








Somkiat

Home
17











Somkiat Puisungnoen

Update Info
View Activity Log
10+
...


Timeline
About
Friends
2,941
Photos
More



When did you attend Ubon Ratchathani University?


...
14 Pending Items


Status


Photo/Video

Live Video

Life Event


What's on your mind?


Public


Post


Intro


Software Craftsmanship



Software Practitioner at สยามชำนาญกิจ พ.ศ. 2556


Agile Practitioner and Technical at SPRINT3r


Software analyst at TARAD.com


Software Developer at True Corporation


Former Software Engineer at Opendream



Somkiat Puisungnoen

1 hr · Twitter ·

พรั่งนี้ไป share เรื่อง NoSQL ที่มหาวิทยาลัย


Like


Comment


Share




Nuttachot Dusitanont, Chitpong Few Wuttanan and 50 others

บริษัท สยามชำนาญกิจ จำกัด และเพื่อนพ้องน้องพี่



somkiat.cc



Somkiat

Home



Page

Messages

Notifications

Insights

Publishing Tools

Settings

Help



somkiat.cc

@somkiat.cc

Home

About

Events

Photos

Likes

Videos

Posts

Services



Liked



Following



Share



+ Add a Button



Write something...



Personal Blog

Page Tips

See All



What's a Boosted Post?

A boosted post is the easiest way to reach more people on Facebook.

Installation

[GO](#)[Socialize](#)[Sign In](#)[About](#)[Downloads](#)[Documentation](#)[Community](#)[Success Stories](#)[News](#)[Events](#)

Download the latest version for Mac OS X

[Download Python 3.6.2](#)[Download Python 2.7.13](#)

Wondering which version to use? [Here's more about the difference between Python 2 and 3.](#)

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)

Want to help test development versions of Python? [Pre-releases](#)



<https://www.python.org/>



First Question ?



<https://wiki.python.org/moin/Python2orPython3>



Overall Picture

Python 2.x is **Legacy**

Python 3.x is the **present and future**



For beginner ?

You should learn Python 2

More documents

More libraries and frameworks



But for this course

Use Python 3

More semantically correct
Support newer features



Style Guide

<https://www.python.org/dev/peps/pep-0008/>

```
long_function_name()  
    Classname  
        _private
```



Basic Python



Interactive mode

\$python

```
Python 3.6.2 (v3.6.2:5fd33b5926, Jul 16 2017, 20:11:06)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello Python")
Hello Python
>>> █
```



Interactive mode

```
>>help(list)
```

```
>>dir(list)
```



Online Help

1. Google with “python <name>”
2. Official Python Doc
3. Stackoverflow
4. Quora



Script mode

```
$python <file>.py
```



Hello world

```
print('Hello World')  
print('Hello' + ' ' + 'World')  
print("Hello World")  
print("Hello" + " " + "World")
```

print is a function in Python 3



Indentation

Python standard is **4 spacebars**



Indentation

Python 3 disallows mixing the use of tabs and spaces for indentation.



Comment

Start with #

Multi-line comment with '''



Reserved words

and	exec	not	else
as	finally	or	lambda
assert	for	pass	yield
break	from	print	except
class	global	raise	del
continue	if	return	in
def	import	try	with



Waiting for the user

```
input("Please any key to continue")
```



Variable and Type



Variable and Type

Completely Object Oriented

Every variable in Python is an object

Not statically typed



Standard data types

Number

String

List

Tuple

Dictionary



Data type conversion

Function	Description
<code>int(x [,base])</code>	Converts x to an integer
<code>float(x)</code>	Converts x to a floating-point number
<code>complex(real [,imag])</code>	Creates a complex number
<code>str(x)</code>	Converts object x to a string representation
<code>tuple(s)</code>	Converts s to a tuple
<code>list(s)</code>	Converts s to a list
<code>set(s)</code>	Converts s to a set
<code>dict(d)</code>	Creates a dictionary
<code>chr(x)</code>	Converts an integer to a character



