ไพธอน

คุณสมบัติเด่น

- การแปลภาษาของภาษาไพธอนเป็นแบบอินเทอร์พรีเตอร์ประมวลผลไปทีละบรรทัด
- ไวยากรณ์อ่านง่าย โดยใช้การย่อหน้าแทน ทำให้สามารถอ่านโปรแกรมที่เขียนได้ง่าย
- ภาษาไพธอน และชุดของไลบารีสนับสนุนการประมวลผลทางด้านวิทยาศาสตร์และวิศวกรรม-ศาสตร์ได้อย่างมีประสิทธิภาพ
- ไพธอนมีไลบรารีที่สนับสนุนงานด้านการสร้างภาพกราฟฟิก และการประมวลผลภาพ (Image processing) มากมาย
- ไพธอนเตรียมไลบรารีสำหรับสนับสนุนการเขียนโปรแกรมทางด้านปัญญาประดิษฐ์
- ไพธอนมีความสามารถในการจัดการหน่วยความจำอัตโนมัติ (Garbage collection)

Keyword

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	Import	pass	
break	except	in	raise	

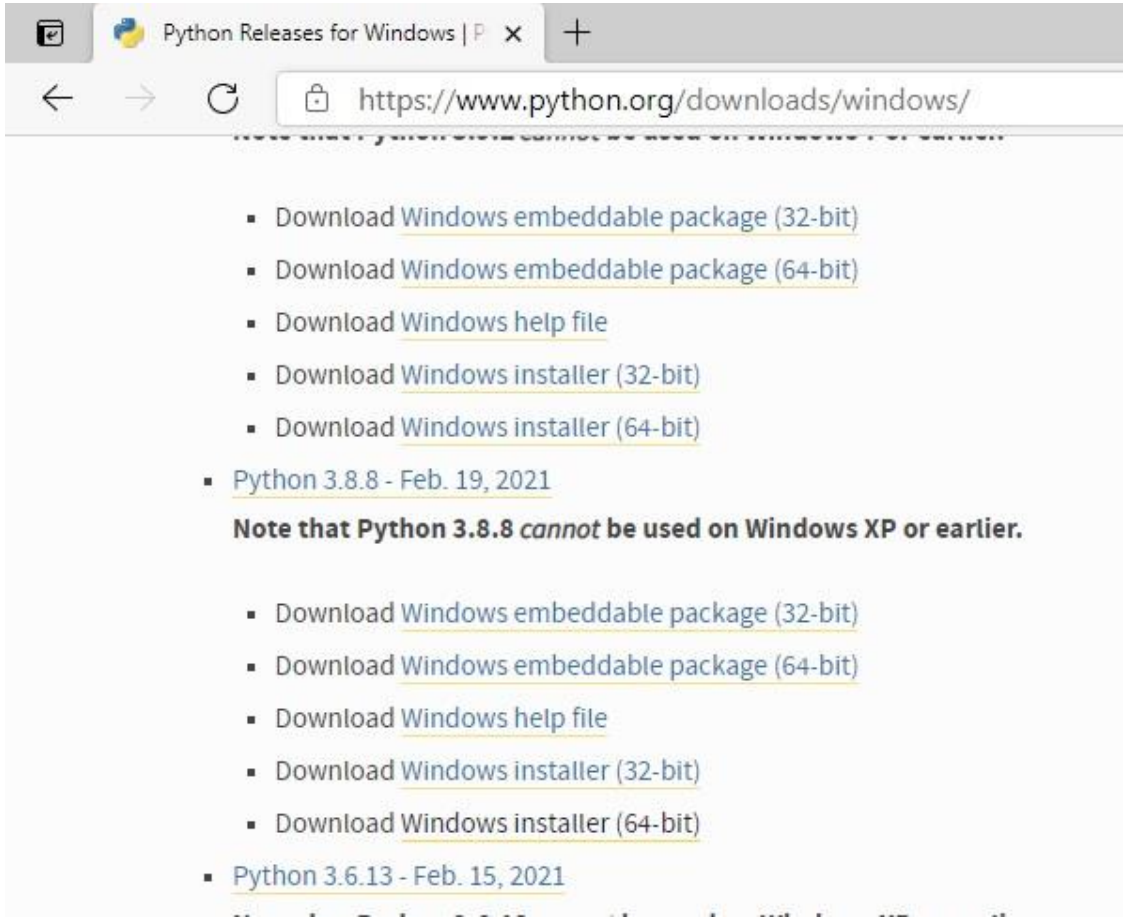
หลักการเบื้องต้น

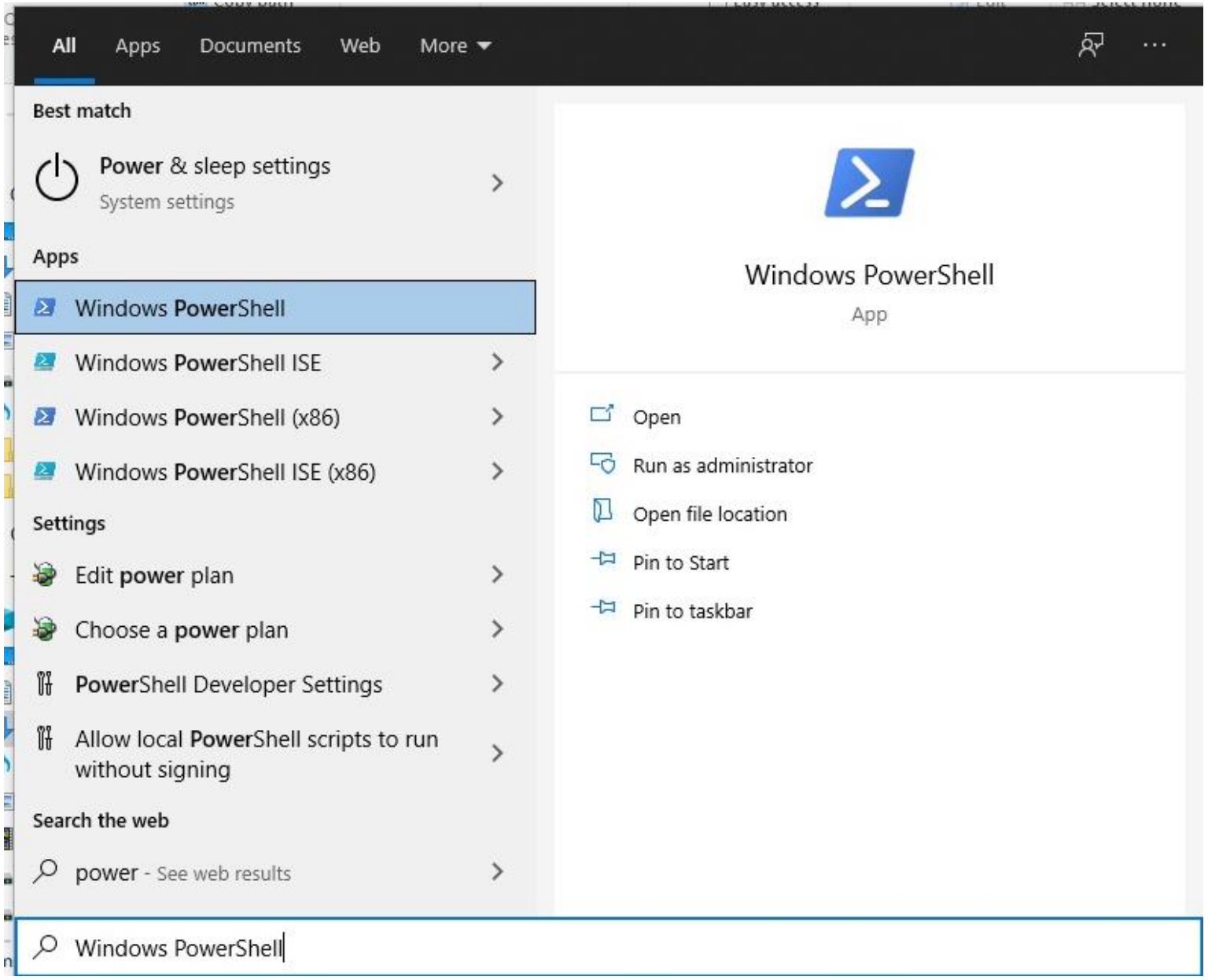
- **การเยื้องย่อหน้า** ผู้เขียนควรใช้ช่องว่าง (space) หรือ แท็บ (tabs) อย่างใดอย่างหนึ่งเท่านั้น

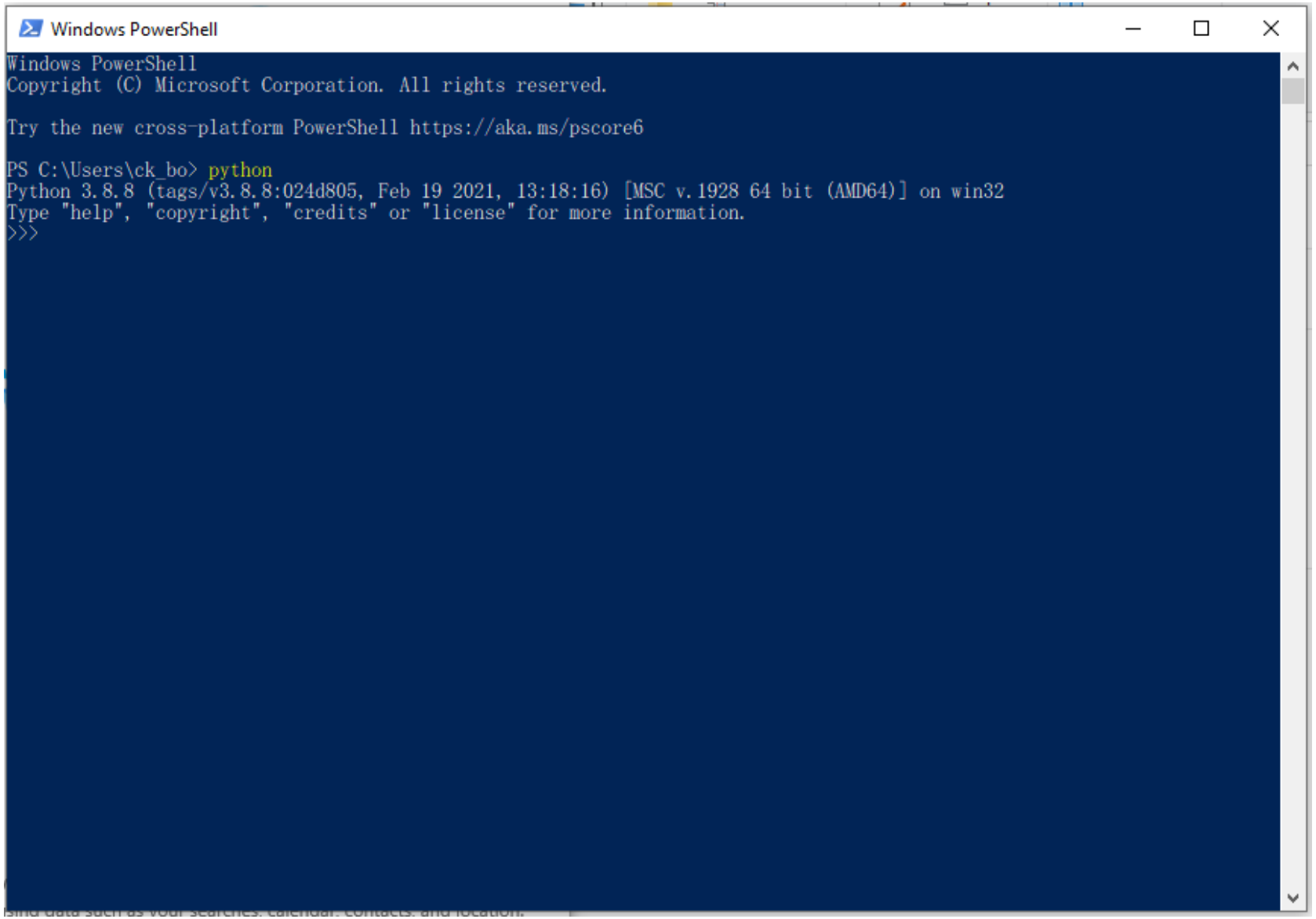
```
def on_relay(relay):
    relay.value(0)
def off_relay(relay):
    relay.value(1)
```

