

# Lab1: Conditionals and loops

Prof. Jonghee Yoon ([jyoon48@ajou.ac.kr](mailto:jyoon48@ajou.ac.kr))

Ajou University, Department of Physics



# Using Google Colab for Python programming

- Google Colaboratory, or "Colab" as most people call it, is a cloud-based Jupyter notebook environment.
- It runs in your web browser.
- Colab is free of charge to use.

Colab 시작 페이지

Colab에 이미 익숙하다면 이 동영상상을 통해 양방향 테이블, 코드 실행 기록 보기, 명령어 팔레트에 관해 알아보세요.

3 Cool Google Colab Features

Colab이란?

Colaboratory(줄여서 'Colab'이라고 함)을 통해 브라우저 내에서 Python 코드를 실행할 수 있습니다.

- 구성이 필요하지 않음
- 무료로 GPU 사용
- 간편한 공유

학생이든, 데이터 과학자든, AI 연구원이든 Colab으로 업무를 더욱 간단하게 시작해 보세요.

시작하기

지금 읽고 계신 문서는 정적 웹페이지가 아니라 코드를 작성하고 실행할 수 있는 Jupyter Notebook입니다. 예를 들어 다음은 값을 계산하여 변수로 저장하고 결과를 출력하는 간단한 코드를 보여줍니다.

