

# .Bootcamp - Basic Python สรุป

```
>>> print('Hello world')
```

- คำสั่ง print คือคำสั่งที่ใช้ในการแสดงผล เพื่อแสดงคำว่า Hello world

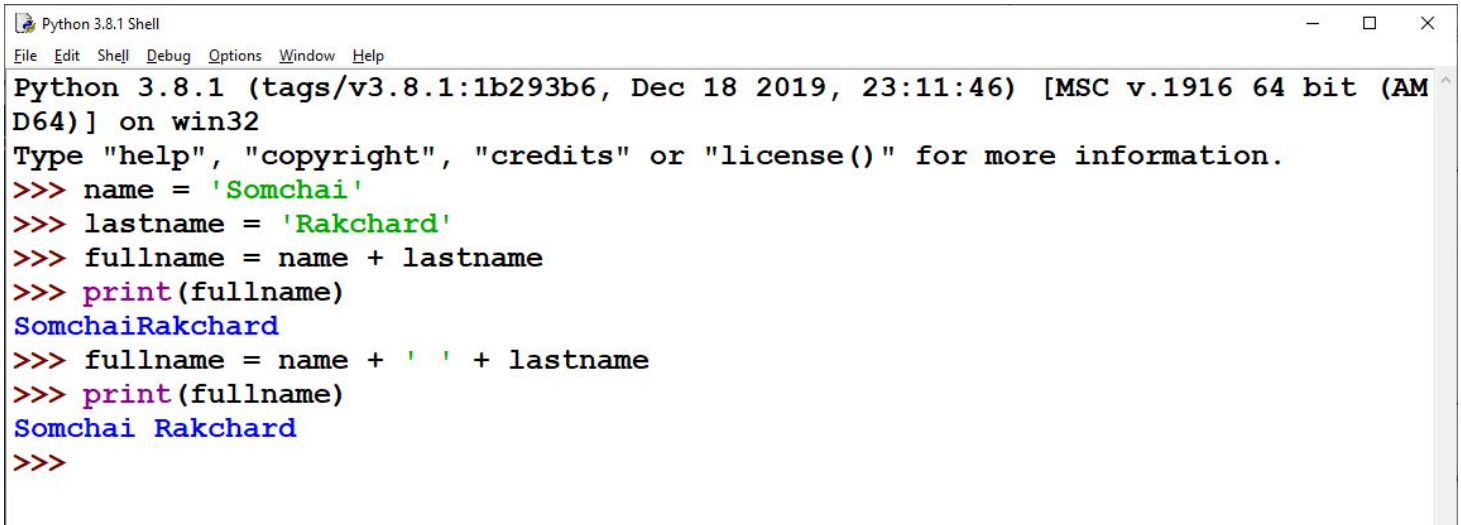
```
>>> name = 'Somchai'
```

- เป็นการประกาศค่าของตัวแปร name ให้มีค่าเท่ากับ Somchai

```
>>> fullname = name + lastname
```

- เราสามารถสร้างตัวแปรใหม่มารับค่าจากตัวแปรตัวอื่น ๆ ได้ ในกรณีนี้คือการ นำค่า name มารวมกับ lastname

ตัวอย่าง :

A screenshot of a Python 3.8.1 Shell window. The window title is "Python 3.8.1 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area shows the following output and code:

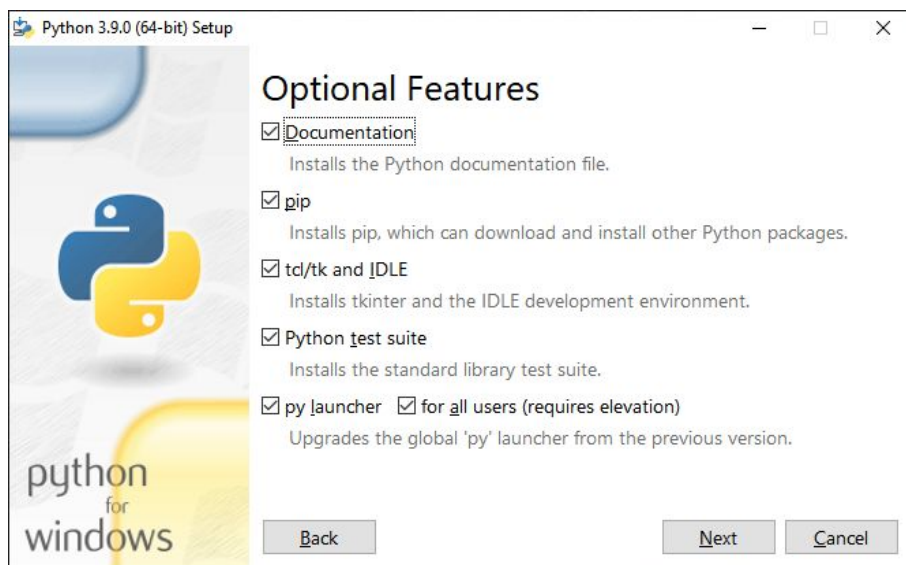
```
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> name = 'Somchai'
>>> lastname = 'Rakchard'
>>> fullname = name + lastname
>>> print(fullname)
SomchaiRakchard
>>> fullname = name + ' ' + lastname
>>> print(fullname)
Somchai Rakchard
>>>
```

เราต้องกำหนดรูปแบบของค่าภายในตัวแปร เพื่อความถูกต้องเหมาะสมของค่าที่ต้องการตั้งภาพด้านบน

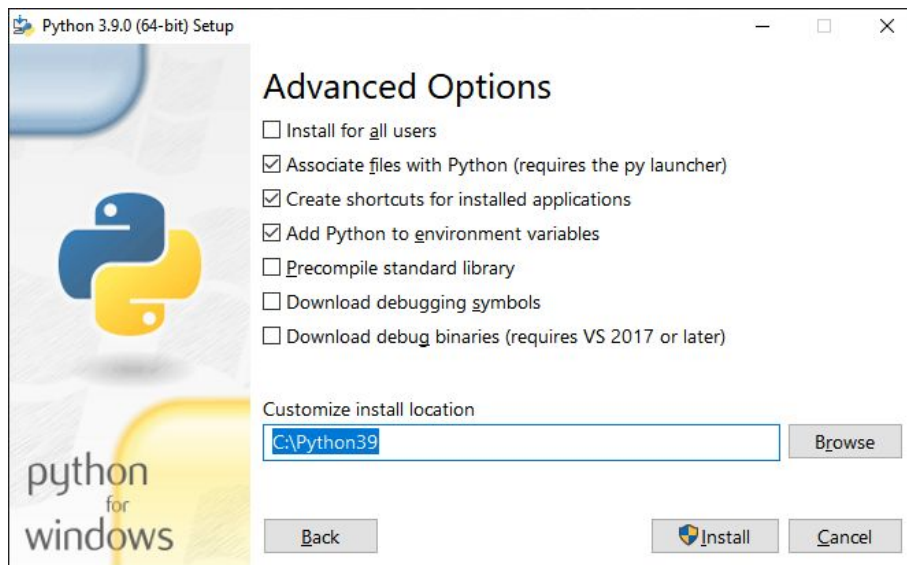
# วิธีการติดตั้ง Python (Python Installation)



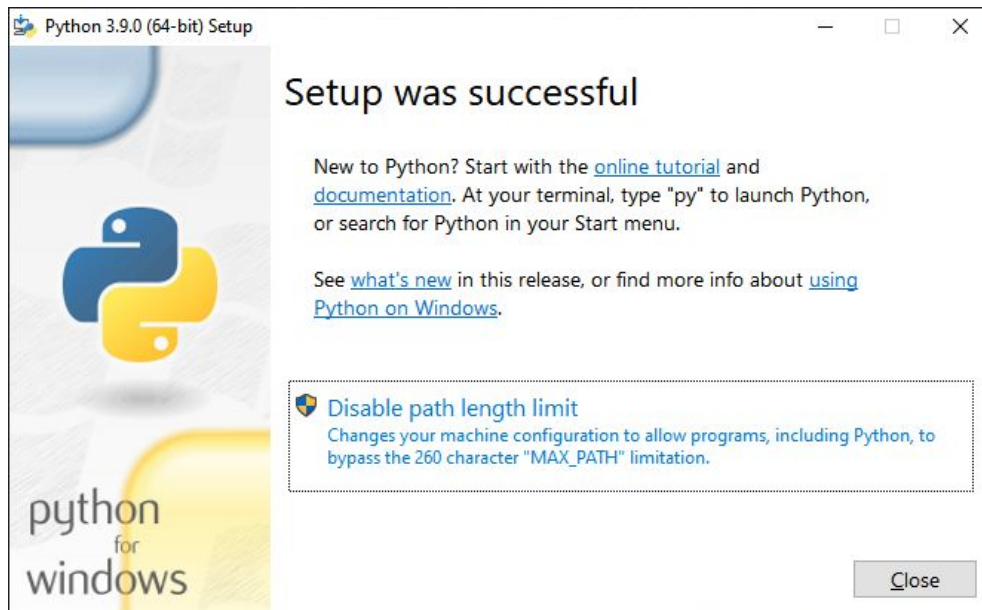
คลิกที่ Add Python 3.9 to PATH เพื่อติดตั้งแล้ว เลือก Customize installation



กด Next



ที่ Customize install location ให้แก้เป็น C:\Python39 หรือ path ที่ต้องการ แล้วกด Install



เสร็จเรียบร้อยแล้ว ;)

## Python Mode:

- Shell

มีเครื่องหมาย >>>

กดปุ่ม enter เพื่อรันคำสั่ง

- Editor (IDE)

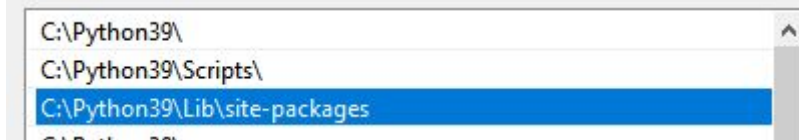
พิมพ์ได้หลายบรรทัด

พิมพ์จบแล้วกด Run > Run Module

# วิธีการตั้งค่าการใช้งานของ Python (Python Setting Path Environment)

ลิงค์สไลด์ ในการติดตั้ง Path Environment ของไพทอน :

<https://docs.google.com/presentation/d/1TodHI53b3WibQ6fzBmk2HAOkxypvGSKJqrpVF4rB1Os/edit?usp=sharing>