- **เขียนคำสั่งเกิน 1 บรรทัด** เครื่องหมาย \ ตามด้วยการกดขึ้นบรรทัดใหม่(Enter)
- **เครื่องหมายอัญประกาศ** ไพธอนใช้เครื่องหมายอัญประกาศเดี่ยว (' , Single quote) และอัญประกาศ" (", Double quote)
- **คอมเมนต์(Comment)** สำหรับบรรทัดที่ไม่ต้องการให้ทำงาน ให้ใช้เครื่องหมาย #

Install Python for Windows

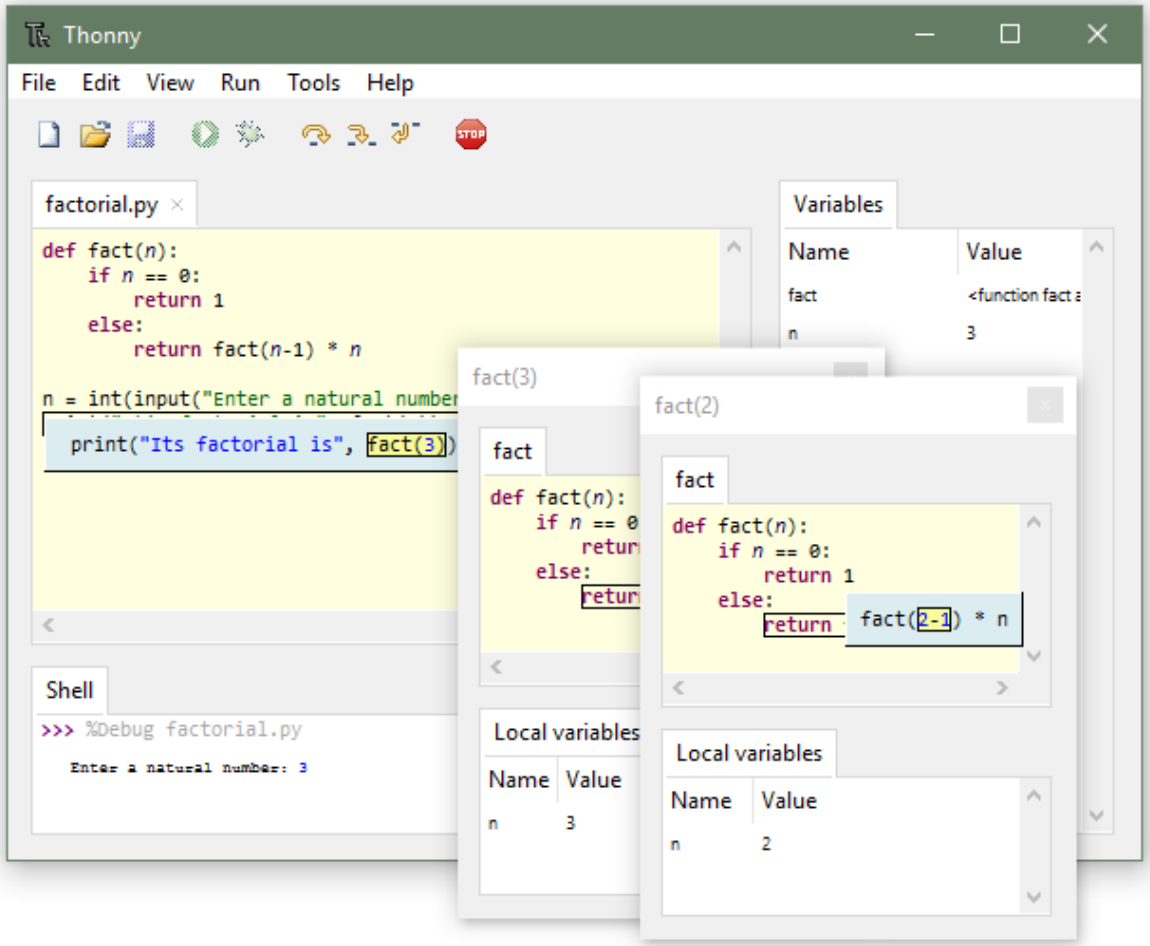



All
Apps
Documents
Web
More
Best match
Power & sleep settings
System settings
Apps
Windows PowerShell
Windows PowerShell ISE
Windows PowerShell (x86)
Windows PowerShell ISE (x86)
Settings
Edit power plan
Choose a power plan
PowerShell Developer Settings
Allow local PowerShell scripts to run without signing
Search the web
power - See web results
Windows PowerShell
App
Open
Run as administrator
Open file location
Pin to Start
Pin to taskbar
Windows PowerShell


Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\ck_bo> python
Python 3.8.8 (tags/v3.8.8:024d805, Feb 19 2021, 13:18:16) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>

https://www.python.org/downloads/

เอดิเตอร์ (https://thonny.org/)



```
PS C:\Users\CK> python
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import platform
>>> platform.machine()
'AMD64'
>>> platform.version()
'10.0.19041'
>>> platform.platform()
'Windows-10-10.0.19041-SP0'
>>> platform.uname()
uname_result(system='Windows', node='DESKTOP-36SPGMQ', release='10', version='10.0.19041', machine='AMD64', processor='I
ntel64 Family 6 Model 158 Stepping 10, GenuineIntel')
>>> platform.system()
'Windows'
>>> platform.processor()
'Intel64 Family 6 Model 158 Stepping 10, GenuineIntel'
>>>
```

การทำงานพื้นฐาน

ประเภท	โอเปอร์เรเตอร์	การใช้งาน	ตัวอย่าง
คณิตศาสตร์	+	บวก	x+10
	-	ลบ	x-10
	*	คูณ	x*10
	/	หาร	x/10
	%	เศษ	x%1
เปรียบเทียบ	==	เท่ากับ	A == B
	!=	ไม่เท่ากับ	A != B
	>	มากกว่า	A > B
	<	น้อยกว่า	A < B
	>=	มากกว่าหรือเท่ากับ	A >= B
	<=	น้อยกว่าหรือเท่ากับ	A <= B
ตรรกศาสตร์	and	และ	A and B
	or	หรือ	A or B

ชนิดข้อมูล (Data types)

- **Integer** เลขจานวนเต็ม (Integers)

```
>>> 1 + 2 + 3
6
>>> 1 + 100 + 25 + 3
129
>>>
```