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day

86400
```

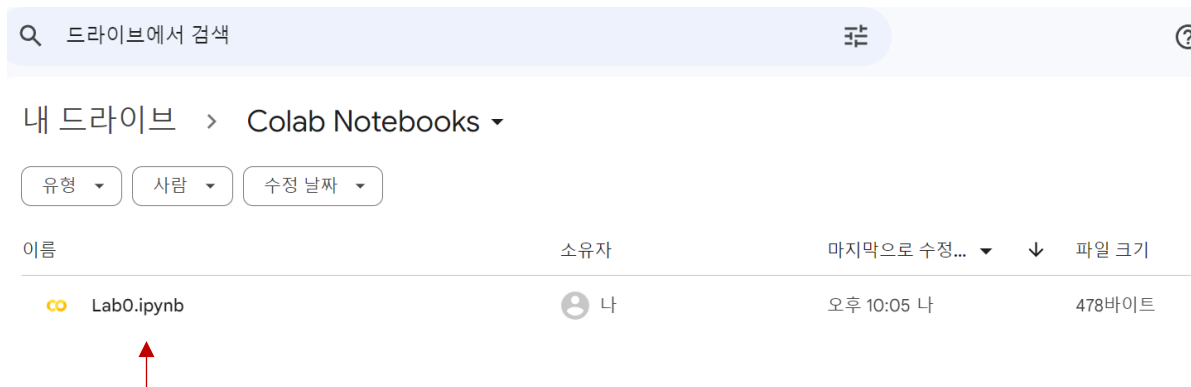
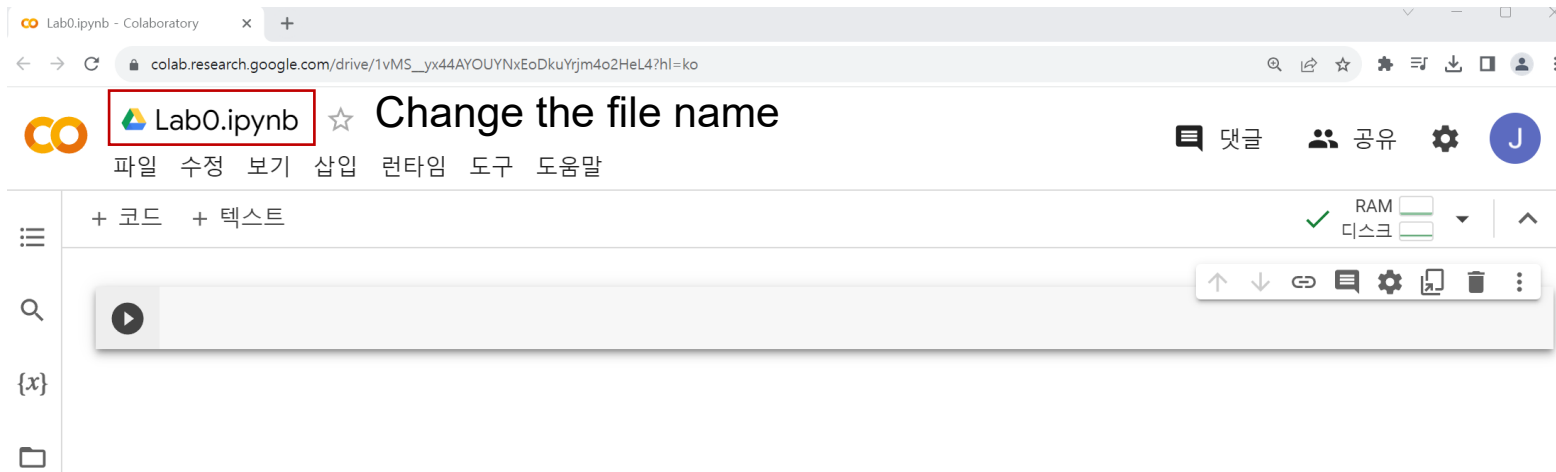
위 셀의 코드를 실행하려면 셀을 클릭하여 선택한 후 코드 왼쪽의 실행 버튼을 누르거나 단축키 Command/Ctrl+Enter를 사용하여 실행할 수 있습니다. 실행 버튼을 클릭하면 코드 수정을 바로 시작할 수 있습니다.

특정 셀에서 정의한 변수를 나중에 다른 셀에서 사용할 수 있습니다.

이름	최근 사용	Google Drive	GitHub	업로드
노트 필터링				
제목		마지막 연 시간	처음 연 시간	
Colaboratory에 오신 것을 환영합니다		오전 11:51	오전 11:51	

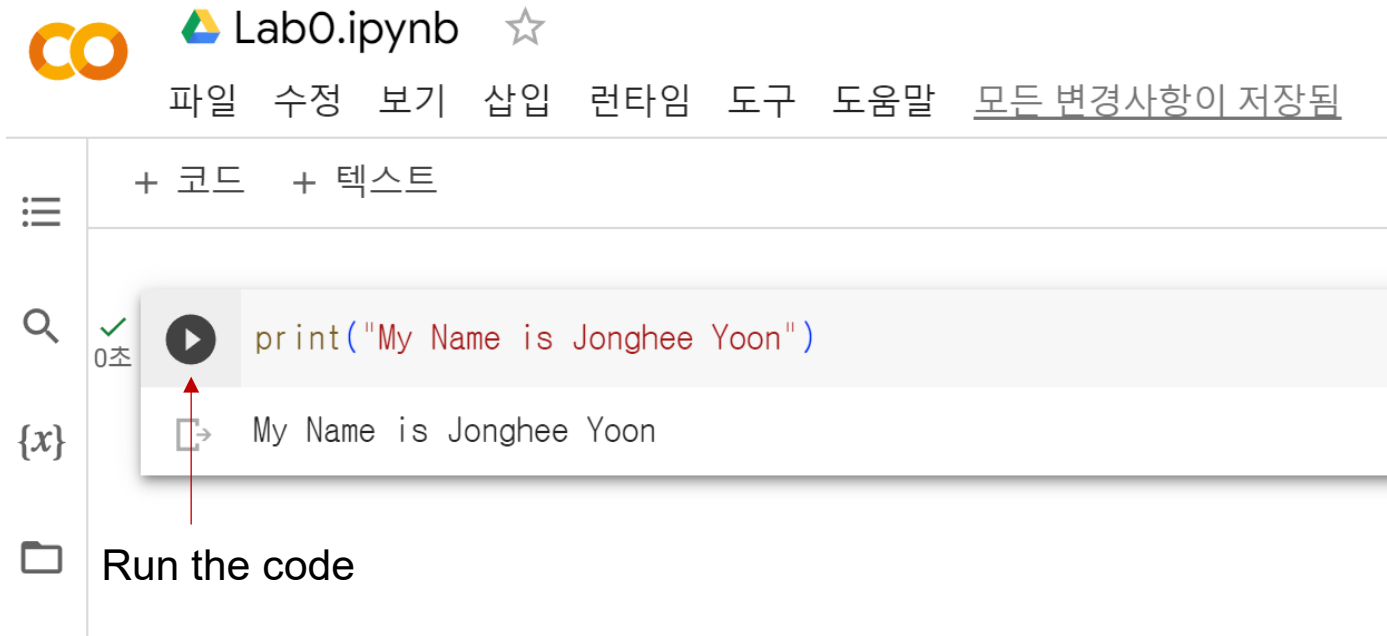
새 노트 취소

# Using Google Colab for Python programming



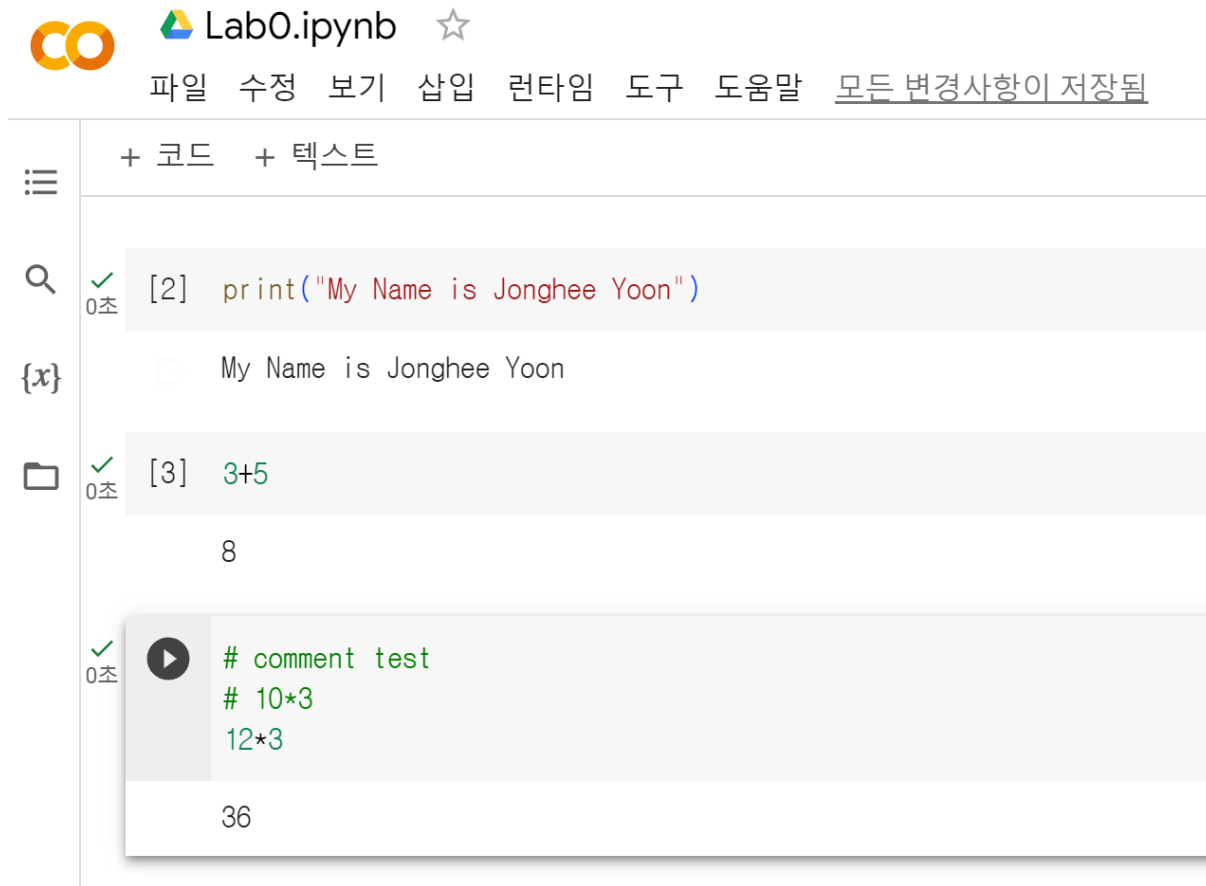
File you can submit

# Using Google Colab for Python programming



The screenshot shows the Google Colab interface. At the top, the Google Colab logo is followed by the file name "Lab0.ipynb" and a star icon. Below this is a navigation bar with Korean text: "파일" (File), "수정" (Edit), "보기" (View), "삽입" (Insert), "런타임" (Runtime), "도구" (Tools), "도움말" (Help), and "모든 변경사항이 저장됨" (All changes are saved). The main workspace has a tab labeled "+ 코드" (Code) and another labeled "+ 텍스트" (Text). On the left sidebar, there are icons for a menu, search, a variable "{x}", and a folder. The search icon has a green checkmark and "0초" (0 seconds) next to it. The variable "{x}" has a small square icon with a right arrow next to it. The folder icon has the text "Run the code" next to it. In the center, a code cell is shown with the code `print("My Name is Jonghee Yoon")`. Below the code cell, a tooltip displays the output "My Name is Jonghee Yoon". A red arrow points from the "Run the code" text to the play button icon on the code cell.

# Using Google Colab for Python programming



CO Lab0.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트

[2] `print("My Name is Jonghee Yoon")`

My Name is Jonghee Yoon

[3] `3+5`

8

`# comment test`  
`# 10*3`  
`12*3`

36

# → comment (not executable)

## Check points

---

- Implement codes to get the following results

### Result1

My Name is

My student number is

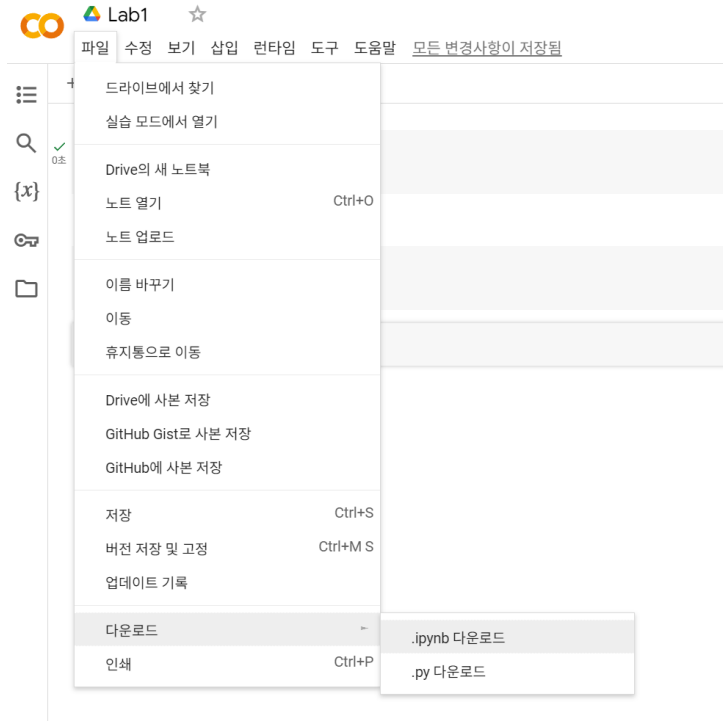
### Result2

$0.5 * 9.8 * 3^2 =$   
44.1



# Submit the code

- Download your code at Colab
- Upload your code to AjouBB



## 과제 지시 사항

수업 중에 작성한 Lab 1 코드를 업로드 하세요.  
기한: 24/09/13 23:59 까지  
24/09/14 23:59 이후 제출된 코드는 채점하지 않습니다.  
딜레이 페널티 50%

## 제출물

여기에 파일을 끌어서 놓거나 텍스트를 클릭하여 추가하십시오.

# What are variables?

---

- Variables are containers for data values.

$$f(x)=3x+5, y=1/x, F=ma$$

- A variable is created at the moment you first assign a value to it (memory address).