Basic Operator



Number

int

float

complex



Int

In Python 3 => Unlimited size



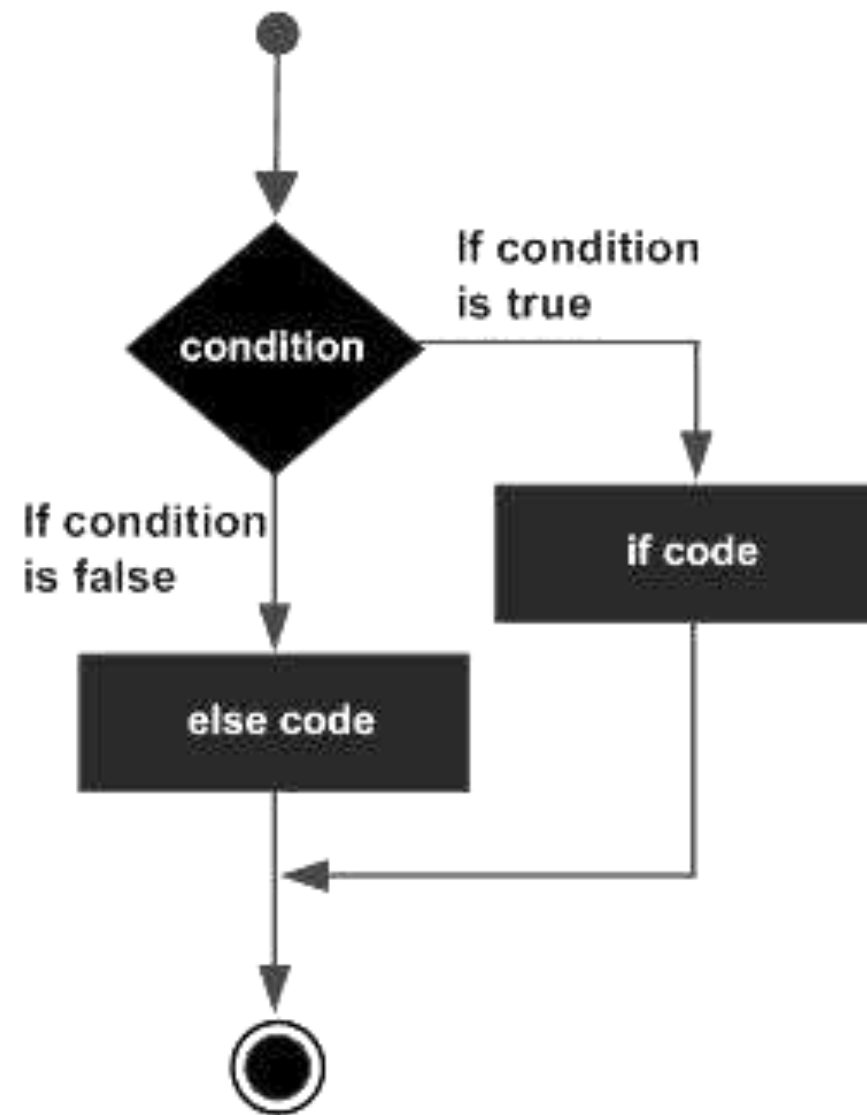
Operation for number

Operation	Symbol	Example
Power (exponentiation)	**	$5 ** 2 == 25$
Multiplication	*	$2 * 3 == 6$
Division	/	$14 / 3 == 4.6666666666666667$
Integer Division	//	$14 / 3 == 4$
Remainder (modulo)	%	$14 \% 3 == 2$
Addition	+	$1 + 2 == 3$
Substraction	-	$4 - 3 == 1$



Decision making





```
score = int(input("Enter score: "))
if score >= 80:
    print("A")
elif score >= 70:
    print("B")
elif score >= 60:
    print("C")
elif score >= 50:
    print("D")
else:
    print("F")
```



