดยจองที่ให้ 20 ตัว ชิดซ้าย
0	%020d	พิมพ์ค่าโดยจองที่ให้ 20 ตัว เต็ม 0 ให้ข้างหน้าจนเต็ม (ใช้ทำอะไร ลองคิดดู)
.	%20.2f	พิมพ์ค่าโดยจองที่ให้ 20 ตัว ทศนิยม 2 ตำแหน่ง
(name)	%(name)d	นำค่าจาก dictionary มาพิมพ์ตาม key name

# String format

```
>>> price = 24
>>> item = "banana"
>>> print("The %s costs %d cents"%(item,price))
The banana costs 24 cents
>>> print("The %+10s costs %5.2f cents"%(item,price))
The      banana costs 24.00 cents
>>> print("The %+10s costs %10.2f cents"%(item,price))
The      banana costs      24.00 cents
>>> itemdict = {"item":"banana","cost":24}
>>> print("The %(item)s costs %(cost)7.1f cents"%itemdict)
The banana costs      24.0 cents
>>>
```

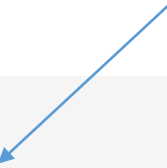
# Control Structures

- while
- for
- If elif else

# while

- ทำไปเรื่อยๆ เมื่อเงื่อนไขยังเป็นจริงอยู่

เงื่อนไข



```
>>> counter = 1
>>> while counter <= 5:
...     print("Hello, world")
...     counter = counter + 1
```

```
Hello, world
Hello, world
Hello, world
Hello, world
Hello, world
```



# while

- ลองใน <http://pythontutor.com/visualize.html>

The screenshot displays the Python Tutor interface for Python 2.7. The code being executed is a while loop that prints 'hello' five times. The execution is currently at line 2, where the loop condition is being checked. A green arrow points to line 1, indicating it has just been executed. A red arrow points to line 2, indicating it is the next line to be executed. The right-hand side of the interface shows the 'Print output' area, which is currently empty. Below the code, there are links for 'Edit code' and 'Live programming'. At the bottom, there is a progress bar and navigation buttons: '<< First', '< Back', 'Step 2 of 17', 'Forward >', and 'Last >>'. The 'Frames' panel on the right shows the 'Global frame' with a variable 'counter' set to 1.

Python 2.7

```
→ 1 counter = 1
→ 2 while counter <= 5:
  3     print("hello")
  4     counter = counter + 1
```

[Edit code](#) | [Live programming](#)

→ line that has just executed  
→ next line to execute

**NEW!** Click on a line of code to set a breakpoint. Then use the Forward and Back buttons to jump there.

Print output (drag lower right corner to resize)

Frames      Objects

Global frame  
counter | 1


<< First   < Back   Step 2 of 17   Forward >   Last >>

# for

- ทำงานกับทุก **item** ใน **list**

```
>>> for item in [1,3,6,2,5]:  
...     print(item)  
...  
1  
3  
6  
2  
5
```

[0, 1, 2, 3, 4]



```
>>> for item in range(5):  
...     print(item**2)  
...  
0  
1  
4  
9  
16  
>>>
```

# for ซ้อน for

Run

Hide Codelens

```
1 wordlist = ['cat', 'dog', 'rabbit']
2 letterlist = [ ]
3 for aword in wordlist:
4     for aletter in aword:
5         letterlist.append(aletter)
6 print(letterlist)
7
```

Python 3.3

```
1 wordlist = ['cat', 'dog', 'rabbit']
2 letterlist = [ ]
→ 3 for aword in wordlist:
→ 4     for aletter in aword:
5         letterlist.append(aletter)
6 print(letterlist)
```

→ line that has just executed  
→ next line to execute

< Back Step 11 of 34 Forward >

Print output (drag lower right corner to resize)

Frames

Objects

Global frame	
wordlist	
letterlist	
aword	"cat"
aletter	"t"

list

0	1	2
"cat"	"dog"	"rabbit"

list

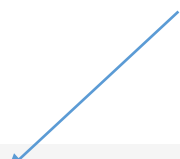
0	1	2
"c"	"a"	"t"