```
78 print(x)
Exception has occurred: NameError
name 'x' is not defined
```

```
x=3
print("A value of X is",x)
x=7
print("A value of X is",x)
```

```
A value of X is 3
A value of X is 7
```

- Arithmetic operators take numerical values (either literals or variables) as their operands and return a single numerical value.

```
x=2
y=5
print("x is",x,"y is",y)
print("x+y=",x+y,"x-y=",x-y,"x*y=",x*y,"x/y=",x/y)
```

```
x is 2 y is 5
x+y= 7 x-y= -3 x*y= 10 x/y= 0.4
```



# What are data types?

---

- There are many data types in Python
- Text type (letters): str
- Numeric type (numbers): int, float, complex
- Boolean type (true, false) : bool

```
x = "Ajou"  
print(x)  
print(x+" University")
```

```
Ajou  
Ajou University
```

```
x = 3  
print(x)  
print(x+5)
```

```
3  
8
```

# Task0: Print your name, date, lab number, and data types

---

- Print your name, date, lab number, and data types using variables

```
# Task 0
name = 
class_name = 
lab_num = 1
print("My name is ", )
print("This class is ", )
print("Lab number is", )
print( (name))
print( (class_name))
print( (lab_num))
```

```
My name is Jonghee Yoon
This class is AI Physics
Lab number is 1
<class 'str'>
<class 'str'>
<class 'int'>
```

# Task1: Motion under Constant Acceleration

- Calculate the velocity and distance at a certain time  $t$

표 2.2 등가속도 운동을 하는 입자의 운동학 방정식

식	방정식	방정식에 표시된 정보
2.13	$v_{xf} = v_{xi} + a_x t$	시간의 함수로 나타낸 속도
2.15	$x_f = x_i + \frac{1}{2}(v_{xi} + v_{xf})t$	속도와 시간의 함수로 나타낸 위치
2.16	$x_f = x_i + v_{xi}t + \frac{1}{2}a_x t^2$	시간의 함수로 나타낸 위치
2.17	$v_{xf}^2 = v_{xi}^2 + 2a_x(x_f - x_i)$	위치의 함수로 나타낸 속도

운동은  $x$  방향임.

```
# Task 1
v0 = 10 # m/s
d0 = -35 # m
a = 2 #m/s^2
t = 10 #s

v = 
d = 

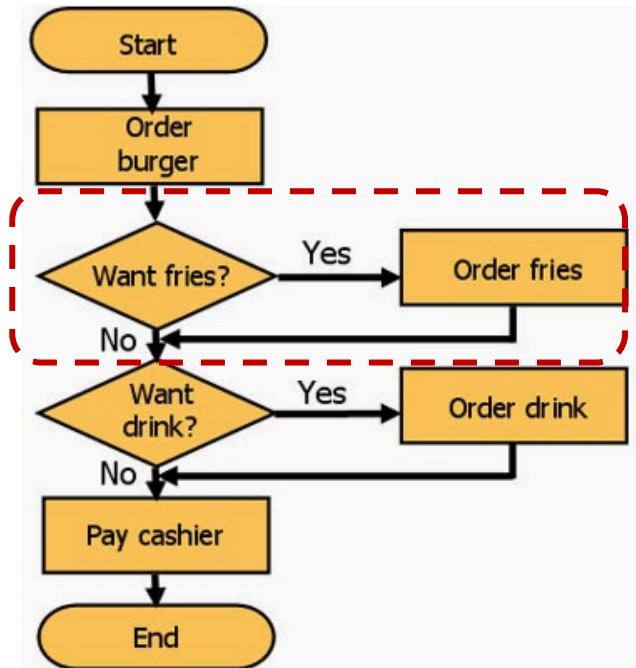
print( )
print( )
```

Velocity at 10 s is 30 m/s  
Distance at 10 s is 165.0 m/s

# What are conditionals?

---

- What makes programming so much more powerful are conditional statements.
- A **condition** is either **True** or **False** (**Boolean**)



if (a condition evaluates to True):  
then do these things only for 'True'

*e.g. Order fries*

else:

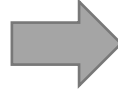
otherwise do these things only for 'False'

*e.g. Ask the next question*

# Conditional Examples