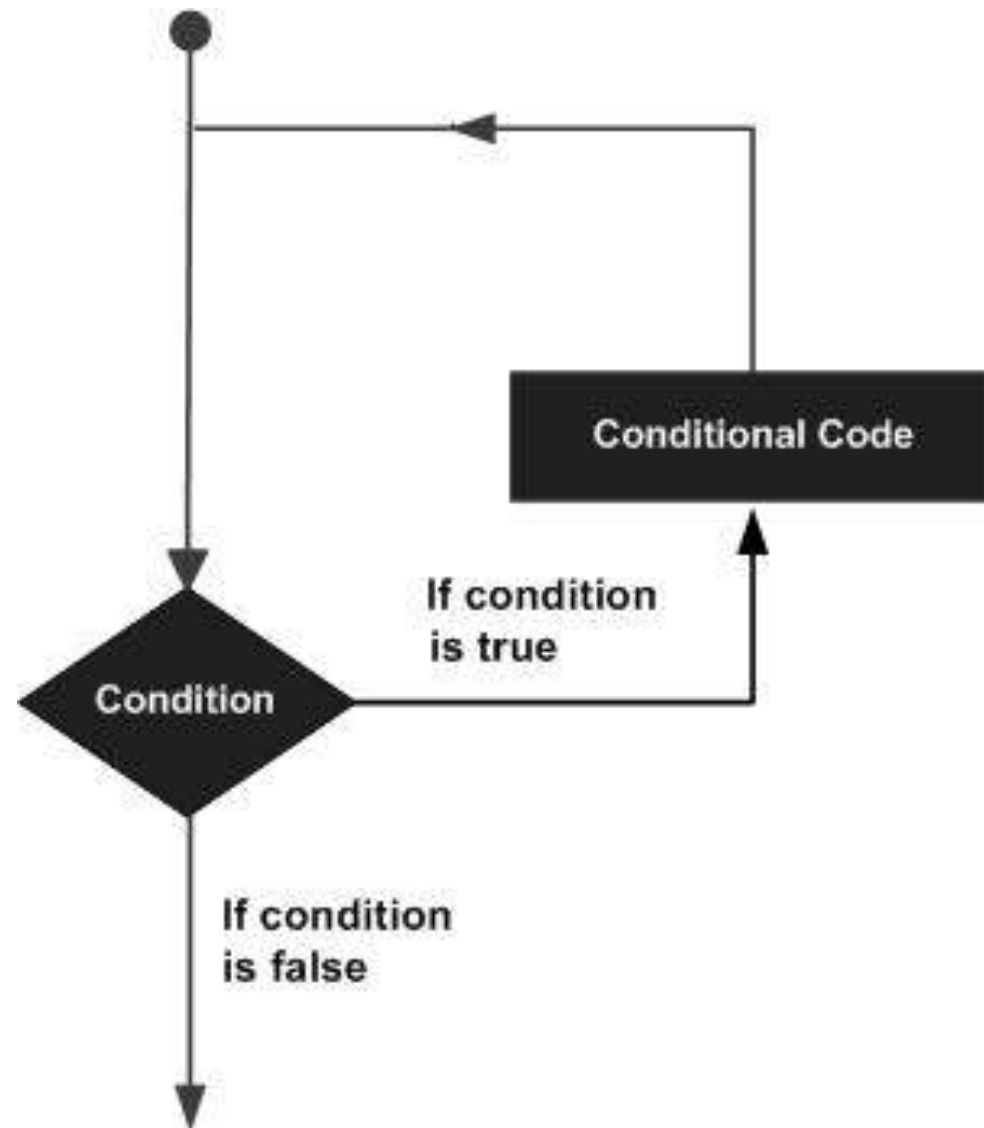
More operator

in operator
is operator
not operator



Loop





Loop types

While loop

For loop

Nested loop



For loop

```
datas = range(0, 9)
```

```
for data in datas:  
    print(data)
```



For loop

```
names = ["Tom", "Mike", "Ko"]  
for name in names:  
    print(name)
```

```
for index in range(len(names)):  
    print(names[index])
```



String



Operation for String

Operation	Symbol	Example
Repetition	*	"i" * 5 == "iiiii"
Concatenation	+	"Hello, " + "World" == "Hello, World"
Slice	[]	data[0]
Range slice	[:]	data[0:5]
Membership	in not in	"i" in "i love you" == True



