스마트 모빌리티 프로그래밍

Ch 3. 파이썬 프로그램 실행 제어, 파이썬 프로그램 디버깅

영남대학교 정보통신공학과

교수 김 영 탁

(Tel : +82-53-810-2497; E-mail : ytkim@yu.ac.kr)

Outline

- ◆ 프로그램 실행 제어, 조건식과 조건문
- ◆ while-반복문
- ◆ for-반복문
- ◆ 반복문 블록 내부의 break, continue
- ◆ 파이썬 프로그램의 디버깅
- ◆ VS Code를 사용한 파이썬 프로그램 디버깅

프로그램 실행 제어 - 조건문, 조건식
(if, if-elif-else)

프로그램 실행제어

◆ 프로그램 실행 제어

- 프로그램 실행 중간의 다양한 상황에 따라 다른 기능을 수행할 수 있도록 구성

◆ 프로그램 실행 제어 구조

- 조건식 (conditional expression), 조건문(conditional statement)
- 반복문 – while-loop, for-loop
- 예외처리

조건식 관련 연산자

◆ 조건식 관련 연산자

연산자의 분류	연산자	의미, 예
관계연산자 (relation-ship)	>	a > b : a가 b보다 크면 True, 아니면 False
	>=	a >= b : a가 b보다 같거나 크면 True, 아니면 False
	<	a < b : a가 b보다 작으면 True, 아니면 False
	<=	a <= b : a가 b보다 같거나 작으면 True, 아니면 False
	==	a == b : a와 b가 같으면 True, 아니면 False
	!=	a != b : a와 b가 다르면 True, 아니면 False
논리연산자 (logical)	and (논리 곱)	A and B : A와 B가 모두 True이면 True, 아니면 False
	or (논리 합)	A or B : A나 B 둘 중 하나가 True이면 True, 아니면 (즉, A와 B모두 False이면) False
	not (논리 역)	not A : A가 True이면 False, A가 False이면 True
Ternary selection	x if condition else y	조건에 따라 선택 max = x if x > y else y; (만약 x가 y보다 크면 x를 선택, 아니면 y를 선택)

조건식의 표현

◆ 논리 연산자를 사용한 조건식의 표현

주어진 조건	산술 연산자를 사용한 수식의 표현
성적 (score)이 90보다 같거나 높고, 95보다 낮은 경우	if 90 <= score < 95: print(“Your grade is A“)

윤년 (leap year)의 조건:
연도가 4의 배수이며 100의 배수가 아니거나,
또는 400의 배수이면 윤년
	if ((year % 4 == 0) and (year % 100 != 0)) or(year % 400 == 0):
	print(“Year(%d) is a leap year“%(year))
	else:
	print(“Year(%d) is not a leap year“%(year))

기온이 30도 이상이며, 날씨가 화창할 때	if ((temp >= 30) and (weather == “SUNNY”)): print(“It’s good for picnic !!”)