- **ประเภท Float** ตัวเลขทศนิยม หรือจำนวนจริง (Floating-Point numbers)

```
>>> 3.6 + 5.0 + 25/4
14.85
```

- ประเภท Boolean

```
>>> x = 10
>>> y = 20
>>> x == y
False
>>> Z = True
>>> print(Z)
True
```

- ประเภท String ข้อมูลชนิดตัวอักษร

```
>>> myname = "CK"
>>> print(myname)
CK
```

- **ประเภท list** เก็บข้อมูลได้หลายจำนวนภายในตัวแปรเดียว

```
>>> member = ["chatchai", "wipada", "wisarut"]
>>> print(member)
[chatchai, wipada, wisarut]
>>> print(member[0])
chatchai
```

- **ประเภท Tuple** มีลักษณะคล้ายกับลิสต์สำหรับเก็บข้อมูลที่มีค่าคงที่และไม่สามารถแก้ไขค่าได้

```
>>> nick = ("ck", "june", "opp")
>>> print(nick)
(ck, june, opp)
>>> print(nick[1])
june
```

- **ประเภท Dictionary** เก็บข้อมูลเป็นคู่ ระหว่าง คีย์(Key) กับข้อมูล (Value) โดยที่การกำหนดค่าคีย์ต้องไม่ซ้ำกัน

```
>>> nickname = {"chatchai": "ck", "wipada" : "june", "wisarut" : "opp"}
>>> print(nickname)
{wipada: june, wisarut: opp, chatchai: ck}
>>> print(nickname["wipada"])
june
```

ข้อสังเกต

- List [....]
- Tuple (...)
- Dictionary { ... }

ทดสอบ

ให้เวลา 20 นาที ทดสอบการเขียนการทำงานต่อไปนี้

1. ให้กำหนดตัวแปรประเภท Tuble 1 ตัว ประกอบตัวสมาชิกมีค่าเป็น 1, 3 , 5, และ 9 จากนั้นให้หาค่าผลบวกจากสมาชิกทั้งหมด

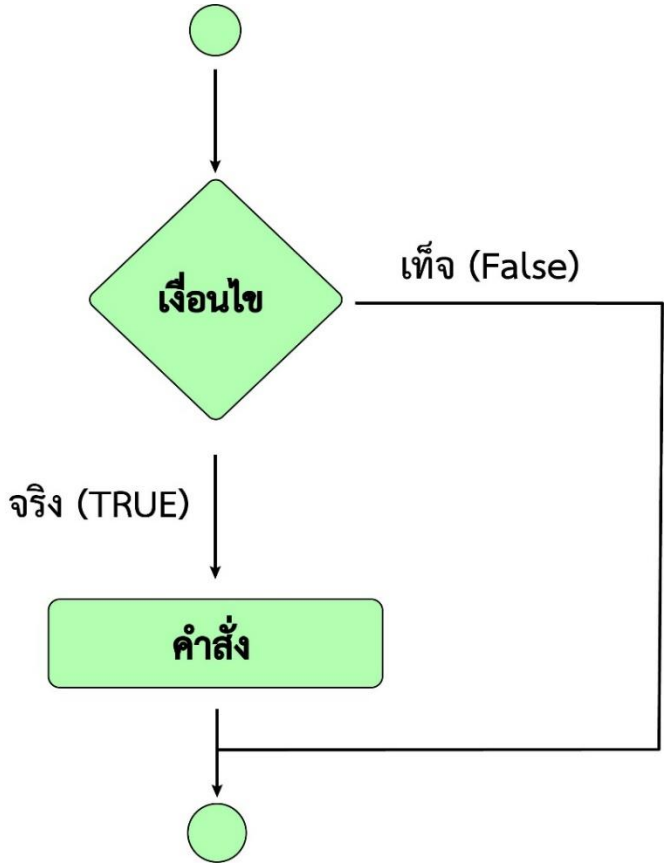
2. ทดสอบการสร้างตัวแปรประเภท Dictionary ประกอบด้วย

```
>>> dic = {"a":1,"b":2,"c":3,“d":4,“f":5}

>>> sum = dic["a"] + dic[“f"]
```