Visualized using [Online Python Tutor](#) by Philip Guo

# if else

- ทำใน **if** ถ้าตรงกับเงื่อนไข
- ทำใน **else** ถ้าไม่ตรงกับเงื่อนไข

เงื่อนไข



```
if n<0:
    print("Sorry, value is negative")
else:
    print(math.sqrt(n))
```

```
if score >= 90:
    print('A')
else:
    if score >=80:
        print('B')
    else:
        if score >= 70:
            print('C')
        else:
            if score >= 60:
                print('D')
            else:
                print('F')
```

# if elif else

- ทำใน **if** ถ้าตรงกับเงื่อนไข
- ทำใน **elif** ถ้าตรงกับอีกเงื่อนไข
- ทำใน **else** ถ้าไม่ตรงกับเงื่อนไขไหนเลย

```
if score >= 90:  
    print('A')  
elif score >= 80:  
    print('B')  
elif score >= 70:  
    print('C')  
elif score >= 60:  
    print('D')  
else:  
    print('F')
```

# for ในการสร้าง list

- Loop เพิ่ม item

```
>>> sqllist=[]
>>> for x in range(1,11):
        sqllist.append(x*x)

>>> sqllist
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
>>>
```

- อีกวิธี

```
>>> sqllist=[x*x for x in range(1,11)]
>>> sqllist
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
>>>
```

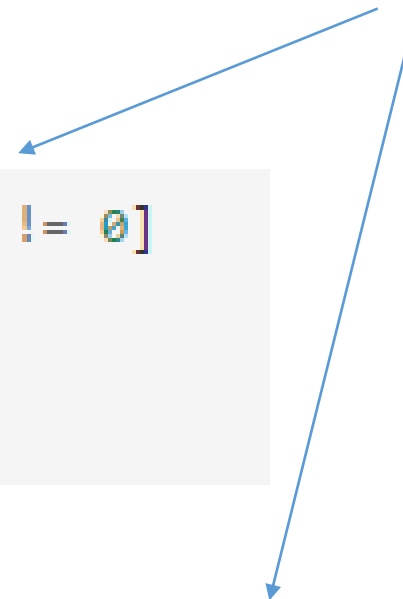
# for และ if ในการสร้าง list

- เพิ่มใน **list** เมื่อตรงกับเงื่อนไข

```
>>> sqlist=[x*x for x in range(1,11) if x%2 != 0]
>>> sqlist
[1, 9, 25, 49, 81]
>>>
```

```
>>>[ch.upper() for ch in 'comprehension' if ch not in 'aeiou']
['C', 'M', 'P', 'R', 'H', 'N', 'S', 'N']
>>>
```

เงื่อนไข



# Exception handling

- **Exception** คือความผิดพลาดทาง logic
- ไม่จัดการกับ exception

```
>>> anumber = int(input("Please enter an integer "))
Please enter an integer -23
>>> print(math.sqrt(anumber))
Traceback (most recent call last):
  File "<pyshell#102>", line 1, in <module>
    print(math.sqrt(anumber))
ValueError: math domain error
>>>
```

- จัดการกับ exception
  - **try** : ทำภายใต้ส่วน try
  - **except** : เกิด exception โยนมาส่วนนี้

```
>>> try:
    print(math.sqrt(anumber))
except:
    print("Bad Value for square root")
    print("Using absolute value instead")
    print(math.sqrt(abs(anumber)))
```

```
Bad Value for square root
Using absolute value instead
4.79583152331
>>>
```



# Function

- def ชื่อ function ( ส่งค่า parameter)
- คืนค่าด้วย return

```
>>> def square(n):  
...     return n**2  
...  
>>> square(3)  
9  
>>> square(square(3))  
81  
>>>
```

# Class

- ชื่อ class
- Constructor (ตัวสร้าง)
  - `__init__` เป็นชื่อ method
  - `self` เป็นตัวแปรที่อ้างถึงตัวมันเอง
- Method พื้นฐาน
  - ขึ้นต้นและลงท้ายด้วย `__`
- Method อื่นๆ