```
if True:
    print("Today is Wednesday")
if False:
    print("Every student will get A+!")
```

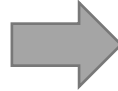
Run



Today is Wednesday

```
if 5>3:
    print("5 is larger than 3")
else:
    print("5 is smaller than 3")
```

Run



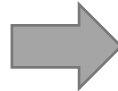
5 is larger than 3

*not True* is *False*

```
print(5>3) True print(not 5>3) False
```

```
if not 5>3:
    print("5 is larger than 3")
else:
    print("5 is smaller than 3")
```

Run



5 is smaller than 3

# Comparison Operators

---

Operator	Description	Example
==	If values of two operands are equal, the condition becomes true	(3==3): True, (3==5): False
!=	If values of two operands are not equal, then the condition becomes true	(3!=3): False, (3!=5): True
>	If the value of left operand is greater than the value of right operand, then condition becomes true.	(3>5): False, (5>3): True
<	If the value of right operand is greater than the value of left operand, then condition becomes true	(3<5): True, (5>3): False
>=	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true	(3>=3): True, (3>=5): False
<=	If the value of right operand is greater than or equal to the value of left operand, then condition becomes true	(3<=3): True, (5<=3): False



# Multiple options (if & elif)

if Option1: e.g. `scores>90`  
    then do Task1 e.g. `get A`  
elif Option2: e.g. `scores>80`  
    then do Task2 e.g. `get B`  
elif Option3: e.g. `scores>70`  
    then do Task3 e.g. `get C`  
elif Option4: e.g. `scores>60`  
    then do Task4 e.g. `get D`  
else: e.g. `scores<=60`  
    then do Task5 e.g. `get F`

```
score = 75
if score>90:
    print("Grade is A")
elif score>80:
    print("Grade is B")
elif score>70:
    print("Grade is C")
elif score>60:
    print("Grade is D")
else:
    print("Grade is F")
```

Grade is C

```
score = 65
if score>90:
    print("Grade is A")
elif score>80:
    print("Grade is B")
elif score>70:
    print("Grade is C")
elif score>60:
    print("Grade is D")
else:
    print("Grade is F")
```

Grade is D

## Task2: Grading program based on an average score

---

- Write a code for a letter grade based on an average score
- A (average score  $\geq 90$ ), B (average score  $\geq 80$ ), C (average score  $\geq 70$ ), D

```
math = 70
english = 57
biology = 83
physics = 93

average_score = 

print()
if 
    print("Grade is A")

    print("Grade is B")

    print("Grade is C")

    print("Grade is D")
```

```
Average score is 75.75
Grade is C
```



# for loop

---

- The *range()* function enables the iteration of a set of code a specified number of times
- The *range()* function returns a sequence of numbers, starting from 0 (by default), and increments by 1 (by default), and ends at a specified number

```
for x in range(3):  
    print("Repeat")
```

```
Repeat  
Repeat  
Repeat
```

```
for x in range(5):  
    print("Repeat")
```

```
Repeat  
Repeat  
Repeat  
Repeat  
Repeat
```

```
for x in range(3):  
    print(x)
```

```
0  
1  
2
```

```
for x in range(5):  
    print(x)
```

```
0  
1  
2  
3  
4
```

```
for x in range(2,5):  
    print(x)
```

```
2  
3  
4
```

```
for x in range(2,15,3):  
    print(x)
```

```
2  
5  
8  
11  
14
```

# while loop

---

- With the **while** loop we can execute a set of statements as long as a condition is true.

```
while True:  
    print(".....")
```

```
.....  
.....  
.....  
.....  
.....  
.....  
.....  
.....  
.....  
.....
```

.....

```
x=0  
while x<10:  
    print(x)  
    x=x+1
```

```
0  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

- 'Control + C' could stop the loop

## Task3: Printing power of numbers using for or while loops

- Make a code for printing power (1 – 10) according to the predefined value
- Use pow() function (e.g.  $\text{pow}(2,3)=2^3=8$ )
- Use for or while loops

```
input_value = 4
for x in
    print
```

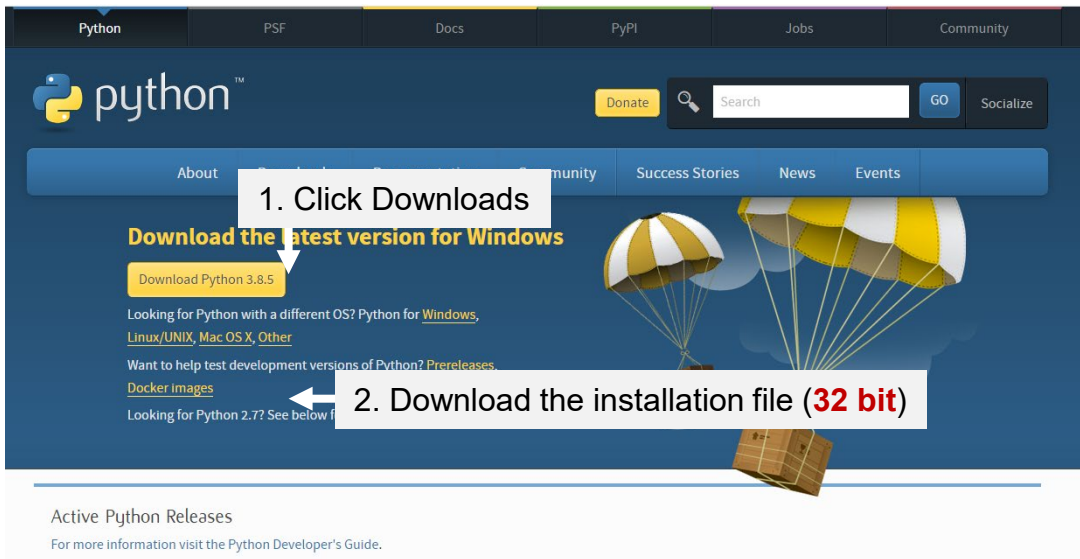
```
1 ^ 4 is 1
2 ^ 4 is 16
3 ^ 4 is 81
4 ^ 4 is 256
5 ^ 4 is 625
6 ^ 4 is 1296
7 ^ 4 is 2401
8 ^ 4 is 4096
9 ^ 4 is 6561
10 ^ 4 is 10000
```

```
input_value = 2
x = 0
while
    print
```

```
1 ^ 2 is 1
2 ^ 2 is 4
3 ^ 2 is 9
4 ^ 2 is 16
5 ^ 2 is 25
6 ^ 2 is 36
7 ^ 2 is 49
8 ^ 2 is 64
9 ^ 2 is 81
10 ^ 2 is 100
```