3 Path ที่จำเป็นจะต้องนำไปใส่ไว้ใน Path Environment ของเครื่อง

```
Select Command Prompt
Microsoft Windows [Version 10.0.18362.1139]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Uncle Engineer>pip -V
pip 20.2.3 from c:\python39\lib\site-packages\pip (python 3.9)

C:\Users\Uncle Engineer>python -V
Python 3.9.0

C:\Users\Uncle Engineer>pip list
Package      Version
-----
pip          20.2.3
setuptools   49.2.1
WARNING: You are using pip version 20.2.3; however, version 20.2.4 is available.
You should consider upgrading via the 'c:\python39\python.exe -m pip install --upgrade pip' command.

C:\Users\Uncle Engineer>
```

หลังจากเราเพิ่ม Path แล้วลองตรวจสอบเวอร์ชันของ pip ด้วยคำสั่ง pip -V ดูเลย

Documentation: googletrans

<https://py-googletrans.readthedocs.io/en/latest/>

สรุป Python (ที่ทสี่ี่ยว):

<https://drive.google.com/file/d/1-CbsQyAYjgZnPw76FOfgITffPPiZDZ7i/view?usp=sharing>

คำสั่งติดตั้งแบบยาวในกรณีที่ใช้ pip ไม่ได้

C:\Python39\python.exe -m pip install googletrans

# โปรแกรมแปลภาษา

```
# GUITranslator.py
from tkinter import *
#จากไลบรารีชื่อ tkinter, * คือให้ดึงความสามารถหลักมาทั้งหมด
from tkinter import ttk # ttk is theme of tk
### -----Google Translate-----
from googletrans import Translator
translator = Translator() #สร้างฟังก์ชันแปลภาษา
GUI = Tk() #สร้างหน้าต่างหลัก
GUI.geometry('500x300') #กว้าง x สูง
GUI.title('โปรแกรมแปลภาษา by Uncle Engineer')
# -----config-----
FONT = ('Angsana New',15)
# -----Label-----
L = ttk.Label(GUI,text='กรุณากรอกคำศัพท์ที่ต้องการแปล',font=FONT)
L.pack()
# -----Entry (ช่องกรอกข้อความ)-----
v_vocab = StringVar() #กล่องเก็บข้อความ
E1 = ttk.Entry(GUI,textvariable = v_vocab,font=FONT,width=40)
E1.pack(pady=20)
# -----Button (ปุ่มแปล)-----
def Translate():
    vocab = v_vocab.get() #.get คือให้แสดงข้อมูลออกมา
    meaning = translator.translate(vocab,dest='th')
    print( vocab + ' : ' + meaning.text)
    print( meaning.pronunciation)
    v_result.set(vocab + ' : ' + meaning.text) #ส่งไว้ใน gui
B1 = ttk.Button(GUI,text='Translate',command=Translate) #สร้างปุ่มขึ้นมา
B1.pack(ipadx=20,ipady=10) # show ปุ่มขึ้นมาวางจากบนลงล่าง
# -----Label-----
L = ttk.Label(GUI,text='คำแปล',font=FONT)
L.pack()
# -----Result-----
v_result = StringVar() #นี่คือกล่องสำหรับเก็บคำแปล
FONT2 = ('Angsana New',20)
R1 = ttk.Label(GUI,textvariable=v_result, font=FONT2, foreground='green')
R1.pack()
GUI.mainloop() #ทำให้โปรแกรมรันได้ตลอดเวลาจนกว่าจะปิด (บรรทัดสุดท้าย)
```

## สรุปจาก Code :

การอธิบายต่อไปนี้จะเราจะไม่เน้นบรรทัดที่เป็นคำอธิบายในโปรแกรม (คอมเมนต์ #...)

```
from tkinter import *
#จากไลบรารีชื่อ tkinter, * คือให้ดึงความสามารถหลักมาทั้งหมด
from tkinter import ttk # ttk is theme of tk
### -----Google Translate-----
from googletrans import Translator
```

ส่วนนี้จะเป็นการเรียก Lib ที่เราจะใช้งาน โดย tkinter จะเป็น Lib ที่มาจาก Python อยู่แล้ว ไม่ต้อง pip install ต่างจาก googletrans ที่เราต้องเข้า cmd เพื่อ pip install googletrans ก่อน

```
translator = Translator() #สร้างฟังก์ชันแปลภาษา
GUI = Tk() #สร้างหน้าต่างหลัก
GUI.geometry('500x300') #กว้าง x สูง
GUI.title('โปรแกรมแปลภาษา by Uncle Engineer')
```

ส่วนนี้เป็นการประกาศตัวแปร เพื่อตั้งชื่อเรียกใช้ฟังก์ชันต่าง ๆ  
translator จะถูกเรียกใช้แทน Translator() จาก Lib googletrans  
GUI จะถูกเรียกใช้แทน Tk() จาก Lib tkinter  
แล้วประกาศขนาดหน้า GUI กว้าง 500 px และ สูง 300 px  
ต่อมาตั้งชื่อโปรแกรมที่แสดงบน Taskbar ด้านบน GUI



```

FONT = ('Angsana New',15)
# -----Label-----
L = ttk.Label(GUI,text='กรุณารอกคำศัพท์ที่ต้องการแปล',font=FONT)
L.pack()
# -----Entry (ช่องกรอกข้อความ) -----
v_vocab = StringVar() #กล่องเก็บข้อความ
E1 = ttk.Entry(GUI,textvariable = v_vocab,font=FONT,width=40)
E1.pack(pady=20)
# -----Button (ปุ่มแปล) -----
def Translate():
    vocab = v_vocab.get() #.get คือให้แสดงผลออกมา
    meaning = translator.translate(vocab,dest='th')
    print( vocab + ' : ' + meaning.text)
    print( meaning.pronunciation)
    v_result.set(vocab + ' : ' + meaning.text) #ส่งไขว้ใน gui
B1 = ttk.Button(GUI,text='Translate',command=Translate) #สร้างปุ่มขึ้นมา
B1.pack(ipadx=20,ipady=10) # show ปุ่มขึ้นมาวางจากบนลงล่าง
# -----Label-----
L = ttk.Label(GUI,text='คำแปล',font=FONT)
L.pack()
# -----Result-----
v_result = StringVar() #นี่คือกล่องสำหรับเก็บคำแปล
FONT2 = ('Angsana New',20)
R1 = ttk.Label(GUI,textvariable=v_result, font=FONT2, foreground='green')
R1.pack()
GUI.mainloop() #ทำให้โปรแกรมรันได้ตลอดเวลาจนกว่าจะปิด (บรรทัดสุดท้าย)

```

กำหนดรูปแบบอักษร (font) ลงในตัวแปร FONT ('Angsana New',15) เพื่อให้ง่ายต่อการเรียกใช้งาน

สร้าง Label ซึ่งจะเป็นข้อความที่แสดงบน GUI โดยที่ข้อความที่แสดงอยู่ที่ค่าใน text="..." โดยตามตัวอย่างจะมีข้อความว่า "กรุณารอกคำศัพท์ที่ต้องการแปล" แล้วให้แสดงบน GUI ด้วยคำสั่ง .pack()

สร้าง v\_vocab ให้เป็น StringVar() เพื่อเป็นค่าข้อมูลที่เป็นข้อความจากการกรอกข้อมูลในช่องกรอก E1 แล้วทำการแสดงบน GUI ด้วยคำสั่ง .pack()

สร้างฟังก์ชัน Translate() เพื่อดึงข้อมูลจากช่องกรอก เมื่อมีการกดปุ่ม Translate แล้วทำการแสดงข้อความที่เป็นคำแปล และคำอ่าน (ในตัวอย่างเป็นภาษาไทย dest='th') ใน shell ด้วยคำสั่ง print() และแสดงบน GUI ด้วยคำสั่ง .set() เช่น Cat : แมว

สร้างปุ่ม Translate เมื่อกดจะทำการรันฟังก์ชัน Translate() ผ่านคำสั่ง command=Translate แล้วทำการแสดงบนหน้า GUI ด้วยคำสั่ง .pack()

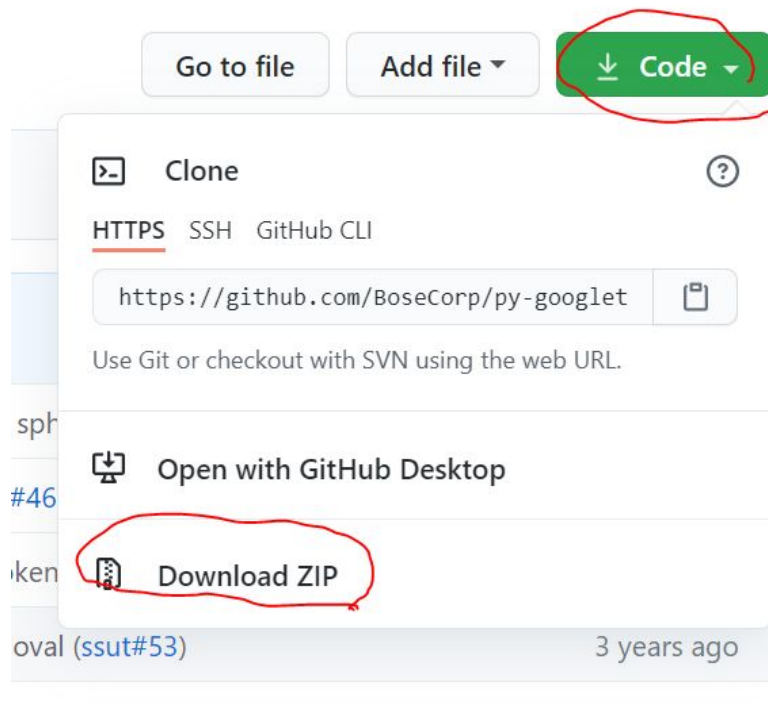
สร้าง Label แสดงคำว่า คำแปล แล้วทำการแสดงบน GUI ด้วยคำสั่ง .pack()

สร้าง v\_result ให้เป็น StringVar() เพื่อเป็นค่าข้อมูลที่เป็นข้อความจากการแปลคำศัพท์ และกำหนด FONT2 ให้เป็นรูปแบบอักษรอีกแบบ ('Angsana New',20) ให้มาแสดงค่าใน R1 ในรูปแบบของ Label แล้วทำการแสดงบน GUI ด้วยคำสั่ง .pack()