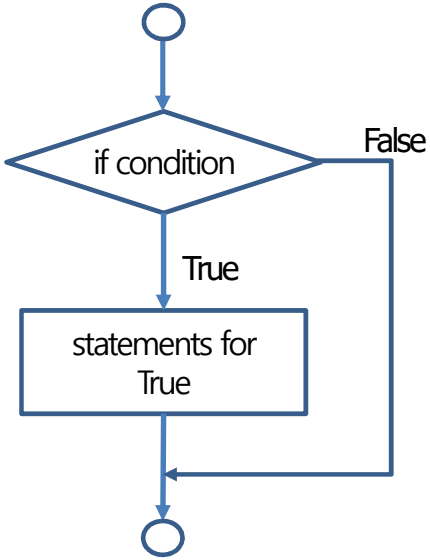
if

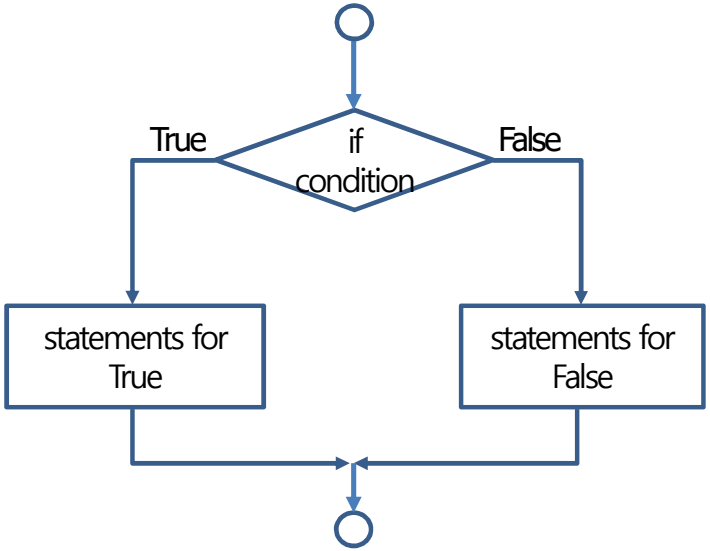
◆ Simple Branch

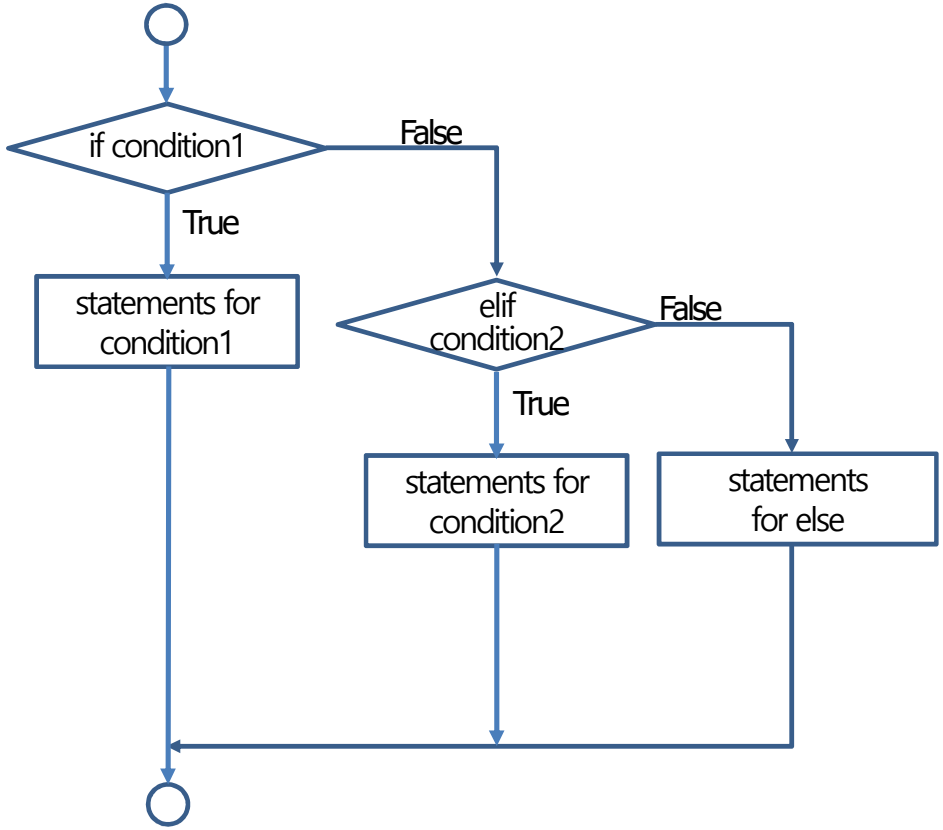
```
# simple program to input two integers and compare
x, y = map(int, input('input two integers (x, y) to compare : ').split())
if x == y:
    print("x(%d) is equal to y(%d)"%(x, y))
if x < y:
    print("x(%d) is less than y(%d)"%(x, y))
if x > y:
    print("x(%d) is greater than y(%d)"%(x, y))
```

```
============= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
mpleCondition_if.py =============
input two integers (x, y) to compare : 3 5
x(3) is less than y(5)
>>>
============= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
mpleCondition_if.py =============
input two integers (x, y) to compare : 5 7
x(5) is less than y(7)
>>>
============= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
mpleCondition_if.py =============
input two integers (x, y) to compare : 7 5
x(7) is greater than y(5)
>>>
============= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
mpleCondition_if.py =============
input two integers (x, y) to compare : 5 5
x(5) is equal to y(5)
```