# Install Python

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



If your computer supports 64 bit, then download 'Windows x86-64 executable installer'

## Python 3.8.5 - July 20, 2020

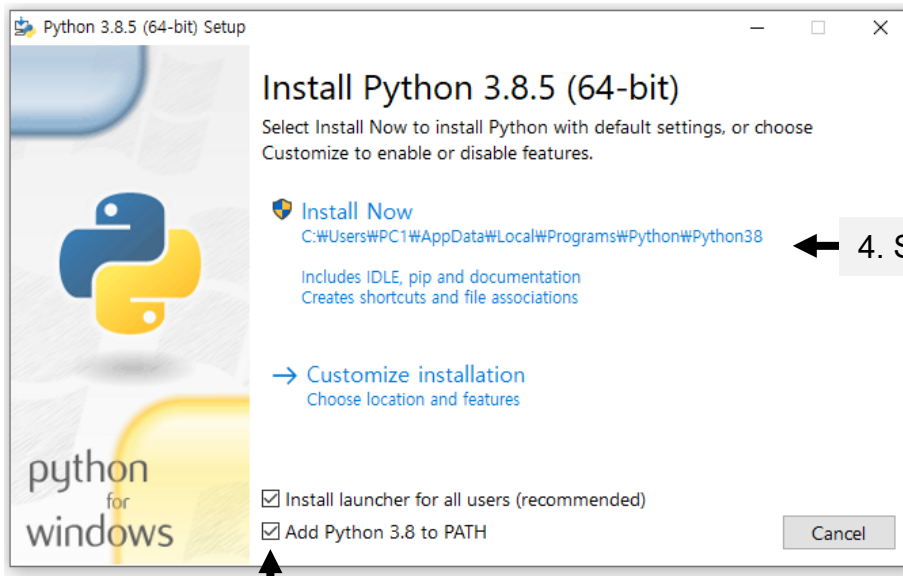
**Note that Python 3.8.5 cannot be used on Windows XP or earlier.**

- Download [Windows help file](#)
- Download [Windows x86-64 embeddable zip file](#)
- Download [Windows x86-64 executable installer](#)
- Download [Windows x86-64 web-based installer](#)
- Download [Windows x86 embeddable zip file](#)
- Download [Windows x86 executable installer](#)
- Download [Windows x86 web-based installer](#)

2. Download the installation file (64 bit)

# Install Python

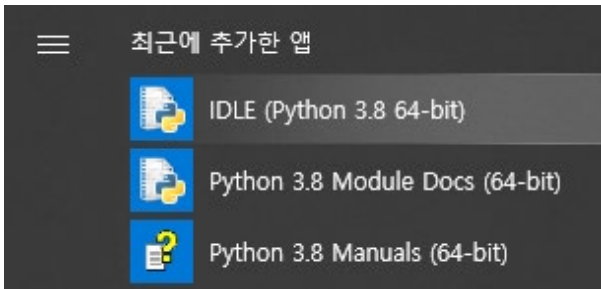
Run the installation file



4. Start install

3. Click this check box

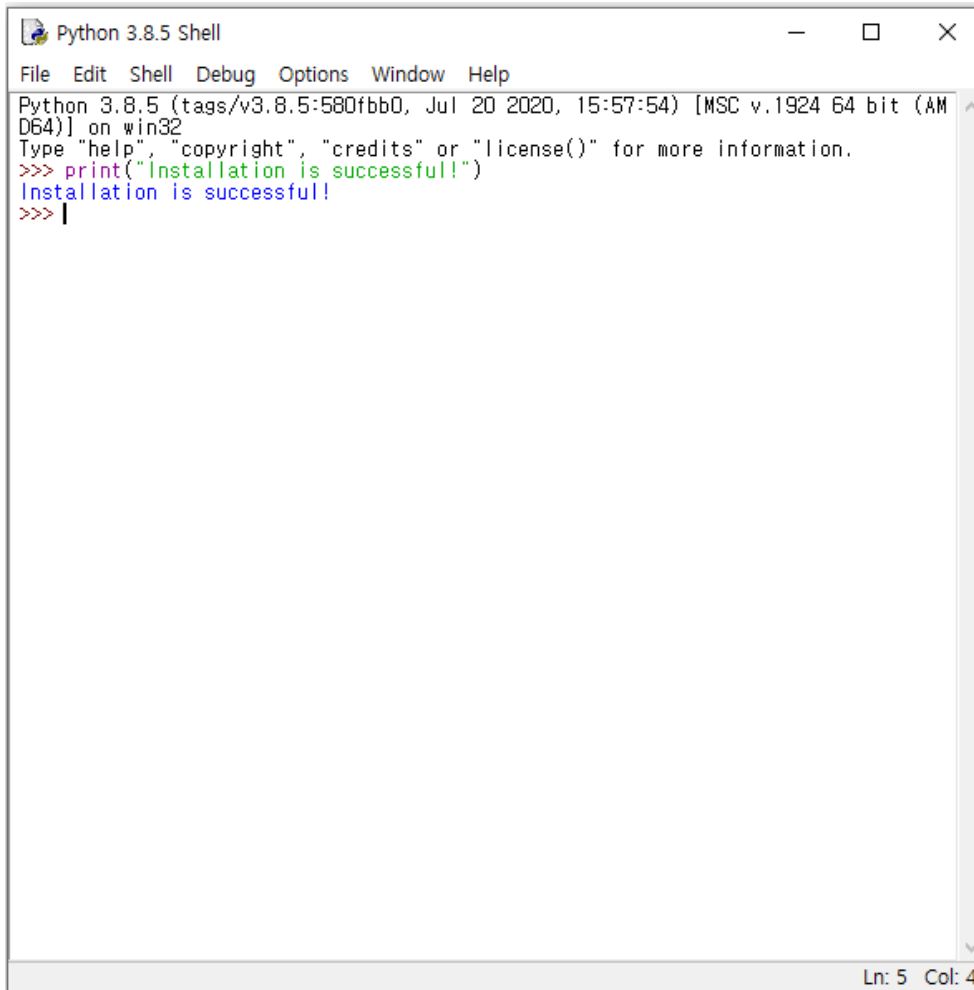
Run IDLE (Python 3.8)



# Install Python

If you can see this window, then the installation of Python is successful.