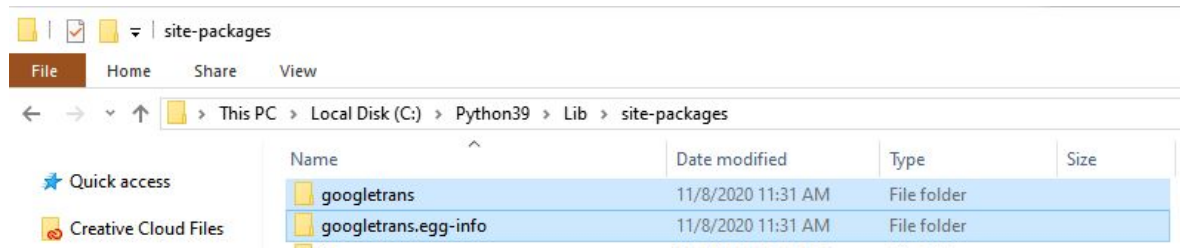
สุดท้าย สั่ง .mainloop() เพื่อสั่งให้หน้าต่าง GUI ตลอดจนกว่าจะมีการกดปิด

## วิธีการแก้ไข Google Translate

- 1- <https://github.com/BoseCorp/py-googletrans>
- 2- Click : Code > Download ZIP



- 3- คลิกขวาที่ไฟล์ py-googletrans-master.zip แล้ว Extract All
- 4- เข้าไปโฟลเดอร์ py-googletrans-master/py-googletrans-master (จนเห็นไฟล์ setup.py)
- 5- เปิด cmd ขึ้นมา
- 6- `pip uninstall googletrans`
- 7- `pip install requests`
- 8- พิมพ์ใน cmd ว่า `cd C:\Users\Uncle Engineer\Desktop\py-googletrans-master\py-googletrans-master` คล้ายๆแบบนี้แต่ต้องเป็นตำแหน่งโฟลเดอร์ในเครื่องตัวเอง
- 9- `python setup.py install`
- 10- ก๊อปปี้ 2 โฟลเดอร์นี้ไปเก็บไว้ในตำแหน่งของ Python : `C:\Python39\Lib\site-packages\`



11- ปิดโปรแกรม IDLE ทั้งหมดแล้วเปิดไฟล์ GUITranslator.py แล้วทดสอบรันโปรแกรม



# โปรแกรมค้นหาข้อมูลผ่าน wikipedia

```
*test.py - C:/Users/SPT/Desktop/BootCamp 2021/test.py (3.9.0)*
File Edit Format Run Options Window Help

# test-docx.py
from docx import Document
import wikipedia

def Wiki(keyword, lang='th') :
    wikipedia.set_lang(lang)
    # summary สำหรับบทความที่สรุป
    data = wikipedia.summary(keyword)

    # page + content บทความทั้งหน้า
    data2 = wikipedia.page(keyword)
    data2 = data2.content

    doc = Document() #สร้างไฟล์ word ใน python
    doc.add_heading(keyword, 0)

    doc.add_paragraph(data2)
    doc.save(keyword + '.docx')
    print('สร้างไฟล์สำเร็จ')

try:
    Wiki('asdfasdfadf', 'en')
except:
    print('ERROR')

# Wiki('united state of america', 'en')
# Wiki('ประเทศไทย', 'jp')
```

สรุปจาก Code :

การอธิบายต่อไปนี้จะไม่นับบรรทัดที่เป็นคำอธิบายในโปรแกรม (คอมเมนต์ #...)

```
# test-docx.py
from docx import Document
import wikipedia
```

เรียกใช้ฟังก์ชัน Document จาก Lib python-docx ดังนั้นอย่าลืม pip install python-docx เพื่อติดตั้งไลบรารีผ่าน cmd และ wikipedia ก็เช่นกัน

```
def Wiki(keyword, lang='th') :
    wikipedia.set_lang(lang)
    # summary สำหรับบทความที่สรุป
    data = wikipedia.summary(keyword)

    # page + content บทความทั้งหน้า
    data2 = wikipedia.page(keyword)
    data2 = data2.content

    doc = Document() #สร้างไฟล์ word ใน python
    doc.add_heading(keyword, 0)

    doc.add_paragraph(data2)
    doc.save(keyword + '.docx')
    print('สร้างไฟล์สำเร็จ')
```

สร้างฟังก์ชัน Wiki() โดยมีค่าที่จะส่งให้ฟังก์ชัน 2 ค่า ด้วย คือ keyword และ lang

.set\_lang() เพื่อสั่งให้ข้อมูลที่หาออกมาในภาษาใด ๆ ที่เราเลือกค่า lang ให้กับฟังก์ชัน

.summary() เป็นฟังก์ชันเพื่อค้นหาข้อมูลโดยสรุปของ keyword หรือคำค้นหาที่เราส่งให้

.page() เราจะได้ข้อมูลในหน้าต่างข้อมูลตามบนเว็บ wikipedia เกี่ยวกับ keyword ที่เราหา แต่รูปแบบข้อมูลนี้เราจะต้องแปลงด้วย .content เพื่อแปลงรูปแบบให้ออกมาสวย ตาม Lib ที่ออกแบบมา

จากนั้นใช้ Document() เพื่อสร้างไฟล์ word

.add\_heading() เป็นการใส่ข้อความหัวข้อ โดยใช้ keyword มาเป็นหัวข้อ 'วัด' และ .add\_paragraph() เพื่อใส่ข้อมูลจากการค้นหาด้วย wiki

.save() เป็นการสั่งบันทึกไฟล์ โดยในตัวอย่างเป็นการใช้ชื่อไฟล์ จาก keyword ที่เราใช้ค้นหา แล้วลงท้ายด้วย '.docx' เพื่อบันทึกเป็นไฟล์ word

แล้ว print('สร้างไฟล์สำเร็จ') เพื่อแสดงใน shell หลังจากมีการบันทึกไฟล์

\*หลังจากขั้นตอนนี้เราสามารถไปดูไฟล์ word ในตำแหน่งเดียวกับโปรแกรมของเราได้ จะพบไฟล์ word ขึ้นมา

```
try:
    Wiki('asdfasdfadf', 'en')
except:
    print('ERROR')

# Wiki('united state of america', 'en')
```

สร้างเงื่อนไข try except เพื่อเป็นการเลี่ยงการเกิด Error ของโปรแกรม จากการค้นหาข้อมูลไม่พบใน Wikipedia

โดยใน try: จะเป็นการเรียกฟังก์ชัน Wiki() เพื่อค้นหาข้อมูลที่เราพิมพ์ว่า 'asdfasdffadf' ซึ่งเป็นข้อมูลที่ไม่สามารถค้นหาได้ ออกมาเป็นภาษาอังกฤษ

แล้วหากไม่สามารถหาข้อมูลได้ โปรแกรมจะแสดงคำว่า 'ERROR' บน Shell

**\*\*ลองเปลี่ยนค่า ที่เราจะหา แล้วทำการรันใหม่เพื่อดูการทำงาน**

## โปรแกรมค้นหาข้อมูลผ่าน wikipedia (GUI)

```
#เปลี่ยนเป็นภาษาไทย
wikipedia.set_lang('th')
from tkinter import *
from tkinter import ttk
from tkinter import messagebox
```

## สรุปจาก Code :

การอธิบายต่อไปนี่เราจะไม่นับบรรทัดที่เป็นคำอธิบายในโปรแกรม (คอมเมนต์ #...)

```
#เปลี่ยนเป็นภาษาไทย
wikipedia.set_lang('th')
from tkinter import *
from tkinter import ttk
from tkinter import messagebox
```

จากโปรแกรมที่แล้วเราจะลบ try except ออก เพื่อไปประกาศใช้หลังจากเราเริ่มกดปุ่ม Search ซึ่งเราจะต้องสร้างหน้าต่าง GUI ให้การใช้โปรแกรมง่ายขึ้น  
โดยลำดับแรกแปลงให้ข้อมูลที่เรามาออกมาเป็นภาษาไทย ด้วย .set\_lang()  
และทำการเรียกฟังก์ชันที่จะใช้จาก tkinter โดยจากภาพเราสามารถได้

```
from tkinter import *                                #เป็นการ import ฟังก์ชันทั้งหมด จากไฟล์ __init__
from tkinter import ttk,messagebox                   #import ฟังก์ชัน หรือโมดูลเสริม
```

แทนได้ เพื่อลดบรรทัด และโปรแกรมสะดวกขึ้น (เราสามารถ import บรรทัดเดียวกันได้หลายตัว จาก Lib ตัวเดียวกัน  
ในบรรทัดเดียว ยกเว้น import \* ต้อง import แยก)

```
GUI = Tk()
GUI.title('โปรแกรม wiki')
GUI.geometry('400x300')
# config
FONT1 = ('Angsana New',15)

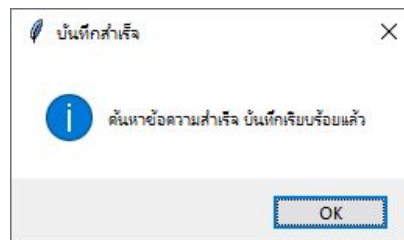
# คำอธิบาย
L = ttk.Label(GUI, text='ค้นหาบทความ',font=FONT1)
L.pack()
# ช่องค้นหาข้อมูล
v_search = StringVar() #กล่องสำหรับเก็บผลลัพธ์
E1 = ttk.Entry(GUI,textvariable = v_search,font=FONT1,width=40)
E1.pack(pady=10)
```

สร้างหน้าต่างโปรแกรมด้วย Tk() กำหนดชื่อที่แสดงบน Taskbar “โปรแกรม wiki” และขนาดเท่ากับ 400\*300 px กำหนด FONT1 เพื่อใช้กับ L และ E1 (ข้อความ และตัวอักษรในช่องกรอก)

```
# บุ่มค้นหา
def Search() :
    keyword = v_search.get() # .get() คือ ดึงข้อมูลเข้ามา
    try:
        # ลองค้นหาว่าได้ผลลัพธ์หรือไม่ หากได้ให้ผ่านไป
        language = v_radio.get() # th / en / zh
        Wiki(keyword,language)
        messagebox.showinfo('บันทึกสำเร็จ', 'ค้นหาข้อความสำเร็จ บันทึกเรียบร้อยแล้ว')
    except:
        # หากรันคำสั่งแล้วมีปัญหา แสดงข้อความแจ้งเตือน
        messagebox.showwarning('Keyword Error', 'กรุณากรอกคำค้นหาใหม่')

# print(wikipedia.search(keyword))
# result = wikipedia.summary(keyword)
# print(result)
```