# ตัวอย่าง Fraction

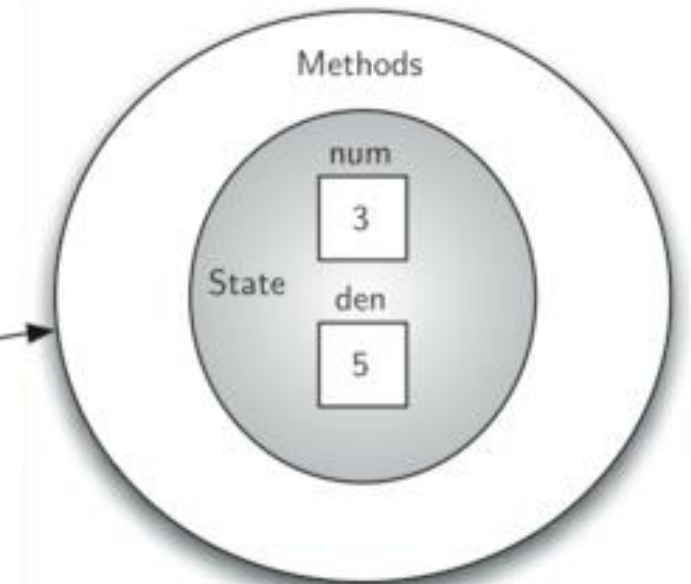
constructor

```
1 class Fraction:
2
3     def __init__(self, top, bottom):
4
5         self.num = top
6         self.den = bottom
7
8 myfraction = Fraction(3,5)
9 print(myfraction)
```

<\_\_main\_\_.Fraction object>

Print ไม่ออก

myfraction



# ตัวอย่าง Fraction

```
1 class Fraction:
2
3     def __init__(self, top, bottom):
4         self.num = top
5         self.den = bottom
6
7     def show(self):
8         print(self.num, "/", self.den)
9
10 myfraction = Fraction(3, 5)
11 myfraction.show()
12 print(myfraction)
```

สร้าง method show มาแสดงผล

```
3 / 5
<__main__.Fraction object>
```

แสดงผลแบบนี้ใช้งานยาก

# ตัวอย่าง fraction

- ใช้การสร้าง method พื้นฐาน (override)
  - `__str__`

```
1 class Fraction:
2
3     def __init__(self, top, bottom):
4         self.num = top
5         self.den = bottom
6
7     def __str__(self):
8         return str(self.num) + "/" + str(self.den)
9
10 myfraction = Fraction(3,5)
11 print(myfraction)
```

3/5

# ตัวอย่าง fraction

- `__add__` บวก
- `__eq__` เปรียบเทียบ

```
def __add__(self, otherfraction):  
    newnum = self.num*otherfraction.den + self.den*otherfraction.num  
    newden = self.den * otherfraction.den  
    common = gcd(newnum,newden)  
    return Fraction(newnum//common,newden//common)
```

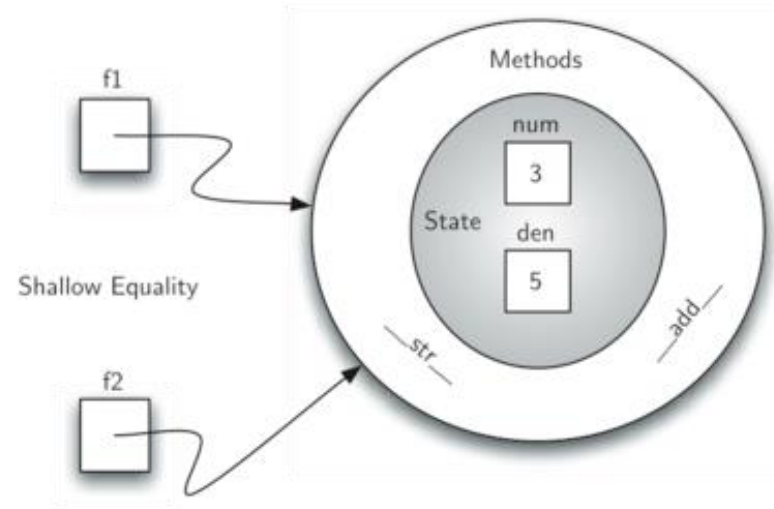
`gcd` เขียนเองไว้หาหรม.

```
>>> f1=Fraction(1,4)  
>>> f2=Fraction(1,2)  
>>> f3=f1+f2  
>>> print(f3)  
3/4  
>>>
```