Type `print("Installation is successful!")` in prompt (`>>>`), then '`Installation is successful!`' will be printed.

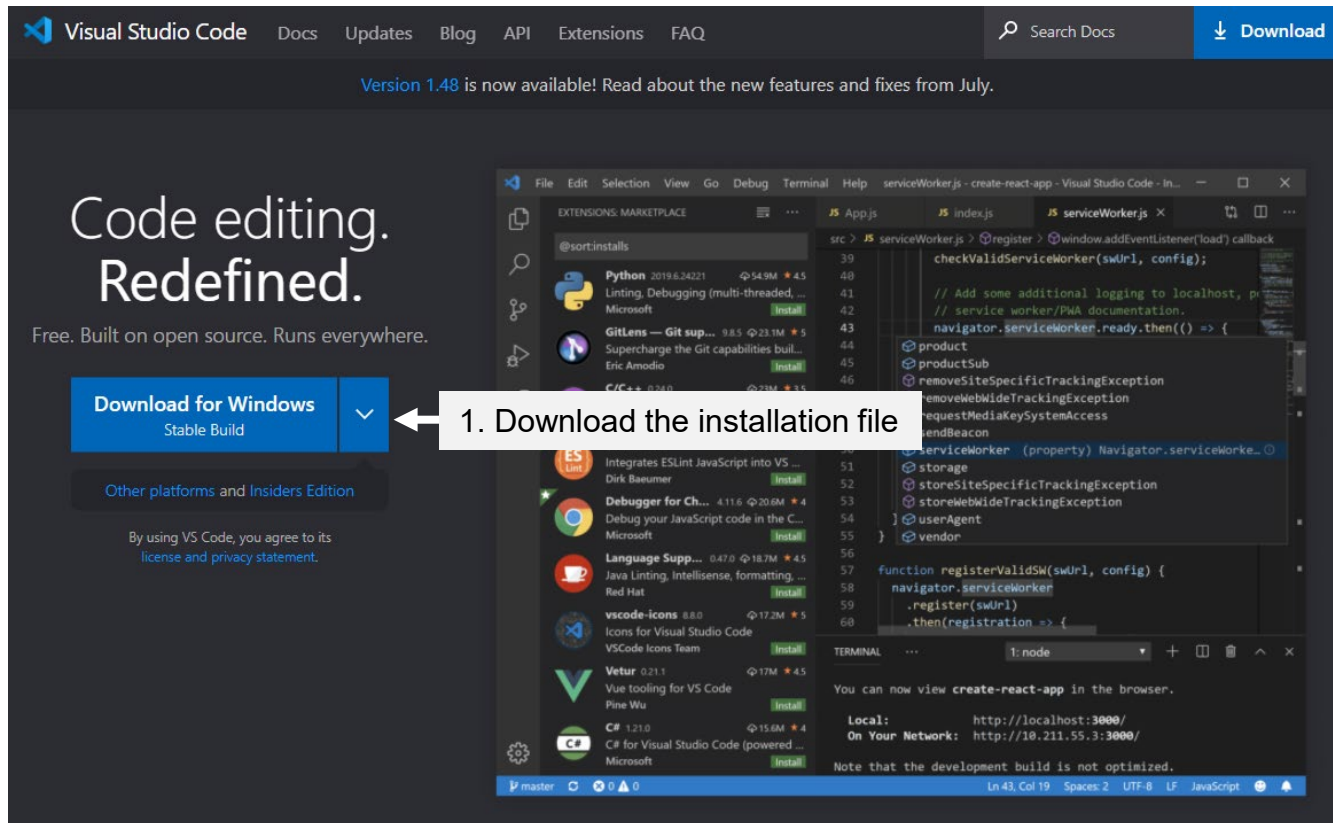


```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Installation is successful!")
Installation is successful!
>>> |
```

Ln: 5 Col: 4

# Install Text Editor

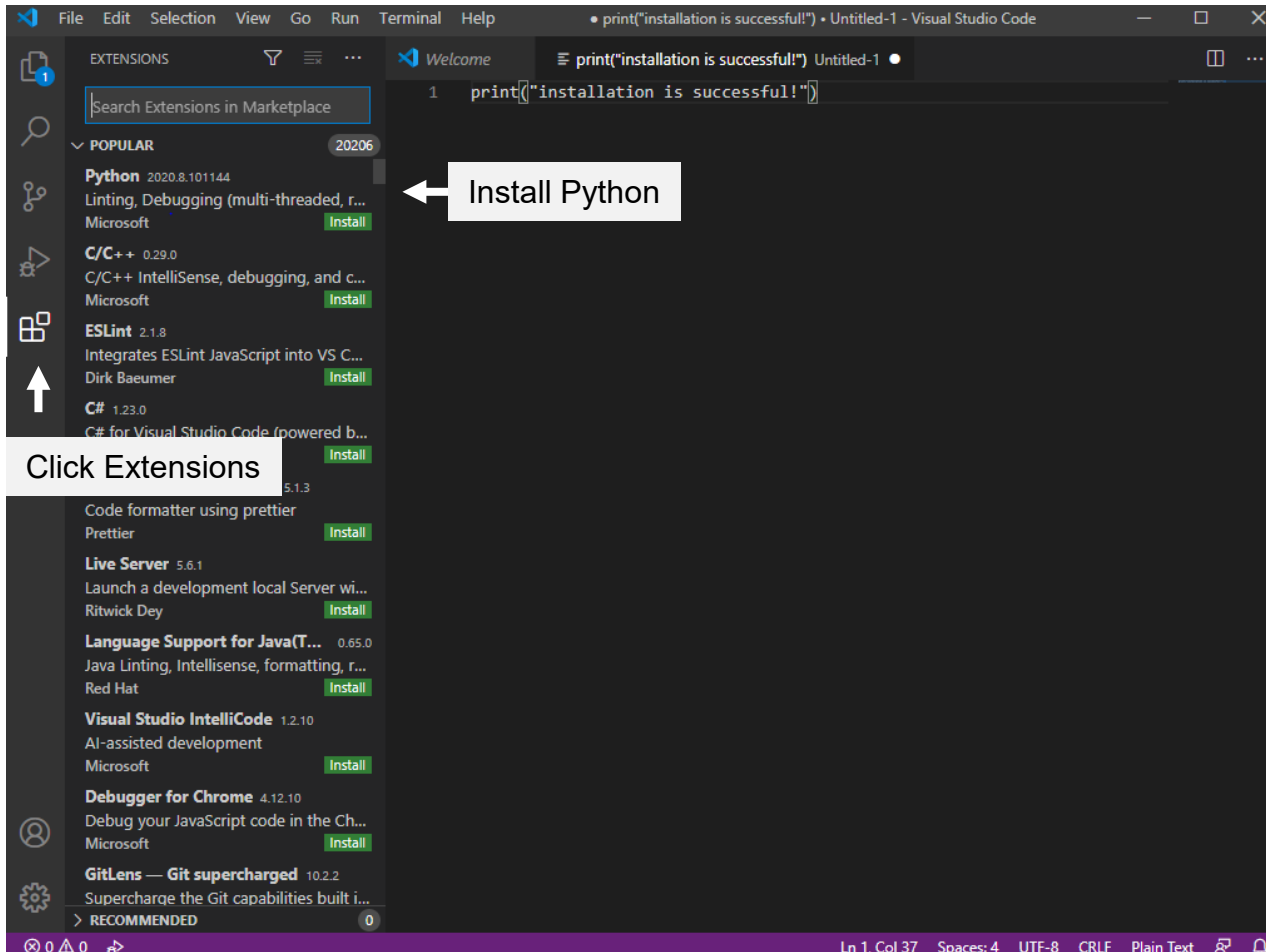
Download Visual Studio Code: <https://code.visualstudio.com/>