สร้างฟังก์ชัน Search() เพื่อทำการค้นหาข้อมูลต่าง ๆ ด้วยฟังก์ชัน Wiki() ที่เรามีการทำไว้ เพื่อค้นหาข้อมูลซึ่งค่าของหัวข้อที่ส่งค้นหาจะเป็นค่าที่เรากรอกในช่องกรอก (E1) และขึ้น messagebox เพื่อเตือนผู้ใช้งานว่า ค้นหาข้อมูลที่ต้องการได้หรือไม่ และบันทึกไฟล์ word รียบ โดยในฟังก์ชันจะมีการใช้ try except เพื่อเป็นเงื่อนไข หากเจอข้อมูลที่ค้นหา จะขึ้น MessageBox “บันทึกสำเร็จ” แล้วมีข้อความแจ้งว่า “ค้นหาข้อมูลสำเร็จ บันทึกเรียบร้อยแล้ว” หลังวงกลมสัญลักษณ์ i สีฟ้า



แต่ถ้าหากไม่เจอ หรือข้อมูลจากหัวข้อมีหลายคำตอบข้อมูลที่ค้นหา จะขึ้น MessageBox “Keyword Error” แล้วมีข้อความแจ้งว่า “กรุณากรอกคำค้นหาใหม่” หลังสามเหลี่ยมสัญลักษณ์ ! สีเหลือง



```

B1 = ttk.Button(GUI, text='Search', command=Search)
B1.pack(ipadx=20, ipady=10) #ipadx ขยายภายในปุ่มแนวนอน

# เลือกภาษา
F1 = Frame(GUI)
F1.pack(pady=10)

v_radio = StringVar() # ช่องเก็บข้อมูลภาษา

RB1 = ttk.Radiobutton(F1, text='ภาษาไทย', variable=v_radio, value='th')
RB2 = ttk.Radiobutton(F1, text='อังกฤษ', variable=v_radio, value='en')
RB3 = ttk.Radiobutton(F1, text='จีน', variable=v_radio, value='zh')
RB1.invoke() #สั่งให้คำเริ่มเป็นภาษาไทย

RB1.grid(row=0, column=0)
RB2.grid(row=0, column=1)
RB3.grid(row=0, column=2)

GUI.mainloop()

```

สร้างปุ่มกด เพื่อทำงานในฟังก์ชัน Search() แล้วมีการสร้าง F1 เพื่อกำหนด ตำแหน่งในการแสดงผลบน GUI เดิม เพื่อจัดรูปแบบหน้าต่างให้มีความเหมาะสม สำหรับปุ่มตัวเลือกในการเลือกภาษาของข้อมูลที่ค้นหา โดยแต่ละตัวเลือกถูกแสดงออกมาในรูปแบบการวางแบบ grid()

ติดตั้ง Sublime ลิงค์ดาวน์โหลด >>> <https://www.sublimetext.com/3>

## Download

Sublime Text 3 is the current version of Sublime Text. For bleeding-edge releases, see the [dev builds](#).

### Version: Build 3211

- [OS X](#) (10.7 or later is required)
- [Windows](#) - also available as a [portable version](#)
- [Windows 64 bit](#) - also available as a [portable version](#)
- [Linux repos](#) - also available as a [64 bit](#) or [32 bit tarball](#)

Sublime Text may be downloaded and evaluated for free, however a license must be [purchased](#) for continued use. There is currently no enforced time limit for the evaluation.

เลือกตาม OS ของเราเลย สามารถติดตั้งได้ตามขั้นตอนในลิงค์นี้เลย >>>

<https://medium.com/@UncleEngineer/ep-020-python-python-sublime-text-3-%E0%B8%95%E0%B8%B4%E0%B8%94%E0%B8%95%E0%B8%B1%E0%B9%89%E0%B8%87-sublime-%E0%B8%AA%E0%B8%B3%E0%B8%AB%E0%B8%A3%E0%B8%B1%E0%B8%9A%E0%B8%A3%E0%B8%B1%E0%B8%99-python-f92d95b787bf>



# โปรแกรมคำนวณ Vat (GUI การบ้าน)

ลิงค์ Source Code: file: GUI-Vat.py

[https://github.com/UncleEngineer/PythonBootcamp2021?fbclid=IwAR1jhDzrultLL5\\_Jgwdlrzq2eX4EOou-3jb1FubLjkcpSfXuQ0MMAoFJuT4](https://github.com/UncleEngineer/PythonBootcamp2021?fbclid=IwAR1jhDzrultLL5_Jgwdlrzq2eX4EOou-3jb1FubLjkcpSfXuQ0MMAoFJuT4)

```
C:\Users\SPY\Desktop\BootCamp 2021\GUI-Vat.py (Python Script) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

1 from tkinter import *
2 from tkinter import ttk
3
4 GUI = Tk()
5 GUI.geometry('500x400')
6 GUI.title('โปรแกรมคำนวณ vat')
7
8 # FONT ทั้งหมด
9 FONT1 = ('Angsana New',20)
10
11 #####ช่องกรอกข้อมูล (ชื่อสินค้า) #####
12 L = ttk.Label(GUI,text='ชื่อสินค้า',font=FONT1).pack() # ข้อความแสดง
13
14 v_product = StringVar() # ตัวแปรสำหรับเก็บชื่อสินค้าตอนพิมพ์
15 E1 = ttk.Entry(GUI,textvariable=v_product,font=FONT1)
16 E1.pack()
17
18 #####ช่องกรอกข้อมูล (ราคาสินค้า) #####
19 L = ttk.Label(GUI,text='ราคาสินค้า',font=FONT1).pack() # ข้อความแสดง
20
21 v_price = StringVar() # ตัวแปรสำหรับเก็บราคาสินค้าตอนพิมพ์
22 E2 = ttk.Entry(GUI,textvariable=v_price,font=FONT1)
23 E2.pack()
24
25 #####ช่องกรอกข้อมูล (จำนวน) #####
26 L = ttk.Label(GUI,text='จำนวน',font=FONT1).pack() # ข้อความแสดง
27
28 v_quantity = StringVar() # ตัวแปรสำหรับเก็บจำนวนสินค้าตอนพิมพ์
29 E3 = ttk.Entry(GUI,textvariable=v_quantity,font=FONT1)
30 E3.pack()
31
32 #####ปุ่มกดเพื่อคำนวณ #####
33 def Calc(event=None):
34     # int() คำนวณแปลงข้อความเป็นตัวเลข '2' --> 2
35     # print( type( int( v_price.get() ) ) )
36     product = v_product.get()
37     price = int(v_price.get())
38     quantity = int(v_quantity.get())
39     total = price * quantity
40
41     vat7 = total * (7/107)
42     nettotal = total * (100/107)
43
44     print('ราคาก่อน vat: {:.2f} (vat 7%: {:.2f})'.format(nettotal,vat7))
45
46     v_result.set('สินค้า: {} ชิ้นทั้งหมด {} บาท ({} บาท/ชิ้น)\nราคาสินค้า: {:.2f} - VAT7%: {:.2f} -'.format(product,
47                                                                 quantity,total,
48                                                                 price,
49                                                                 nettotal,vat7))
50
51 B1 = ttk.Button(GUI,text='Calculate',command=Calc)
52 B1.pack(ipadx=20,ipady=10,pady=10)
53
54 E3.bind('<Return>',Calc)
55
56 #####ผลลัพธ์จากการคำนวณ #####
57 v_result = StringVar()
58 v_result.set('<<<ผลลัพธ์ที่นี่>>>') # โชว์ข้อมูลเริ่มต้น
59
60 R1 = ttk.Label(GUI,textvariable=v_result,font=FONT1)
61 R1.pack()
62
63 GUI.mainloop()
```

ผลลัพธ์:

สรุปจาก Code :

การอธิบายต่อไปนี้จะไม่นับบรรทัดที่เป็นคำอธิบายในโปรแกรม (คอมเมนต์ #...)

```
C:\Users\SPT\Desktop\BootCamp 2021\GUI-Vat.py (Python Socket) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

GUI-Vat.py x
1 from tkinter import *
2 from tkinter import ttk
3
4 GUI = Tk()
5 GUI.geometry('500x400')
6 GUI.title('โปรแกรมคำนวณ vat')
7
8 # FONT ทั้งหมด
9 FONT1 = ('Angsana New',20)
10
11
```

กำหนดหน้าต่าง และแบบตัวอักษรของโปรแกรม

```

13 L = ttk.Label(GUI,text='ชื่อสินค้า',font=FONT1).pack() # ข้อความแสดง
14
15 v_product = StringVar() # ตัวแปรสำหรับเก็บชื่อสินค้าตอนพิมพ์
16 E1 = ttk.Entry(GUI,textvariable=v_product,font=FONT1)
17 E1.pack()
18 #####ช่องกรอกข้อมูล (ราคาสินค้า) #####
19 L = ttk.Label(GUI,text='ราคาสินค้า',font=FONT1).pack() # ข้อความแสดง
20
21 v_price = StringVar() # ตัวแปรสำหรับเก็บราคาสินค้าตอนพิมพ์
22 E2 = ttk.Entry(GUI,textvariable=v_price,font=FONT1)
23 E2.pack()
24 #####ช่องกรอกข้อมูล (จำนวน) #####
25 L = ttk.Label(GUI,text='จำนวน',font=FONT1).pack() # ข้อความแสดง
26
27 v_quantity = StringVar() # ตัวแปรสำหรับเก็บจำนวนสินค้าตอนพิมพ์
28 E3 = ttk.Entry(GUI,textvariable=v_quantity,font=FONT1)
29 E3.pack()

```

สร้าง Label และช่องกรอก 3 ส่วนคือ 1 ชื่อสินค้า 2 ราคาสินค้า และ 3 จำนวนสินค้า เพื่อรอรับค่าไปใช้ในการคำนวณ และนำมาแสดงผล ภายในฟังก์ชัน Calc()