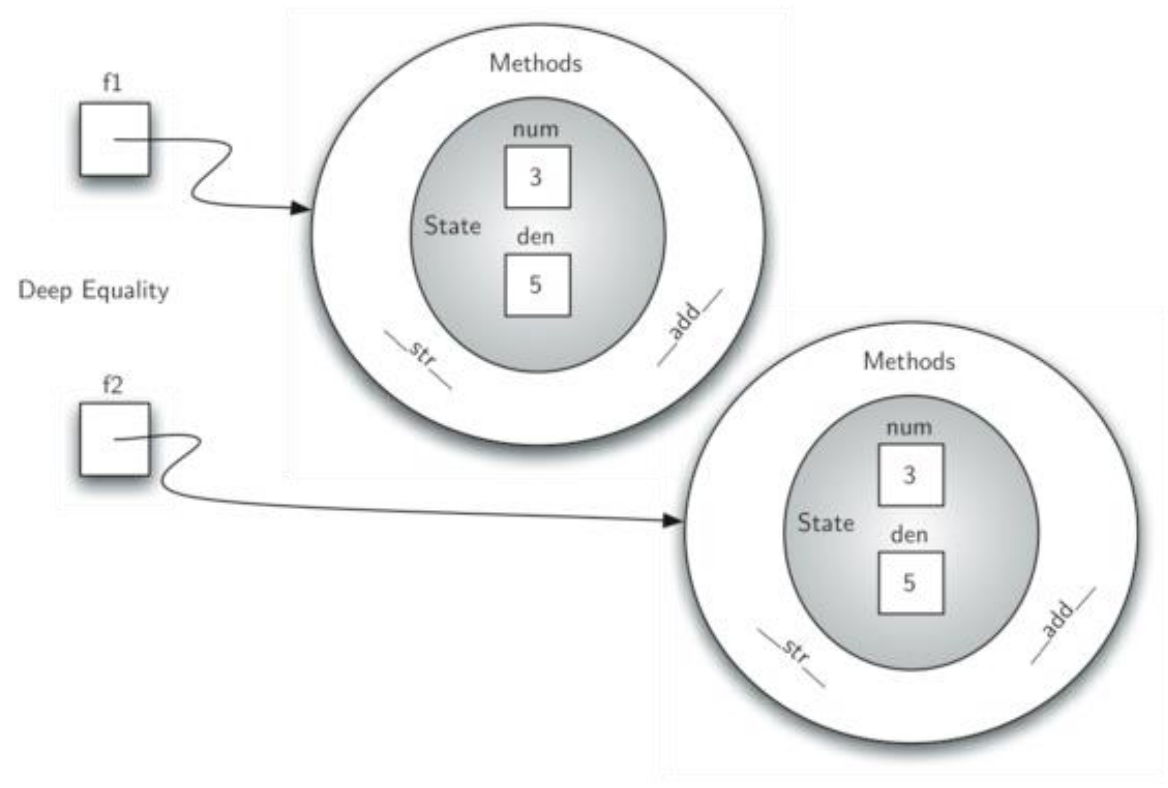
```
def __eq__(self, other):  
    firstnum = self.num * other.den  
    secondnum = other.num * self.den  
  
    return firstnum == secondnum
```

# ตัวอย่าง fraction

- ทำไมต้องสร้าง `__eq__`



เปรียบเทียบ reference



## แบบฝึกหัด

- <http://interactivepython.org/runestone/static/pythonds/Introduction/ProgrammingExercises.html>
- <http://codingbat.com/python>
- <http://10.31.26.54/grader/main/list>