Install Visual Studio Code by running the installation file.

# Install Text Editor

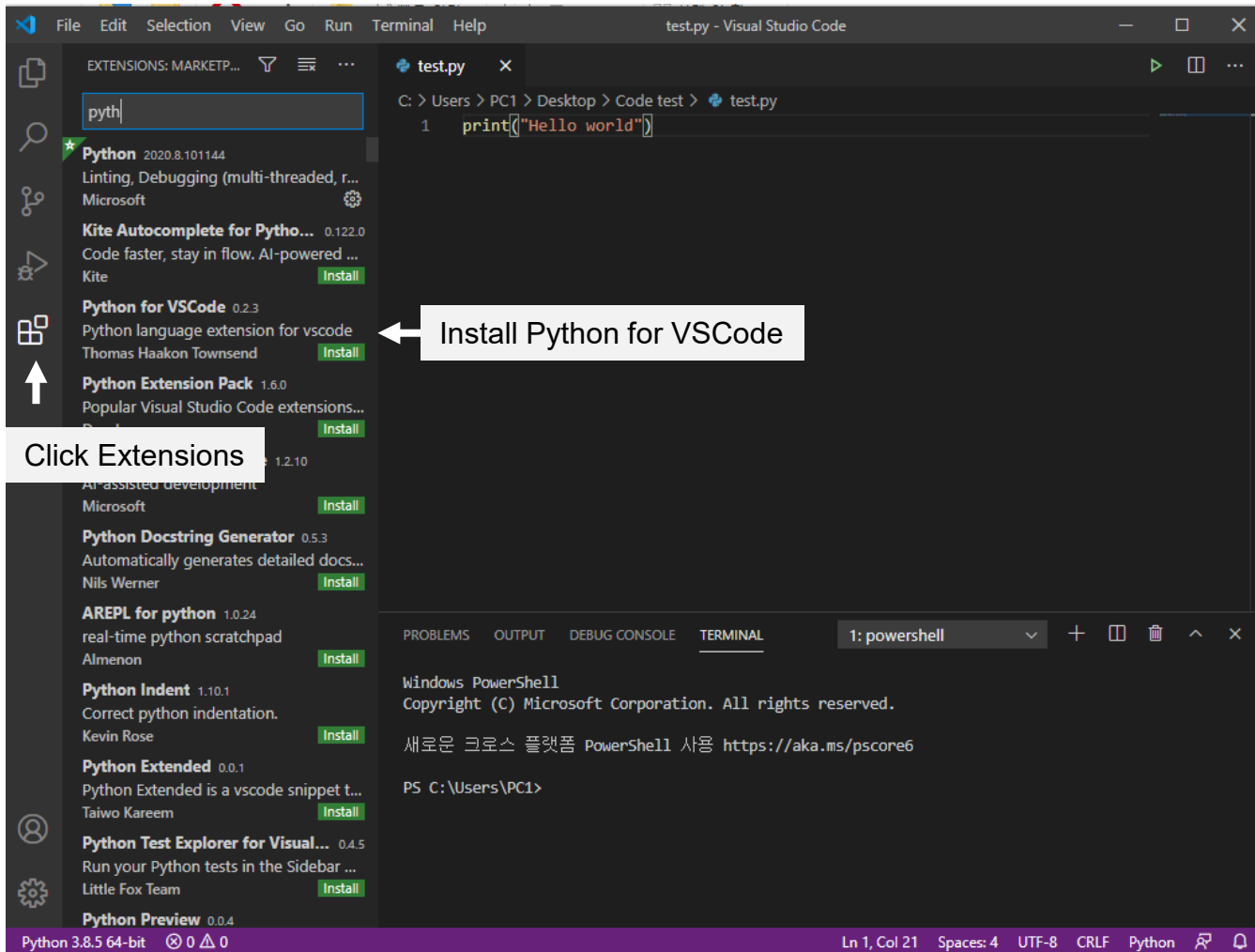
Download Visual Studio Code: <https://code.visualstudio.com/>