จากนั้นทำการสร้างฟังก์ชัน Calc() เพื่อรับค่าจากช่องกรอกทั้ง 3 ช่อง มาคำนวณ แล้วแสดงผลด้วย method .set() ตามภาพ

```

31 def Calc(event=None):
32     # int() คำสั่งแปลงข้อความเป็นตัวเลข '2' --> 2
33     # print( type( int( v_price.get() ) ) )
34     product = v_product.get()
35     price = int(v_price.get())
36     quantity = int(v_quantity.get())
37     total = price * quantity
38
39     vat7 = total * (7/107)
40     nettotal = total * (100/107)
41
42     print('ราคาก่อน vat: {:.2f} (vat 7%: {:.2f})'.format(nettotal,vat7))
43
44     v_result.set('สินค้า: {} {} ชิ้นทั้งหมด {} บาท ({} บาท/ชิ้น)\nราคาสินค้า: {:.2f}.- VAT7%: {:.2f}.-'.format(product,
45                                                         quantity,total,
46                                                         price,
47                                                         nettotal,vat7))

```

โดยจากภาพเราจะเห็นว่าการประกาศ method .get() เพื่อดึงค่าจากช่องกรอก 3 ส่วน แต่มีค่าของ price และ quantity ที่มีการประกาศเป็น int() เพื่อให้ค่าที่เรากรอกถูกเก็บไว้ในรูปแบบค่าของตัวเลขจำนวนเต็มเพื่อนำไปคำนวณต่อได้ แล้วจึงทำการคำนวณภาษี ทั้งเกณฑ์การคำนวณแบบรวมภาษี และไม่รวมภาษี แล้วนำค่าที่ได้ไปแสดงตามบรรทัดที่ 44

```

48 B1 = ttk.Button(GUI, text='Calculate', command=Calc)
49 B1.pack(ipadx=20, ipady=10, pady=10)
50
51 E3.bind('<Return>', Calc)
52
53 ##### ผลลัพธ์จากการคำนวณ #####
54 v_result = StringVar()
55 v_result.set('<<ผลลัพธ์โชว์จุดนี้>>') # โชว์ข้อมูลเริ่มต้น
56
57 R1 = ttk.Label(GUI, textvariable=v_result, font=FONT1)
58 R1.pack()
59
60
61 GUI.mainloop()

```

จากนั้นทำปุ่ม Calculate เพื่อนำมาใช้งานฟังก์ชัน Calc() แล้วทำการใส่ method .bind('<Return>', Calc) เพื่อเป็นการกำหนดให้ การกดปุ่ม Enter จะทำการรันฟังก์ชัน Calc() แต่จะสังเกตเห็นว่าที่ฟังก์ชัน Calc() ตอนประกาศสร้างนั้น เรามีการใส่ Attribute event=None ไว้ด้วย เพื่อให้เราสามารถใช้งานปุ่มคำนวณได้ทั้งการกด Enter หรือใช้เมาส์คลิกแล้วทำการ .set() เพื่อกำหนดตำแหน่งค่าผลลัพธ์หลังคำนวณ แล้วเอามาแสดงที่ตำแหน่งเดียวกับคำว่า '<<ผลลัพธ์โชว์จุดนี้>>'

### ผลลัพธ์

โปรแกรมคำนวณ vat

ชื่อสินค้า  
A01

ราคาสินค้า  
100

จำนวน  
25

Calculate

สินค้า: A01 25 ชิ้นทั้งหมด 2500 บาท (100 บาท/ชิ้น)  
ราคาสินค้า: 2336.45.- VAT7%: 163.55.-

## ฟังก์ชัน input()

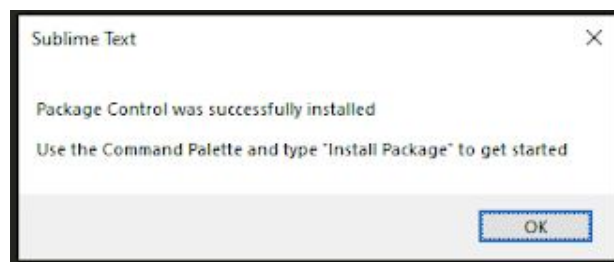
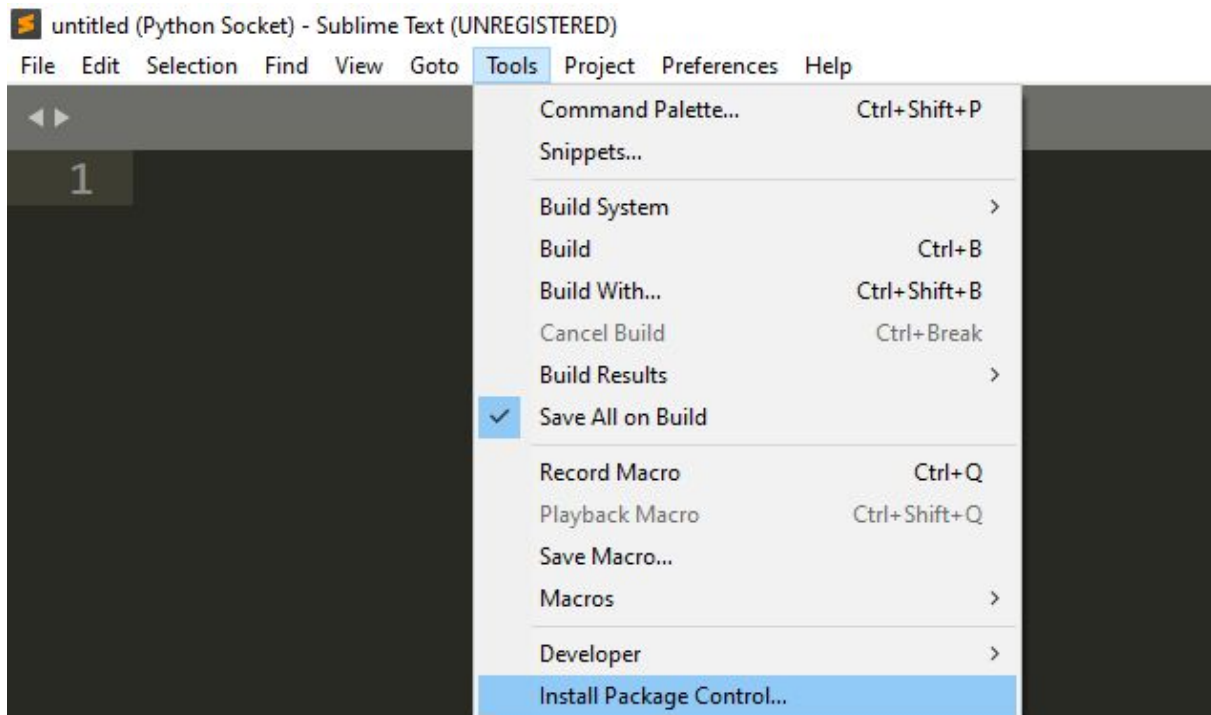
หากเราต้องการจะใช้งานฟังก์ชันที่มีการรับค่าจากการกรอกข้อมูลของผู้ใช้งานไปใช้งานกับโปรแกรม เราก็สามารถใช้ input() มาใช้ได้เลย ดังตัวอย่าง

```
1 money = input('Enter Money: ')\n2 print('Money: ',money)
```

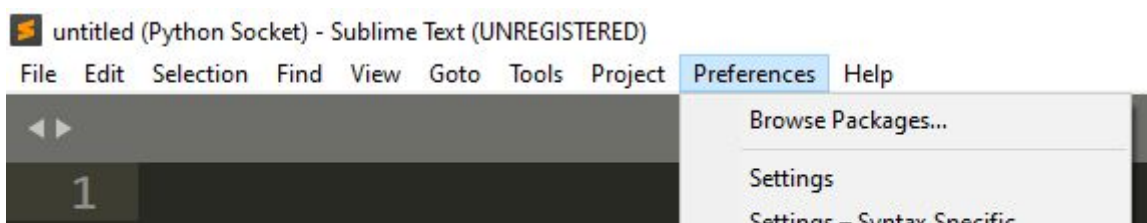
แต่ถ้าเราต้องการจะใช้งานบน Sublime อย่าลืมติดตั้ง package ที่ชื่อว่า SublimeREPL ด้วย โดยต้องติดตั้ง package control ก่อน แล้วก็ทำการเลือก SublimeREPL ดังนี้

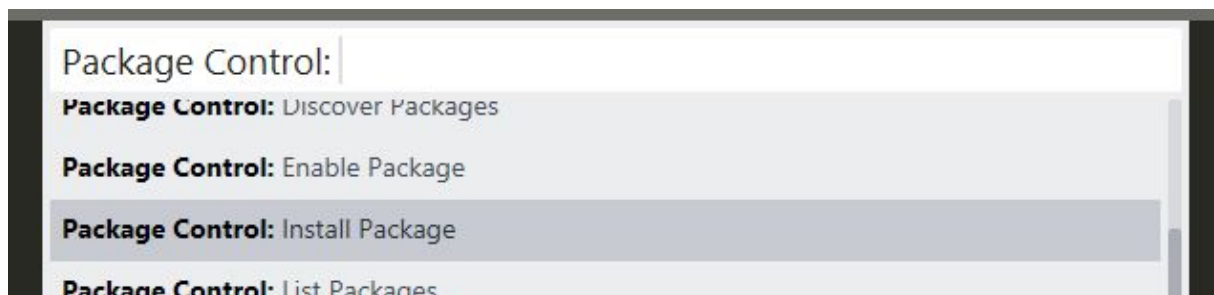
## การติดตั้ง package control และ SublimeREPL

คลิกที่ Tools แล้วเลือก Install Package Control...