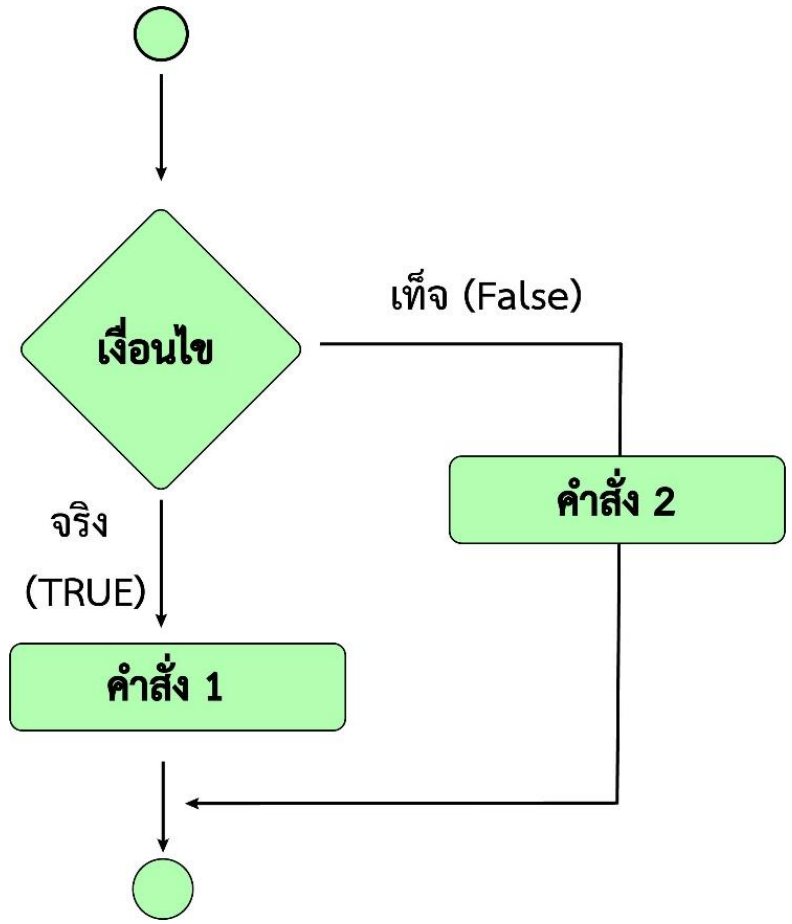
ค่า sum มีค่าเท่ากับเท่าใด

การทำงานแบบเงื่อนไข `if`, `if/else`, `if/elif/else`



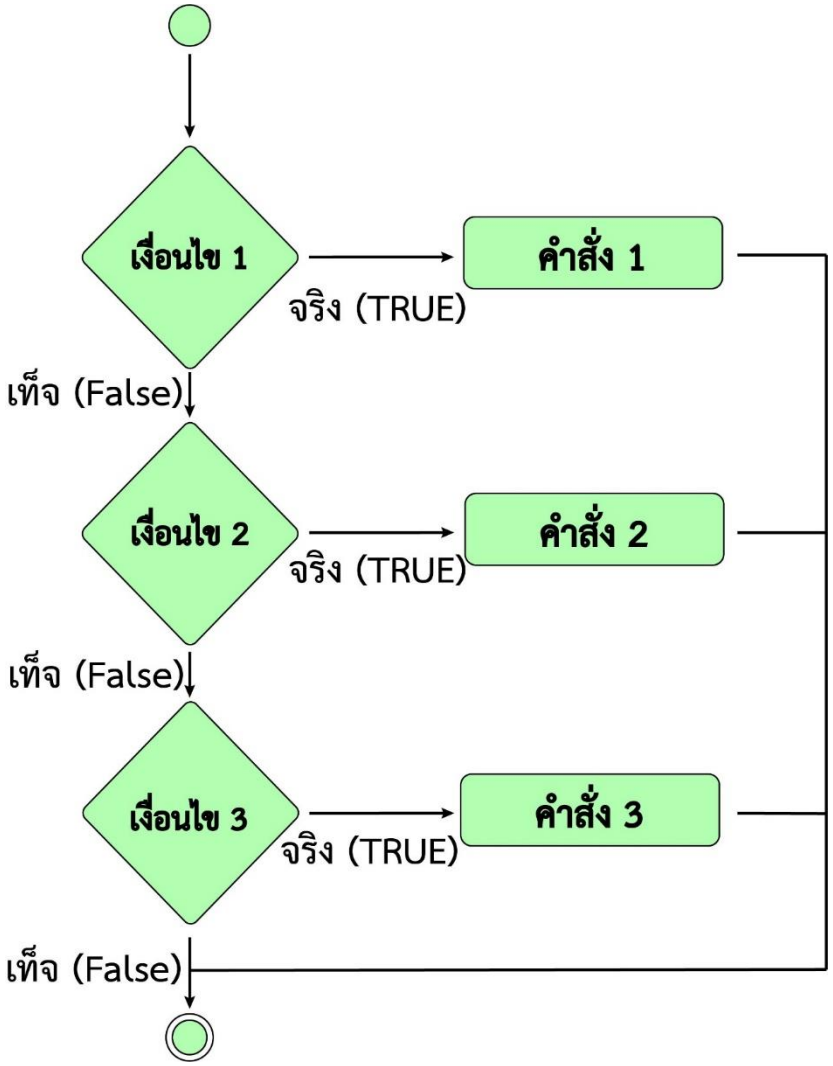

```
if a == 0:
    print ("zero!“)
```

การทำงานแบบเงื่อนไข if/else, if/elif/else




```
if a == 0:
    print ("zero!“)
else:
    print (“non zero”)
```

การทำงานแบบเงื่อนไข if/else, if/elif/else




```
if a == 0:
    print "zero!"
elif a < 0:
    print "negative!"
else:
    print "positive!"
```

การป้อนค่าจาก keyboard

```
>>> y = input("Please Input Current Temperature :")
Please Input Current Temperature :35
>>> print(y)
35
```

แบบฝึกหัด

ให้เขียนโปรแกรมเพื่อรับค่าจาก keyboard โดยกำหนดเงื่อนไข ดังนี้

a. ถ้าค่าที่รับมามีค่า มากกว่า 10 ไม่ทำอะไร ถ้าน้อยกว่า พิมพ์คำว่า OK

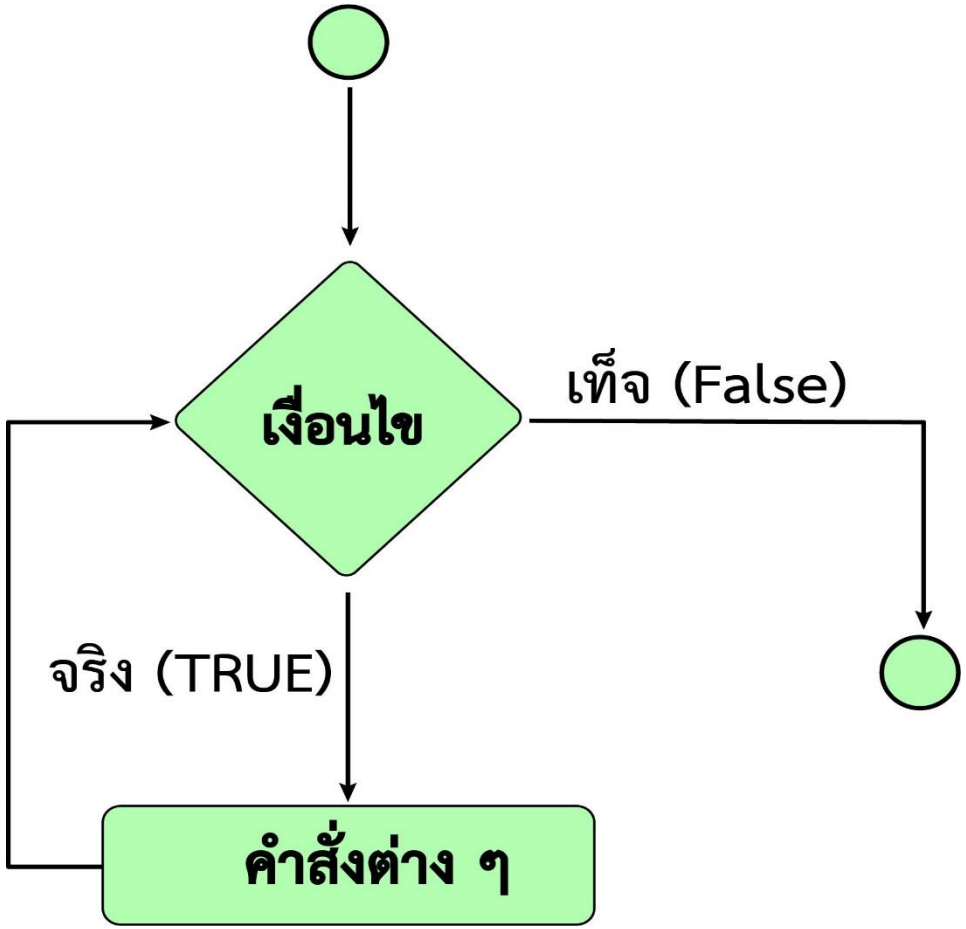
b. ถ้าค่าที่รับมามีค่า มากกว่า 10 พิมพ์ Yes, น้อยกว่า พิมพ์ NO