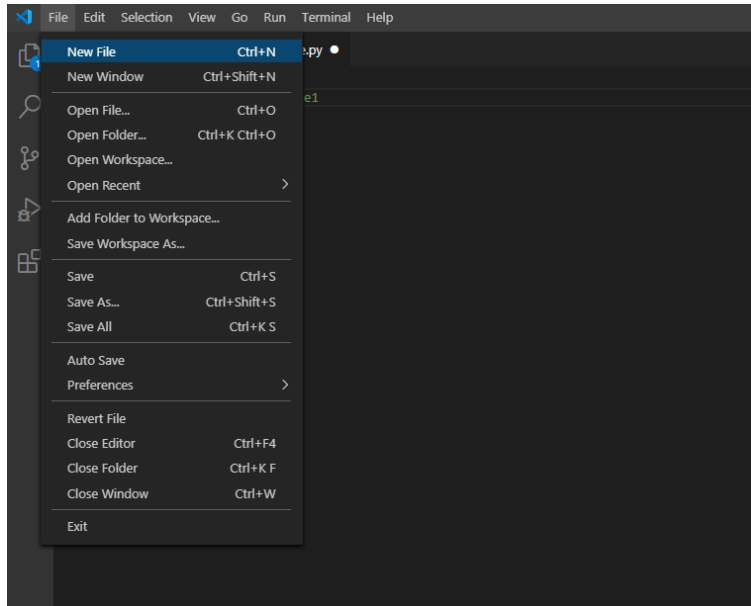


# Install Text Editor

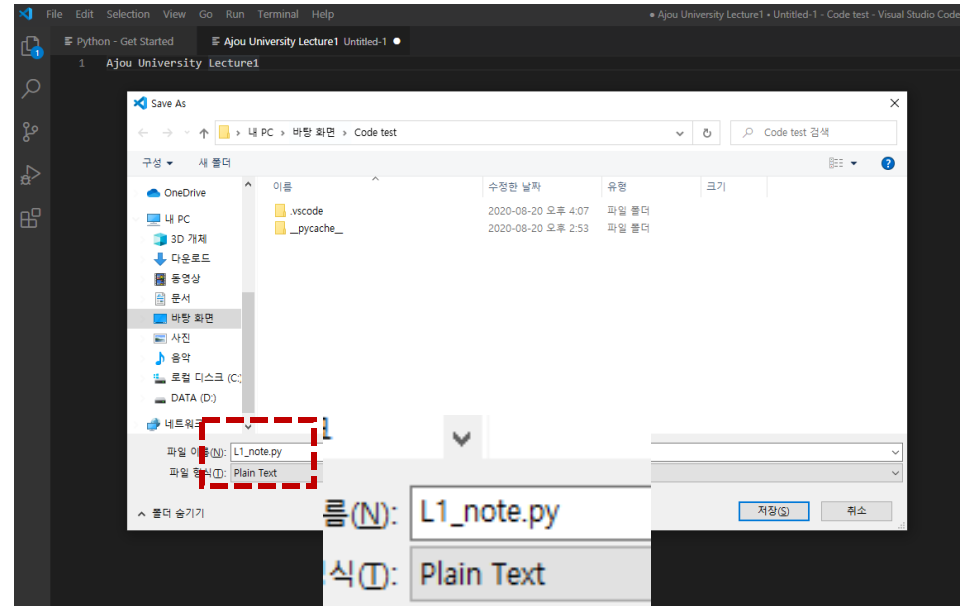
Download Visual Studio Code: <https://code.visualstudio.com/>



# Install Text Editor

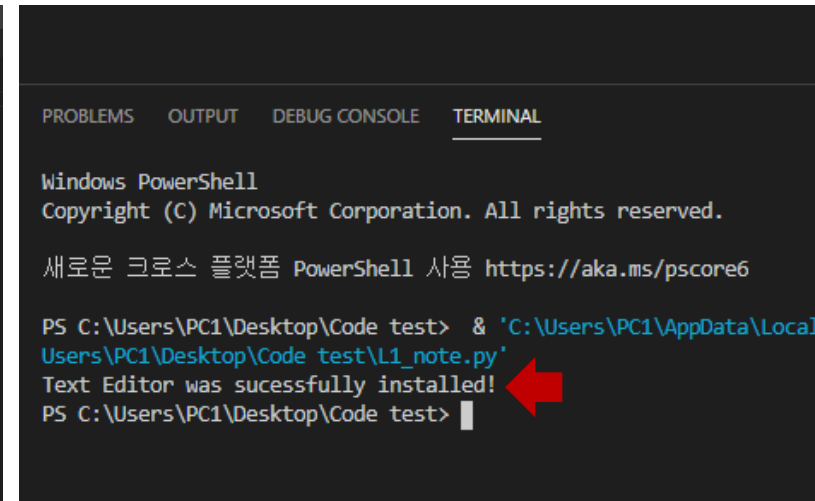
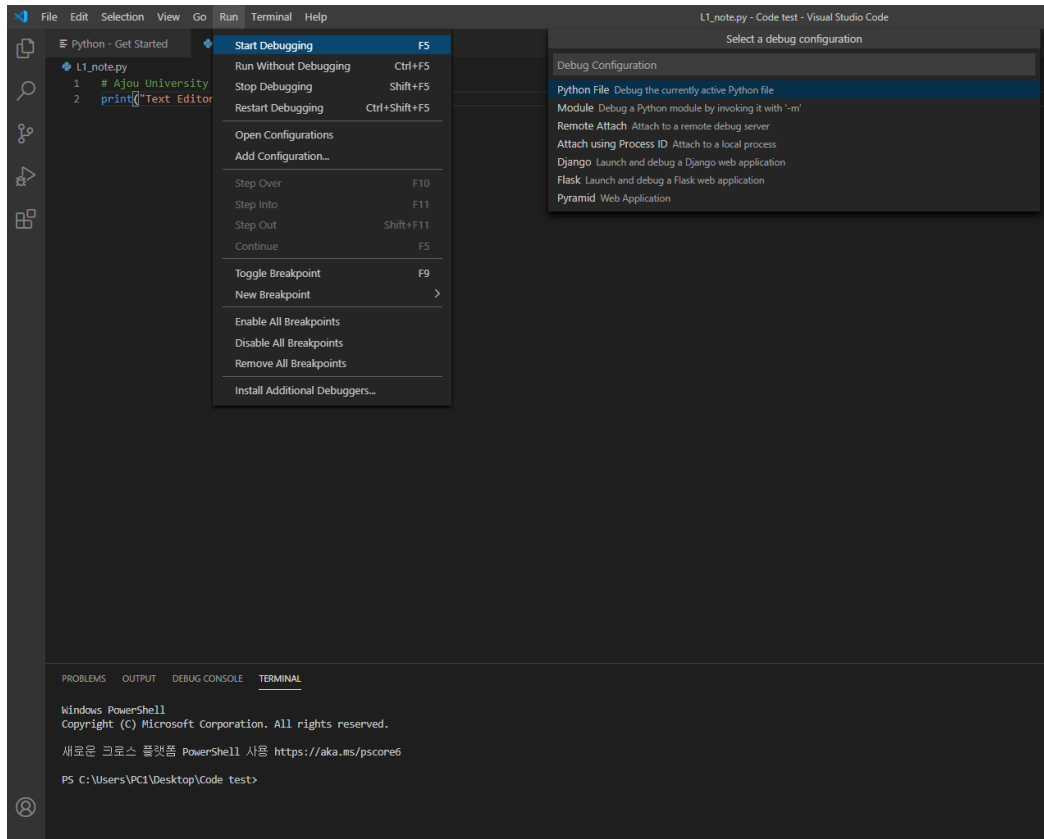


1. Make a new file



2. Save it as {file\_name}.py

# Install Text Editor



3. Type this sentence  
`printf("Text you want to print")`

4. Press 'F5' or Click Run->Start Debugging

5. Select Python File Debug the currently active Python file

(If VScode shows any message that you need to install additional files, then just install it!)