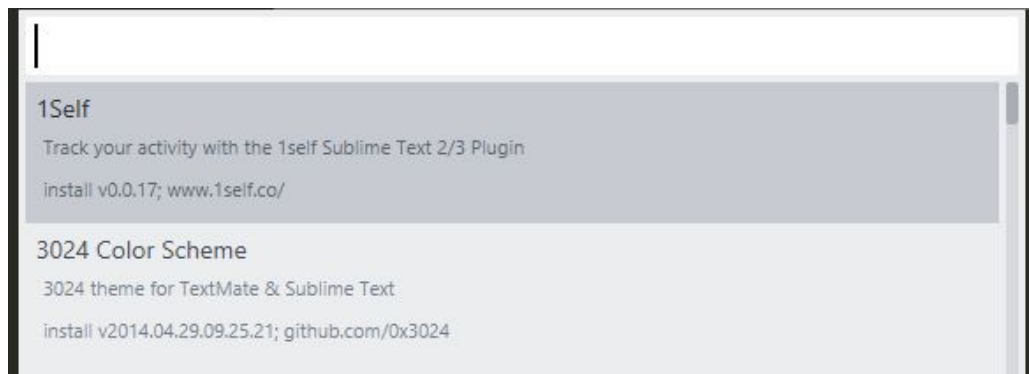


พอขึ้นแจ้งว่าติดตั้งแล้วไปดูที่ Preferences แล้วไปที่ Package Control





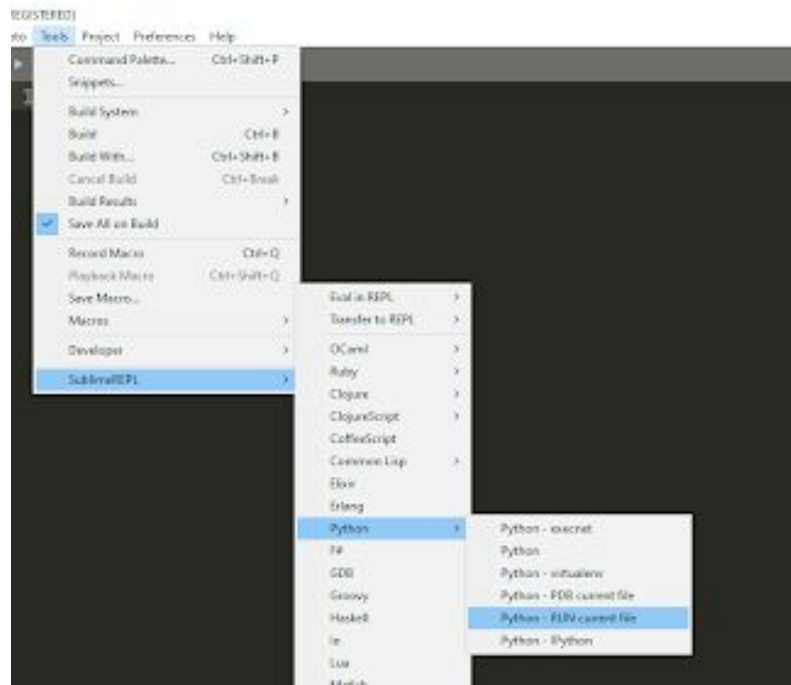
จะมีหน้าต่างขึ้นมา ให้เลือก **Install Package** แล้วรอสักครู่ จะมีหน้าต่างใหม่มา



แล้วใส่ชื่อไลบรารีที่เราจะติดตั้ง (Anaconda, SublimeREPL)



จากนั้นเมื่อเราไปเลือกที่ Tools >> SublimeREPL >> Python >> RUN current file



จะได้หน้าต่างให้เรากรอกค่า money ลงไป ดังตัวอย่าง เราทำการกรอก 500

```
Enter Money: 500
Money: 500

***Repl Closed***
```

โปรแกรมจำลองการโอนเงิน

```

money = 998
transfer = 2000

# print('Condition: ', money < transfer)
print('ต้องการโอน', transfer , '(มีค่าบริการ 15 บาท)')
while money < (transfer + 15 ) :
    print('คุณมีเงิน', money)
    print('กรุณาโอนเงินเข้าบัญชี เงินไม่พอโอน')
    getmoney = int(input('ฝากเงินเท่าไร?: '))
    money = money + getmoney # 998 + xxx
    print('---')

print('คุณมีเงิน', money)
print('โอนเงินได้เลย')
print('เหลือเงินในบัญชี: ', money - (transfer+15))

```

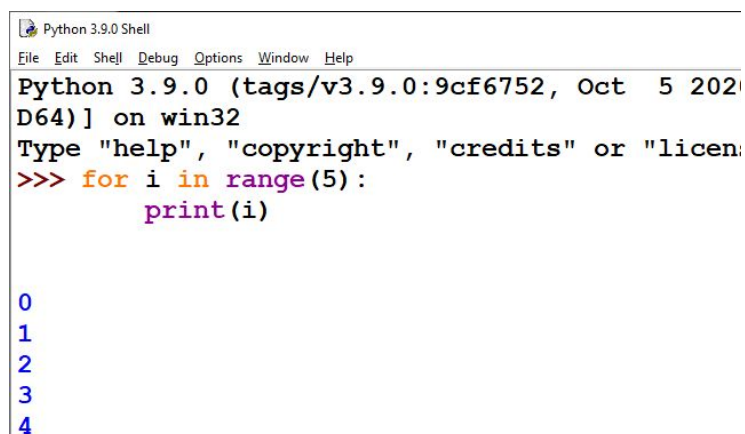
จะเห็นว่ามีการประกาศค่าของ money และ transfer เป็นตัวแปร เพื่อกำหนดค่าของเงินที่เราจะโอน และยอดที่เราจะโอนได้

จากนั้นทำการสร้าง while loop เพื่อให้มีการทำงานแบบเดิมซ้ำ ๆ จนกว่าจะมีการฝากเงินเข้าจนครบจำนวนที่ต้องการโอน

## ความแตกต่างระหว่าง for loop กับ while loop

while loop จากโปรแกรมด้านบน เราจะสังเกตเห็นว่าเป็นการทำงานซ้ำ ๆ จนกว่าจะมีการออกจากเงื่อนไข หรือจากตัวอย่างก็ คือ เมื่อไหร่ที่ค่าของ money น้อยกว่า transfer +15 หากยังไม่ออกจากเงื่อนไขนั้นโปรแกรมจะไม่หยุดทำงาน

แต่ for loop จะเป็น loop ที่ทำงานวนตามจำนวนที่เรากำหนดให้ ดังตัวอย่างต่อไปนี้



```

Python 3.9.0 Shell
File Edit Shell Debug Options Window Help
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5 2021) on win32
Type "help", "copyright", "credits" or "license()"
>>> for i in range(5):
        print(i)
0
1
2
3
4

```

จากภาพด้านบนจะเห็นว่าการ print(i) ได้ออกมาคือ 0 ถึง 4 นั้นเนื่องมาจาก การสั่ง for loop ให้ทำงานในช่วงจำนวน 5 ครั้ง (range(5) นั่นเอง)

แต่ทำไมเราถึงไม่ได้ค่า 5 ละ? นั้นเพราะว่า python เราจะเริ่มนับจาก 0 เป็นค่าเริ่มต้น และคำว่า range(5) หมายถึง ช่วงของค่า จำนวน 5 ค่า โดยเราสามารถกำหนดให้เริ่มจาก 1 ไปถึง 5 ก็ได้เช่นกัน ด้วยการประกาศ range(1,6) ตามภาพต่อไปนี้

```
>>> for i in range(1,6):  
    print(i)  
  
1  
2  
3  
4  
5
```

เราก็จะได้ค่า 1 ถึง 5 โดยที่สามารถมองการให้ค่าของ range() ได้ดังนี้ range(arg1,arg2) โดยที่ ค่า arg1 หมายถึง ค่าเริ่มต้นของช่วง และค่า arg2 เป็นค่าสุดท้ายซึ่งจะไม่นำมาแสดง(ค่าที่เราต้องการ + 1)

หรือเราก็สามารถใช้งาน for loop กับตัวแปรแบบ list เพื่อเรียกค่าใน list มาใช้งานเป็นรอบ ๆ ได้ ตามภาพต่อไป

```
>>> library = ['book1', 'book2', 'book3', 'book4']  
>>> for i in library:  
    print(i)  
  
book1  
book2  
book3  
book4
```

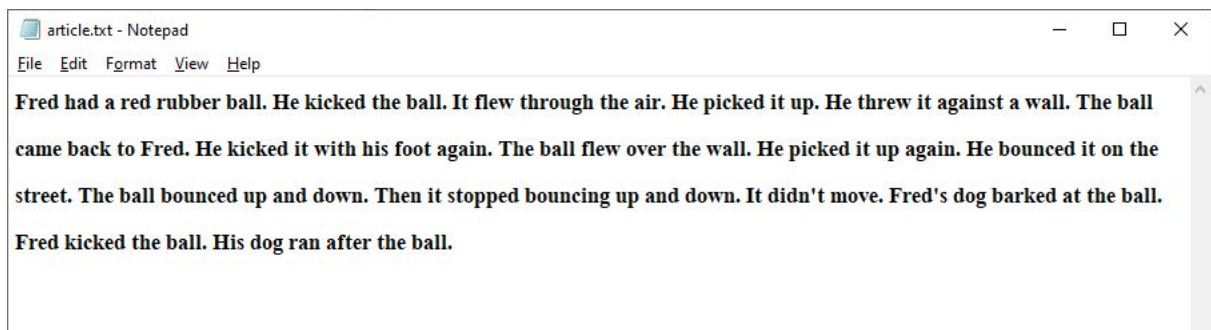
## โปรแกรมแปลคำศัพท์จากบทความ

เราสามารถใช้โปรแกรมนี้แทนตัว Google Translator ได้ในระดับพื้นฐาน ระหว่างรอการอัปเดตแก้ไข Google Translator Lib ลองได้สร้าง Lib ที่ชื่อว่า easyread โดยเราสามารถติดตั้งผ่านคำสั่ง pip install easyread มาใช้งานแบบ Lib ทัวไปได้เลย โดยคำแปลที่ได้มาจาก Lib ตัวนี้เป็นการอ้างอิงคำแปลจากฐานข้อมูลของเว็บไซต์ dict.longdo.com

```
1 #autotranslate.py
2
3 # pip install easyread
4 from easyread.translator import Translate
5 from openpyxl import Workbook
6
7 article = open('article.txt','r',encoding='utf-8')
8 article = article.read()
9 article = article.split()
10
11 print('Count: ',len(article))
12 result = []
13
14 for word in article:
15     #print(word)
16     res = Translate(word)
17     if res != None:
18         #print(res['meaning'])
19         result.append([word,res['meaning']])
20         # result.append(['Cat', '[N] แมว'])
21
22 #print(result)
23 excelfile = Workbook()
24 sheet = excelfile.active
25
26 header = ['Vocab','Translate']
27 sheet.append(header)
28
29 for rs in result:
30     sheet.append(rs)
31
32 excelfile.save('Vocab.xlsx')
```

จากโปรแกรมจะเป็นการเปิดไฟล์ article.txt เพื่อเก็บคำภาษาอังกฤษ ภายในไฟล์มา แล้วทำการ .split() เพื่อแบ่งคำจาก บทความนั้น ๆ มาแปลเป็นคำ ๆ ไป  
แล้วจะเห็นว่ามีการแสดงจำนวนคำศัพท์ที่แบ่งมาจากบทความด้วย len(article)