c. ถ้าค่าที่รับมามีค่า 0 – 10 พิมพ์ 10, 11 – 20 พิมพ์ 20 อย่างอื่น พิมพ์ 0

การทำงานซ้ำหลายครั้ง

while loops

 # do something



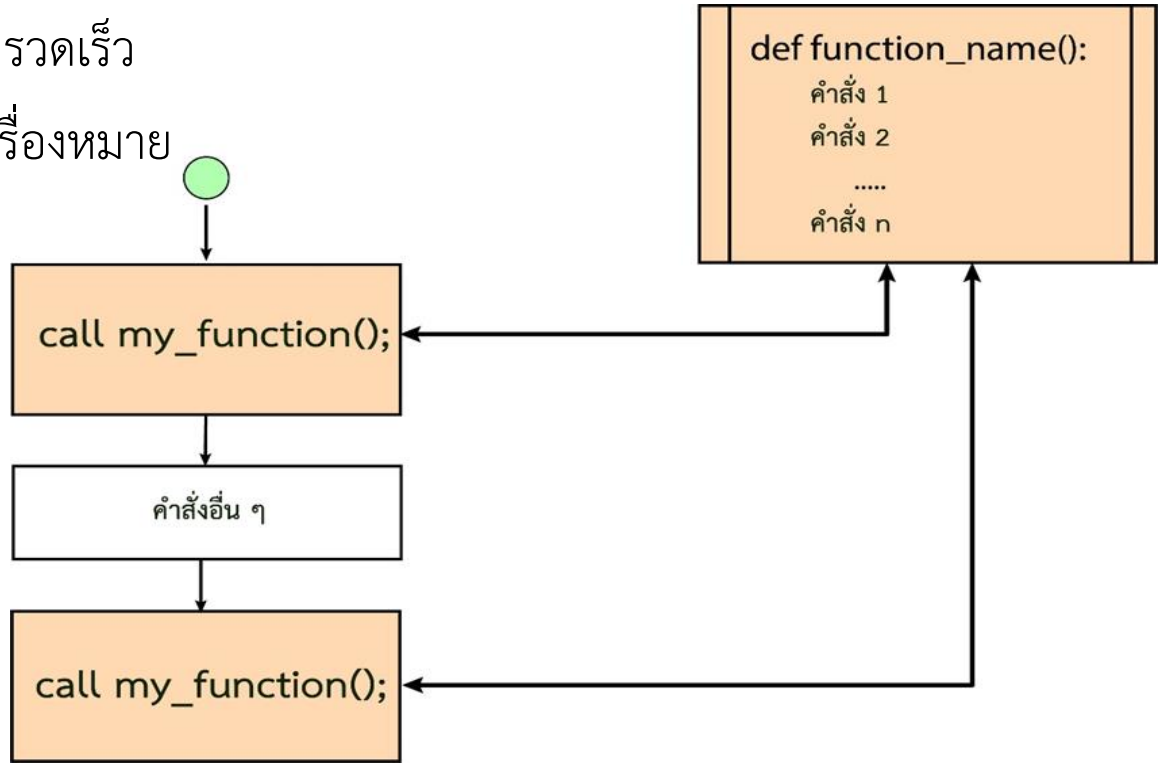

```
a = 10
while a > 0:
    print(a)
    a = a - 1
```

ให้ทดสอบ run พร้อมแสดงผลที่ได้

ฟังก์ชัน

- โปรแกรมย่อยหรืองานย่อยๆ (Sub-program) ภายในโปรแกรมขนาดใหญ่หรือ
- ประโยชน์ของฟังก์ชัน
    1. ช่วยลดคำสั่ง ที่ซ้ำซ้อนกันในโปรแกรม
    2. สามารถปรับปรุงและแก้ไขโปรแกรมได้อย่างรวดเร็ว
    3. ช่วยทำให้โปรแกรมมีความกะทัดรัด ทำให้เข้าใจง่ายและรวดเร็ว
- การประกาศฟังก์ชันใช้คำว่า def นำหน้าตามด้วยชื่อฟังก์ชันและเครื่องหมายวงเล็บ ():

```
def foo(x):
    y = 10 * x +2
    return y
```