List



Create List

```
empty = []  
numbers = [1, 2, 3, 4, 5]  
string = ["H", "e", "l", "l"]  
mix = [1, 2, "three", True]  
  
for data in mix:  
    print(type(data))
```



Access List

```
mix = [1, 2, "three", True]  
print(mix[0])  
print(mix[1])  
print(mix[1:3])
```



Operation for List

Operation	Symbol	Example
Length	len()	
Concatenation	+	$[1] + [2] = [1, 2]$
Repetition	*	$[1] * 3 = [1, 1, 1]$
Membership	in	$1 \text{ in } [1, 2, 3] = \text{True}$
Range slice	[:]	



Tuple



Tuple

Immutable list

Can not delete or update data



Create Tuple

```
empty = ()  
countries = ("Thai", "Indo", "China")  
print(countries)
```



Dictionary



Dictionary

Key:Value data structure

Keys are unique

Keys must be of immutable data type

Values can be of any type



Create Dictionary

```
empty = {}  
employee = {"firstname": "Somkiat",  
            "lastname": "Pui",  
            "age": 30}  
  
print(employee)
```



Duplication key ?

```
data = {"Key1": "first", "Key1": "second"}  
print(data)
```



Immutable key ?

Immutable key

```
data = { 1: "first",  
        "two": "second",  
        (1, 2): "third" }  
print(data)
```

Mutable key

```
data = { [1, 2]: "first" }
```



Function



Function

Block of reusable code

Single responsibility

All parameters are passed by reference



Create function

```
def say_hi(name):  
    result = "Hello " + name  
    return result  
  
print(say_hi("Somkiat"))
```



Pass by reference

```
def try_to_change(data):  
    data[2] = 300  
    return
```

```
input = [0, 0, 0]  
print("Before ", input)  
try_to_change(input)  
print("After ", input)
```



Function arguments

Required arguments

Keyword arguments

Default arguments

Variable-length arguments



Keyword arguments

```
def say_hi( name, age ):  
    print("Hello %s, age = %d" %(name, age))
```

```
say_hi(name = "somkiat", age = 30)
```

```
say_hi(age = 30, name = "somkiat")
```



Default arguments

```
def say_hi2( name, age = 20 ):
    print("Hello %s, age = %d" %(name, age))
```

```
say_hi2(name = "somkiat")
say_hi2("somkiat")
```



Variable-length arguments

```
def sum(*numbers):  
    result = 0  
    for number in numbers:  
        result = result + number  
    return result
```

```
sum()  
sum(1)  
sum(1, 2)  
sum(1, 2, 3)  
sum(1, 2, 3, 4)  
sum(1, 2, 3, 4, 5)
```



Workshop with factorial



RecursionError

RecursionError: maximum recursion depth
exceeded in comparison



RecursionError (1,000)

```
import sys

def factorial(n):
    if n <= 1:
        return 1
    return n * factorial(n-1)

# Max of recursion in python
print(sys.getrecursionlimit())
sys.setrecursionlimit(15000)
print(factorial(5000))
```



Modules



Modules

Organize your Python code
Grouping related code into a module
Easy to understand and use



Create new module

```
#file hello.py
```

```
def say_hi():  
    print("Say hi")
```

```
if __name__ == "__main__":  
    say_hi()
```



Using module with import

```
#file caller.py
```

```
from hello import *
```

```
say_hi()
```



Locating Modules

Current directory

PYTHONPATH

Python installation directory



Locating Modules

```
import sys  
from pprint import pprint  
  
pprint(sys.path)
```



Workshop module



Workshop module

second_module

hello.py
+ say_hi()



Module structure

```
.
├── caller.py
├── second_module
│   ├── __init__.py
│   └── hello.py
```



1. create folder second_module



2. create file hello.py

```
def say_hi():  
    print("From hello 1")
```



3. create caller.py outside the module folder

```
import sub01
```

```
sub01.say_hi()
```



4. create `__init__.py` inside module

```
from .hello import say_hi
```



Exception handling



Class