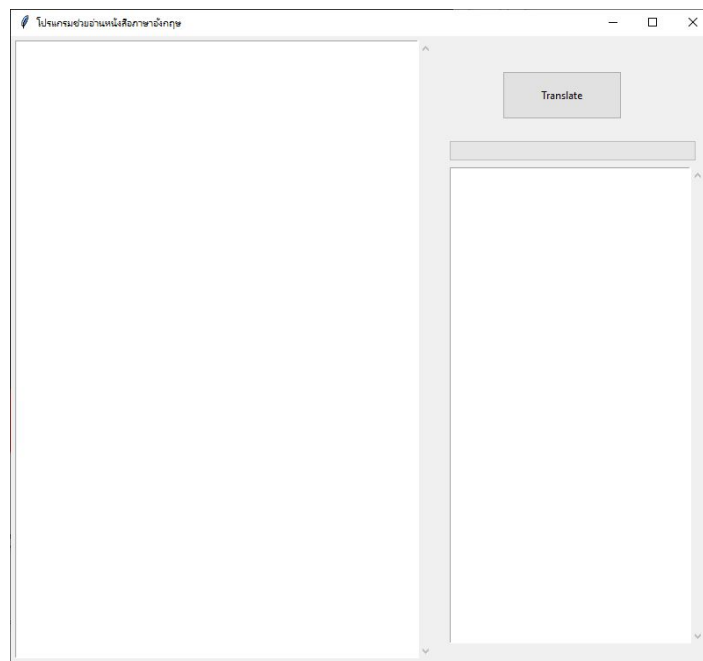
## ไฟล์ article.txt



แล้วมีการนำคำศัพท์ไปแปลด้วยคำสั่ง `.Translate()` ที่ละคำศัพท์ จากนั้นถ้าคำศัพท์ใดมีคำแปล จะทำการเก็บค่าไว้ในตัวแปร `result`

ภายในตัวแปร `result` เราจะได้ค่าของคำศัพท์ และคำแปลมา แล้วทำการนำ `result` เหล่านั้นไปวางใน `excel` เพื่อเป็นไฟล์ที่สามารถเปิดดูได้

จากนั้นเมื่อสั่งรันจะมีหน้าต่างโปรแกรมเด้งขึ้นมา ให้ปิดไปก่อน



จากนั้นเปิดดูไฟล์ `Volcab.xlsx` เพื่อดูคำศัพท์ที่แปลออกมาได้เลย โดยจำนวนคำ และคำศัพท์ และคำแปลที่ได้มาจะขึ้นอยู่กับข้อความใน `article.txt` นั้นเอง

## Dictionary

เป็นข้อมูลที่มีการอ้างอิงแบบ **key** และ **value** นั้นเหมือนการหาคำศัพท์จากพจนานุกรมเราจะต้องมีคำศัพท์(**key**) เพื่อนำไปเปิดหาความหมายของคำศัพท์นั้น ๆ มีอยู่อะไรบ้าง(**value**)

ดังภาพ

```
>>> fruits = {'japan': 'เมลอน', 'usa': 'องุ่น', 'china': 'แอปเปิ้ล'}
>>> print(fruits['japan'])
เมลอน
```

โดย **key** จะเป็นค่าที่ 1 ด้านหน้าเครื่องหมาย : ส่วน **value** คือค่าที่ 2 ด้านหลังเครื่องหมาย :  
นั่นเอง

หากเราต้องการเพิ่มค่าใหม่ก็สามารถประกาศได้แบบนี้เลย

```
>>> fruits['thailand'] = 'ทุเรียน'
>>> print(fruits)
{'japan': 'เมลอน', 'usa': 'องุ่น', 'china': 'แอปเปิ้ล', 'thailand': 'ทุเรียน'}
```

แต่ด้วยว่าค่าของตัวแปรแบบ **dictionary** จะมีค่าเป็น 2 ค่า (**key,value**) เวลาเราเรียกใช้กับ **for loop** จะไม่เหมือน **list** ในส่วนของค่าที่ออกมาจะเป็นเพียงค่า **key**

```
>>> for f in fruits:
...     print(f)
...
japan
usa
china
thailand
```

หากอยากได้ค่าของ **value** เราต้องเรียก **.values()** จากตัวแปรด้วย

```
>>> for f in fruits.values():
...     print(f)
...
เมลอน
องุ่น
ลูกห่อ
ทุเรียน
```



หรือถ้าเราต้องการใช้งานทั้ง 2 ค่า เราสามารถใช้ .items() มาใช้งานได้

```
>>> for k,v in fruits.items():
...     print('ลูกค้าครับ รบกวนหยิบ{} ในกล่องที่มีป้ายติดไว้ว่า{}'.format(v,k))
...
ลูกค้าครับ รบกวนหยิบเมล่อน ในกล่องที่มีป้ายติดไว้ว่าjapan
ลูกค้าครับ รบกวนหยิบองุ่น ในกล่องที่มีป้ายติดไว้ว่าusa
ลูกค้าครับ รบกวนหยิบลูกท้อ ในกล่องที่มีป้ายติดไว้ว่าchina
ลูกค้าครับ รบกวนหยิบทุเรียน ในกล่องที่มีป้ายติดไว้ว่าthailand
```

หากเราต้องการเก็บข้อมูลแบบหลาย ๆ รูปแบบ ตัวแปรแบบ dict ก็สามารถ แบ่งเป็นหมวดหมู่ได้ ดังภาพตัวอย่าง

```
>>> mobile = {}
>>> mobile['samsung'] = {'title': 'Samsung A51 2020 Ram 8/Rom 128GB',}
>>> mobile['samsung'] = {'title': 'Samsung A51 2020 Ram 8/Rom 128GB', 'price': 7660, 'discount': 2, 'oriprice': 7777, 'star': 5}
>>> print(mobile)
{'samsung': {'title': 'Samsung A51 2020 Ram 8/Rom 128GB', 'price': 7660, 'discount': 2, 'oriprice': 7777, 'star': 5}}
>>> mobile['iphone7'] = {'title': 'iPhone7 128gb เครื่องแท้', 'price': 10300, 'color': ['red', 'blue', 'yellow']}
>>> print(mobile)
{'samsung': {'title': 'Samsung A51 2020 Ram 8/Rom 128GB', 'price': 7660, 'discount': 2, 'oriprice': 7777, 'star': 5}, 'iphone7': {'title': 'iPhone7 128gb เครื่องแท้', 'price': 10300, 'color': ['red', 'blue', 'yellow']}}
```

ก็จะเห็นว่าจากตัวอย่าง เราสามารถเก็บข้อมูล ของโทรศัพท์ไว้ 2 ยี่ห้อ และมีการจำแนกข้อมูลของรุ่น ๆ ราคา สี ค่าความนิยม ของยี่ห้อนั้น ๆ ได้ด้วย

แต่ถ้าจากตัวอย่างเราจะเห็นว่า มี key รongบางตัวที่ต่างกัน เช่น color จะมีแค่ใน key iphone7 หากเราต้องการแสดง เราจะต้องมีเงื่อนไขกำกับด้วย ดังภาพด้านล่าง

```
>>> for m,d in mobile.items():
...     print(f'Mobile: {m}')
...     print('Title: {}'.format(d['title']))
...     print('Price: {} Baht'.format(d['price']))
...     if 'color' in d:
...         for c in d['color']:
...             print(c)
...     print('-----')
...
Mobile: samsung
Title: Samsung A51 2020 Ram 8/Rom 128GB
Price: 7660 Baht
-----
Mobile: iphone7
Title: iPhone7 128gb เครื่องแท้
Price: 10300 Baht
red
blue
yellow
```

ตัวอย่างการใช้งาน random.choice() และการทำเงื่อนไข if...else

```
import random

restaurant = {'high': [{'name': 'shitsuka sushi', 'price': 700},
                       {'name': 'Peporini', 'price': 500}],
              'medium': [{'name': 'เสวย', 'price': 200},
                         {'name': 'รสดี', 'price': 250}],
              'low': [{'name': 'ป้าส้ม', 'price': 40},
                     {'name': 'ป้าเล็กกระเพรา', 'price': 50}]}

money = 1000

if money >= 500:
    select = random.choice(restaurant['high'])
    print('คุณผู้หุงหาณร้าน{}ดีไหมครับ? ราคาเริ่มต้น {} บาท'.format(select['name'],select['price']))
elif money >= 250:
    select = random.choice(restaurant['medium'])
    print('คุณเพืหาณร้าน{}ดีไหมครับ? ราคาเริ่มต้น {} บาท'.format(select['name'],select['price']))
else:
    select = random.choice(restaurant['low'])
    print('พืหาณร้าน{}ดีไหมครับ? ราคาเริ่มต้น {} บาท'.format(select['name'],select['price']))
```

จากการโปรแกรมเราจะได้ออกความว่า

‘คุณผู้หุงหาณร้านshitsuka sushiดีไหมครับ? ราคาเริ่มต้น 700 บาท’

หรือ

‘คุณผู้หุงหาณร้านPeporiniดีไหมครับ? ราคาเริ่มต้น 500 บาท’

เพราะ money=1000 นั้น เข้าเงื่อนไขแรก ดังนั้นข้อมูลใน key ‘high’ จะถูกสุ่มมาใช้

```
money = int(input('คุณมีเงินเท่าไร?: '))
import random

restaurant = {'high': [{'name': 'shitsuka sushi', 'price': 700},
                       {'name': 'Peporini', 'price': 500}],
              'medium': [{'name': 'เสวย', 'price': 200},
                         {'name': 'รสดี', 'price': 250}],
              'low': [{'name': 'ป้าส้ม', 'price': 40},
                     {'name': 'ป้าเล็กกระเพรา', 'price': 50}]}

if money >= 500:
    select = random.choice(restaurant['high'])
    print('คุณผู้หุงหาณร้าน{}ดีไหมครับ? ราคาเริ่มต้น {} บาท'.format(select['name'],select['price']))
elif money >= 250:
    select = random.choice(restaurant['medium'])
    print('คุณเพืหาณร้าน{}ดีไหมครับ? ราคาเริ่มต้น {} บาท'.format(select['name'],select['price']))
else:
    select = random.choice(restaurant['low'])
    print('พืหาณร้าน{}ดีไหมครับ? ราคาเริ่มต้น {} บาท'.format(select['name'],select['price']))
```

จากนั้นลองใช้การ input() เพื่อให้ค่า money เป็นค่าที่เราต้องการ เพื่อทดสอบแต่ละเงื่อนไขของโปรแกรม