ทดสอบการใช้งานฟังก์ชัน

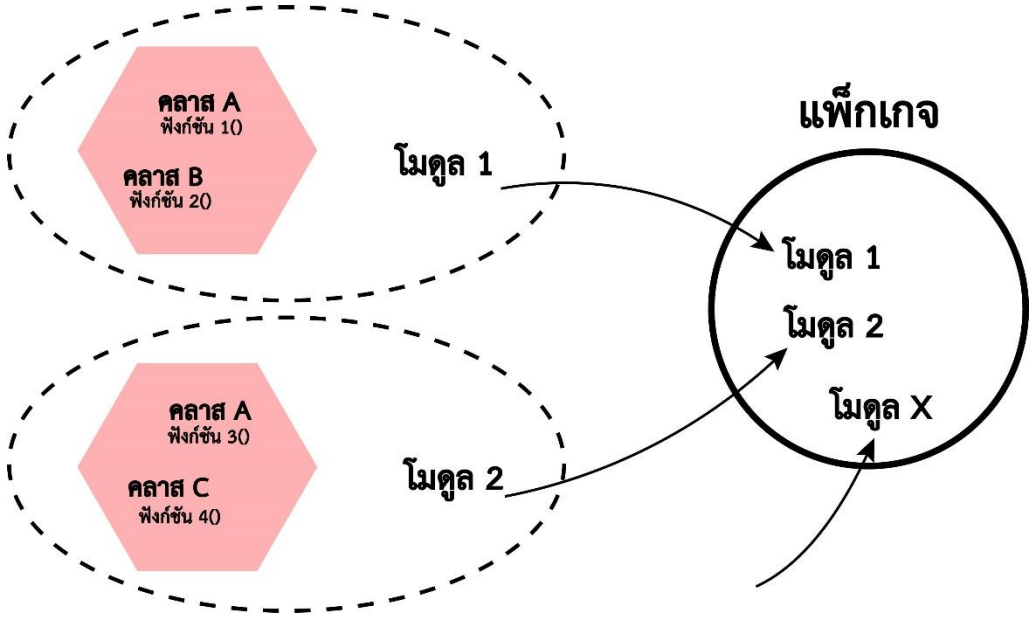
```
def my_compute(x):
      y = 10*x + 2
      print(y)

my_compute(12.5)
```

- ให้เขียนฟังก์ชัน เพื่อคำนวณค่า y = x^2 + z^2

แสดงตัวอย่างการเรียนใช้

โมดูล



- การใช้คำสั่ง import เรียกใช้งานฟังก์ชันและตัวแปรทั้งหมดในโมดูลเข้ามาทางานในโปรแกรม สามารถเรียกใช้มากกว่าหนึ่งโมดูลพร้อมกัน

```
import math [, module2, module3 ..., moduleN]

print(math.sqrt(2.0))
```

- การใช้คำสั่ง from module import function เรียกฟังก์ชันหรือตัวแปรเฉพาะที่ต้องการมาใช้งานเท่านั้น

```
from math import sqrt

print(sqrt(2.0))
```

- การใช้คำสั่ง from module import * เป็นการเรียกฟังก์ชันทั้งหมดของโมดูลนั้น

```
from math import *
print sqrt(2.0)
```

- การใช้คำสั่ง import moduleName as newName เป็นการเรียกใช้โมดูลชื่อเดิมไปเป็นชื่อโมดูล

```
import pandas as pd
data_in = pd.read_csv('temperature.csv')
```

การแสดงผล

- การใช้งานรูปแบบ %

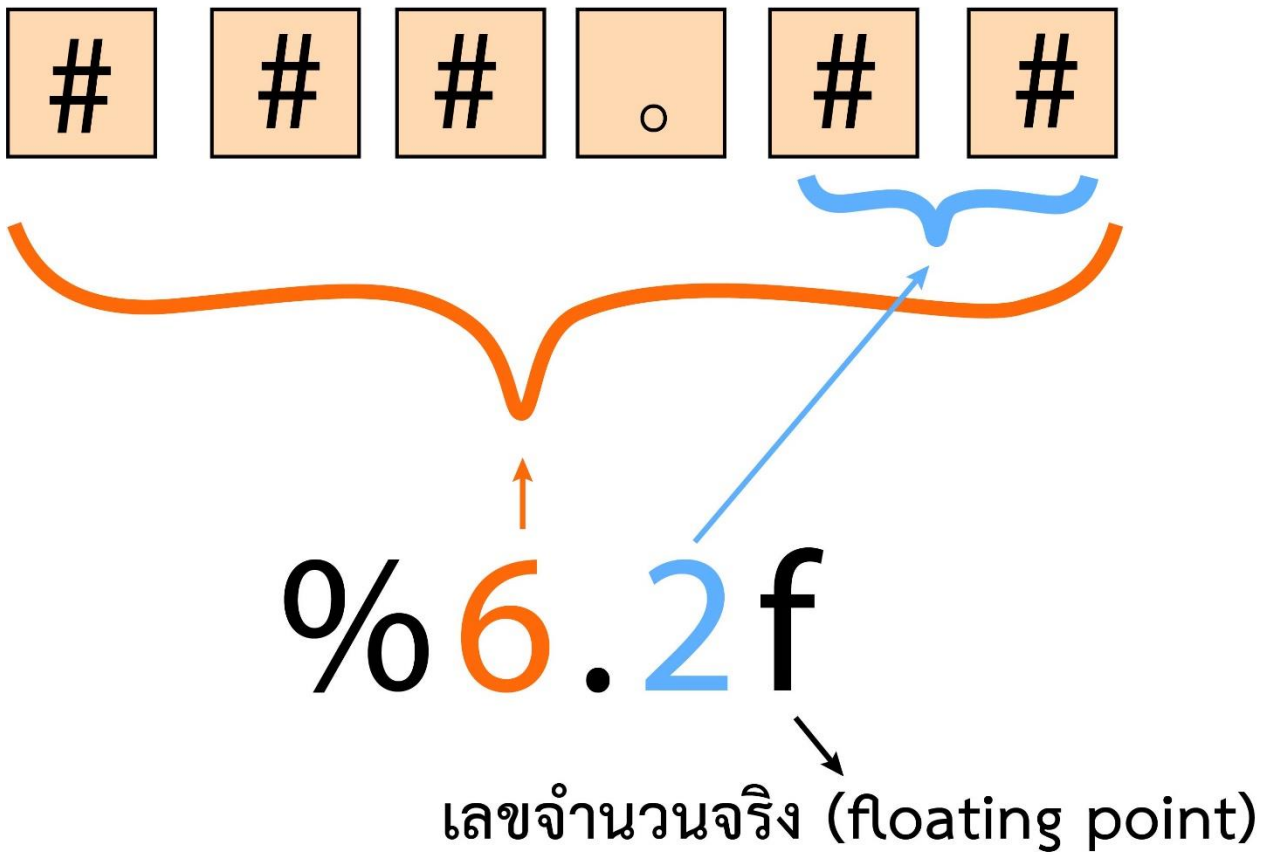
รูปแบบ	ประเภท	การใช้งาน
%d, %i	integer	แสดงจำนวนเต็ม
%u	unsigned int	เลขจำนวนเต็มแบบไม่มีเครื่องหมาย
%f	float	แสดงจำนวนทศนิยม
%c	character	ตัวอักษา
%s	string	แสดงผลแบบสตริง
%%	-	แสดงเครื่องหมาย %