## โปรแกรมคำนวณ Vat (ต่อ)

จากโปรแกรมเดิม เราจะมาทำให้โปรแกรมมีฟังก์ชันมากขึ้นตามลำดับ  
โดยขั้นแรกเราจะเพิ่ม Radiobutton เพิ่มขึ้นมา เพื่อเป็นปุ่มตัวเลือกรูปแบบของการคำนวณ

```
29 E3.pack()
30
31
32
33 ##### Radio เลือกประเภท VAT #####
34
35 F1 = Frame(GUI)
36 F1.pack(pady=10)
37
38 v_radio = StringVar()
39
40 R1 = ttk.Radiobutton(F1, text='ราคารวม vat แล้ว', variable=v_radio, value='ic')
41 R1.grid(row=0, column=0)
42
43 R2 = ttk.Radiobutton(F1, text='ราคา + vat 7%', variable=v_radio, value='av')
44 R2.grid(row=0, column=1)
45
46 R3 = ttk.Radiobutton(F1, text='ราคาไม่รวม vat', variable=v_radio, value='nic')
47 R3.grid(row=0, column=2)
48
49 ##### ปุ่มกดเพื่อคำนวณ #####
50 def Calc(event=None):
51     print('RADIO: ', v_radio.get())
52     # int() ค่าส่งแปลงข้อความเป็นตัวเลข '2' -- > 2
```

จากนั้นสร้างเงื่อนไขให้กับปุ่มรูปแบบการเลือกคำนวณค่า Vat

```
59 total = price * quantity
60
61 if v_radio.get() == 'ic':
62     vat7 = total * (7/107)
63     nettotal = total * (100/107)
64     #print('ราคาก่อน vat: {:.2f} (vat 7%: {:.2f})'.format(nettotal, vat7))
65     v_result.set('สินค้า: {} จำนวน {} ชิ้น ทั้งหมด {} บาท ({} บาท/ชิ้น)\nราคาสินค้า: {:.2f} - VAT7%: {:.2f}.'.format(product,
66                                                                 quantity, total,
67                                                                 price,
68                                                                 nettotal, vat7))
69
70 elif v_radio.get() == 'av':
71     vat7 = (total * (7/100))
72     nettotal = total
73     sumtotal = total + vat7
74     v_result.set('สินค้า: {} จำนวน {} ชิ้น ทั้งหมด {:.2f} บาท ( {:.2f} บาท/ชิ้น)\nราคาสินค้า: {:.2f} - VAT7%: {:.2f}.'.format(product,
75                                                                 quantity, sumtotal,
76                                                                 price + (vat7 / quantity),
77                                                                 nettotal, vat7))
78
79 else:
80     v_result.set('สินค้า: {} จำนวน {} ชิ้น ทั้งหมด {:.2f} บาท ( {:.2f} บาท/ชิ้น)\n'.format(product, quantity, total, price))
81
82 B1 = ttk.Button(GUI, text='Calculate', command=Calc)
```

และกลับไปประกาศ .invoke() ที่ R1 หรือ Radiobutton ที่เราต้องการให้เป็นค่าเริ่มต้น

```
41 R1.grid(row=0, column=0)
42
43 R1.invoke() #เลือกเป็นค่าเริ่มต้น
44
```

จากนั้นลองรัน แล้วใช้งานโปรแกรมดูเลย

โปรแกรมคำนวณ vat

ชื่อสินค้า  
แอปเปิ้ล

ราคาสินค้า  
100

จำนวน  
1

☒ รวม vat แล้ว ☐ ราคา + vat 7% ☐ ราคาไม่รวม vat

Calculate

สินค้า: แอปเปิ้ล จำนวน 1 ชิ้น ทั้งหมด 100 บาท (100 บาท/ชิ้น)  
ราคาสินค้า: 93.46.- VAT7%: 6.54.-

## Class (basic-class.py)

โครงสร้างที่เราจะได้ยินว่า OOP หรือการเขียนโปรแกรมเชิงวัตถุ คือการอ้างอิงถึงโครงสร้างที่มีพื้นฐานอยู่แล้ว เพื่อเพิ่มเติม แก้ไขข้อมูลตัวต่อ ๆ ไป โดยในไพธอน Class จะเป็นส่วนที่ถูกสร้างมาเพื่อให้มีการอ้างอิงถึงตัวภาพ

```
1 #basic-class.py
2
3 class Student:
4     def __init__(self):
5         self.name = 'Albert'
6
7
8 student1 = Student()
9 print(student1.name)
```

โดยจะเห็นว่า เมื่อเรามีการสร้าง class Student ขึ้นมา มีการประกาศฟังก์ชัน `__init__(self)`

ซึ่งชื่อ `__init__` เป็นชื่อที่บอกให้เป็นค่าเริ่มต้นของ class นั้น ๆ แล้วเราจะเห็นว่ามีการสร้างตัวแปร `student1` ขึ้นมาเก็บค่าของ class ไว้ โดยการที่เราสร้างตัวแปรขึ้นมาเก็บค่า class นั้น ๆ เราจะ



เรียกว่า student1 นั้นเป็น object หรือคือการมองว่าเราจะนำวัตถุชิ้นนี้ไปทำงานอะไรต่อ โดยที่ชื่อของ object นั้น ๆ ก็คือ self ที่เราประกาศไว้ใน `__init__` นั่นเอง

จากโปรแกรมด้านบนเราจะได้รับการแสดงผลชื่อ Albert ออกมา เพราะมีการให้ค่า `name = 'Albert'` ไว้

หรือหากต้องการเปลี่ยนให้มีการแปลงค่าของชื่อ เราก็สามารถสร้าง arg ต่อจาก self เหมือนที่เราเคยลองสร้างฟังก์ชันกันได้เลย ดังภาพ

```
3 class Student:
4     def __init__(self, name):
5         self.name = name
6
7
8 student1 = Student('Albert')
9 print(student1.name)
```

ลองเพิ่มฟังก์ชันเพิ่มใน คลาสนี้ดู

```
7     def Hello(self):
8         print('สวัสดีจ้าาา')
9
10
11 student1 = Student('Albert')
12 print(student1.name)
13 student1.Hello
14
15 student2 = Student('Steve')
16 print(student2.name)
17
```

เราก็จะได้ผลลัพธ์ดังนี้

```
Albert
สวัสดีจ้าาา
Steve
[Finished in 0.1s]
```

สำหรับ class แรกให้เพิ่มเติม code ดังนี้ตามเลย

```
1 #basic-class.py
2
3 class Student:
4     def __init__(self,name):
5         self.name = name
6         self.exp = 0
7         self.lesson = 0
8         #Call Function
9         # self.AddEXP(10)
10
11     def Hello(self):
12         print('สวัสดีจ้าาา ผมชื่อ{}'.format(self.name))
13
14     def Coding(self):
15         print('{}: กำลังเขียนโปรแกรม..'.format(self.name))
16         self.exp += 5
17         self.lesson += 1
18
19     def ShowEXP(self):
20         print('- {} มีประสบการณ์ {} EXP'.format(self.name,self.exp))
21         print('- เรียนไป {} ครั้งแล้ว'.format(self.lesson))
22
23     def AddEXP(self,score):
24         self.exp += score # self.exp = self.exp + score
25         self.lesson += 1
26
```



แล้วลองเรียกใช้ฟังก์ชันดูตามนี้

```
28 print('====1 Jan====')
29 student1 = Student('Albert')
30 print(student1.name)
31 student1.Hello()
32
33 print('-----')
34 student2 = Student('Steve')
35 print(student2.name)
36 student2.Hello()
37 print('====2 Jan====')
38 print('-----uncle: ใครอยากเรียนโค้ดดิ้ง?---(ให้ 10 exp)----')
39 student1.AddEXP(10)
40
41 print('====3 Jan====')
42 student1.name = 'Albert Einstein'
43 print('ตอนนี้ exp ของแต่ละคนได้เท่าไรกันแล้ว')
44
45 print(student1.name, student1.exp)
46 print(student2.name, student2.exp)
47 print('====4 Jan====')
48
49 for i in range(5):
50     student2.Coding()
51
52 student1.ShowEXP()
53 student2.ShowEXP()
```

ต่อมลองไปรู้จักการนำ class ใหม่ และ class เก่ามาใช้งานร่วมกัน หรือเราจะเรียกว่าการสืบทอด  
คลาส class inheritance จะเห็นการเรียก super().\_\_init\_\_(name) ดังภาพ

```
28 class SpecialStudent(Student):
29
30     def __init__(self, name, father):
31         super().__init__(name)
32         self.father = father
33         mafia = ['Bill Gates', 'Thomas Edison']
34         if father in mafia:
35             self.exp += 100
```

แล้วลองเรียกใช้งาน class ใหม่ดู

```
28 class SpecialStudent(Student):
29
30     def __init__(self, name, father):
31         super().__init__(name)
32         self.father = father
33         mafia = ['Bill Gates', 'Thomas Edison']
34         if father in mafia:
35             self.exp += 100
36
37
38 print('=====1 Jan=====')
39 student0 = SpecialStudent('Mark Zuckerberg', 'Bill Gates')
40 student0.ShowEXP()
```

จะเห็นว่า ในตัวแปร student0 เราสามารถเรียกฟังก์ชันจากคลาสเก่ามาใช้งานได้แม้เป็น obj ของ class ใหม่

จากนั้นลองเปลี่ยนชื่อ arg father จากตัวอย่างที่เป็น Bill Gates ให้เป็นชื่อที่ไม่ตรงเงื่อนไขรายชื่อ mafia แล้วสังเกตผลรับของค่า EXP ที่นักเรียนได้รับ

แล้วกลับมาเพิ่มเติม code ดังนี้ เพื่อทำการสร้างฟังก์ชันให้เหมาะสมกับ class ใหม่

```
34         if father in mafia:
35             self.exp += 100
36
37     def AddEXP(self, score):
38         self.exp += (score * 3)
39         self.lesson += 1
40
41     def AskEXP(self, score=10):
42         print('ครู!! ขอคะแนนพิเศษให้ผมหน่อยสิลึ่ก {} EXP'.format(score))
43         self.AddEXP(score)
44
45
46
47 print('=====1 Jan=====')
48 student0 = SpecialStudent('Mark Zuckerberg', 'Bill Gates')
49 student0.AskEXP()
50 student0.ShowEXP()
51 student1 = Student('Albert')
```