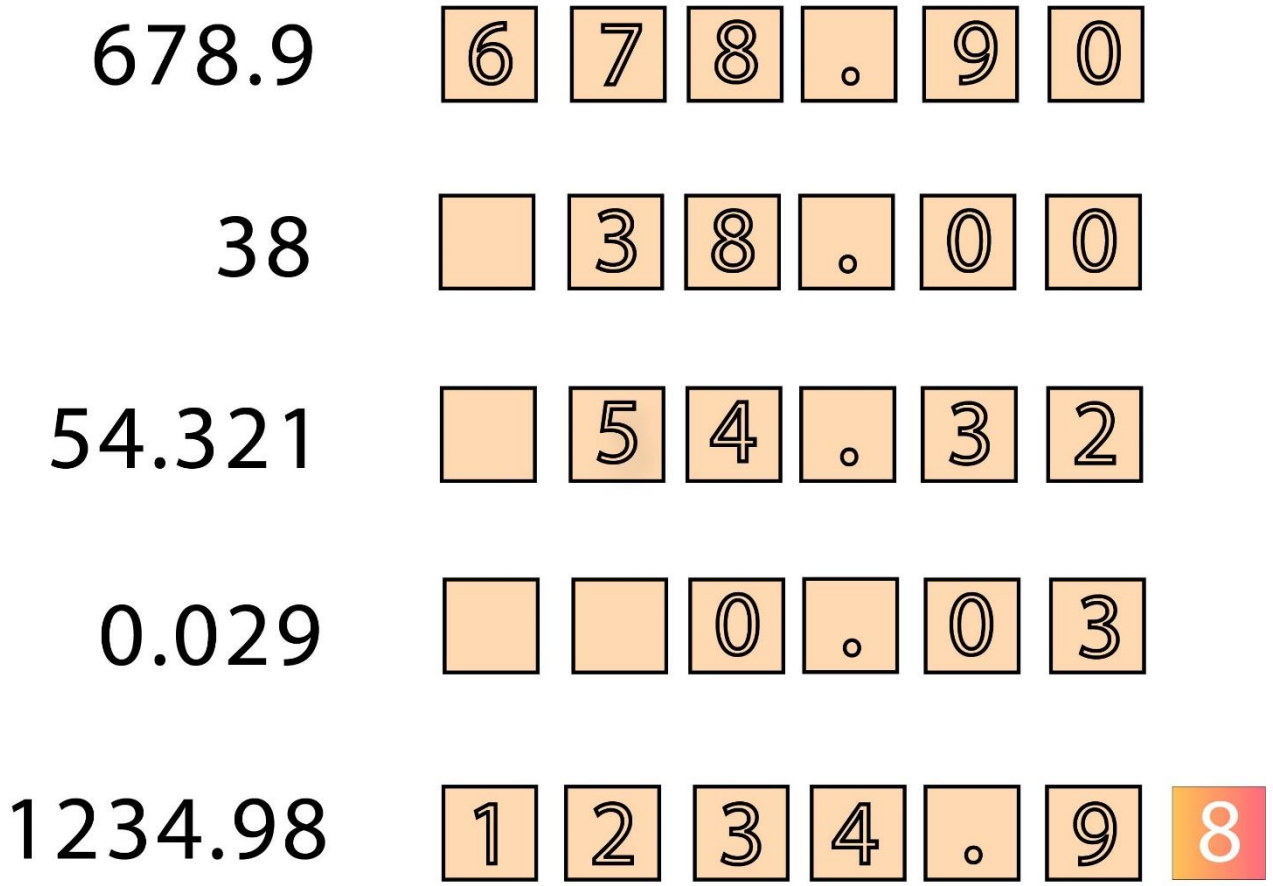
ข้อควรระวัง

- ข้อควรระวังการเลือกชนิดที่ถูกต้อง

```
>>> x= 65
>>> print ("Temperature %d" %x)
Temperature 65
>>> print ("Temperature %c" %x)
Temperature A
```

จัดรูปแบบการแสดงผล






```
>>> Huminity = 75.56
>>> print("Huminity %5.1f, %5.2f, %5.3f" %(Huminity, Huminity, Huminity))
Huminity 75.6, 75.56, 75.560
>>> print("Huminity %8.1f, %8.2f, %8.3f" %(Huminity, Huminity, Huminity))
Huminity   75.6,  75.56, 75.560
```

- การใช้งานรูปแบบ .format()
- การแสดงการใช้ format เป็นการจัดรูปแบบสตริง

```
>>> Temperature = 25.0
>>> Huminity = 75.5
>>> print ("Temperature {}, Huminity {}" .format(Temperature, Huminity) )
Temperature 25.0, Huminity 75.5
```

- กำหนดตัวแปรล่วงหน้า เพื่อความสะดวก

```
>>> DHT_Value = "Temperature {}, Huminity{}"
>>> print (DHT_Value.format(Temperature, Huminity) )
Temperature 25.0, Huminity75.5
```

```
>>> Temperature = 25.0
>>> Huminity = 75.5
>>> print ("Temperature {}, Huminity {}" .format(Temperature, Huminity) )
Temperature 25.0, Huminity 75.5
>>> print ("Temperature {0}, Huminity {1}" .format(Temperature, Huminity) )
Temperature 25.0, Huminity 75.5
>>> print ("Temperature {1}, Huminity {0}" .format(Temperature, Huminity) )
Temperature 75.5, Huminity 25.0
>>> print ("Temperature {0:5.2f}, Huminity {0:5.2f}" .format(Temperature, Huminity) )
Temperature 25.00, Huminity 25.00
```

เกิดอะไรขึ้นครับ ? ค่าที่ควรแสดง?

ทดสอบการแสดงผล

- สมมติให้กำหนดรูปแบบการแสดงผลต่อไปนี้
- >>> Humidity = 75.56
- >>> print("Humidity %6.1f, %6.2f, %6.3f" %(Humidity, Humidity, Humidity))
- >>> print("Humidity %10.1f, %10.2f, %10.3f" %(Humidity, Humidity, Humidity))
- >>> print ("Humidity {0:5.3f} {0:6.3f} {0:10.3f} " .format(Humidity))