if, if-else, if-elif-else statement







if ~ else

◆ Conditional branch with if - else

```
# conditional branch with if - else
x, y = map(int, input('input two integers (x, y) : ').split())
if x>y:
    Max = x
    Min = y
else:
    Max = y
    Min = x
print("x = %d, y = %d"%(x, y))
print("Max = %d, Min = %d"%(Max, Min))
```

```
==================== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
input two integers (x, y) : 3 5
x = 3, y = 5
Max = 5, Min = 3
>>>
==================== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
input two integers (x, y) : 7 5
x = 7, y = 5
Max = 7, Min = 5
```

if ~ elif ~ else

◆ Conditional branch with if - else

```
# conditional branch with if - elif - else
score = int(input('course score [0..99] = '))
if 90 <= score <= 100:
    grade = 'A'
elif 80 <= score:
    grade = 'B'
elif 70 <= score:
    grade = 'C'
elif 60 <= score:
    grade = 'D'
else:
    grade = 'F'
print("score = %d, grade = %s" %(score, grade))
```

```
=============== RESTART: C:/YTK-PythonProg/2_3 if_elif_else.py ===============
course score [0..99] = 95
score = 95, grade = A
>>>
=============== RESTART: C:/YTK-PythonProg/2_3 if_elif_else.py ===============
course score [0..99] = 84
score = 84, grade = B
>>>
=============== RESTART: C:/YTK-PythonProg/2_3 if_elif_else.py ===============
course score [0..99] = 65
score = 65, grade = D
>>>
=============== RESTART: C:/YTK-PythonProg/2_3 if_elif_else.py ===============
course score [0..99] = 50
score = 50, grade = F
>>> |
```

while 반복문

while 반복문 기본 구조

◆ while 반복문 기본 구조

- 조건식에서 사용되는 조건의 초기값 설정
- 조건식이 만족하는 동안 while-반복구문 실행
- while 반복구문 내부에서 조건식의 update가 반드시 있어야 함

```
조건의 초기값 설정
while 조건식 (condition):
    반복 구문
    반복 구문
    . . . .
    반복 구분
    조건의 갱신 (update)
```

while loop

◆ while-loop

```
# while-loop

L = list()
print("Input integers (-1 to end)")
x = int(input("data : "))
n = 0
while x >= 0:
    L.append(x)
    n += 1
    x = int(input("data : "))

print("Input data : ", L)
```

```
================ RESTART: C:/YTK-Py
Input integers (-1 to end)
data : 2
data : 3
data : 4
data : 5
data : 6
data : 7
data : -1
Input data :  [2, 3, 4, 5, 6, 7]
>>>
```

while-반복문과 break, continue

반복문	설 명
condition initialize **while** condition: statements condition update	▪ 먼저 while 반복문 조건식의 초기화를 실행 ▪ 조건식의 연산 결과가 True이면 while 반복 구문을 실행 ▪ while 반복 구문 내에서 조건식을 update

while condition1:
if condition2:
break
if condition3:
continue
statements
	▪ 만약 condition1이 True이면 while 반복 구문 실행
	▪ 만약 conditon2가 True이면 while 반복문을 중단하고 빠져 나감
	▪ 만약 condition3가 True이면 continue 이후 구간을 생략하고 while 반복문을 계속 실행

```
while True:
    . . . . .
    d = int(input("data : "))
    if d == -1:
        break;
    sum = sum + d
    . . . . .
```

(a) break in while-loop

```
while True:
    . . . . .
    d = int(input("data : "))
    if d < 0:
        continue;
    sum = sum + d
    . . . . .
```

(b) continue in while-loop

while-loop구조의 데이터 입력, 리스트 저장, 통계 분석

```
# while_loop with list and finding max and min

n = int(input("Input the number of data to process: "))
print("Input %d integers"%(n))
L = [] # create an empty list
count = 0
while count < n:
    d = int(input())
    L.append(d)
    if count == 0:
        Max = Min = d
    else:
        if d > Max:
            Max = d
        if d < Min:
            Min = d
    count = count + 1
print("Input data : ", L)
print("Max : ", Max)
print("Min : ", Min)
```

```
Input the number of data to process: 5
Input 5 integers
1
9
-5
2
-9
Input data :  [1, 9, -5, 2, -9]
Max :  9
Min :  -9
```

Sentinel 데이터를 사용한 while 반복문 제어

◆ Sentinel 데이터

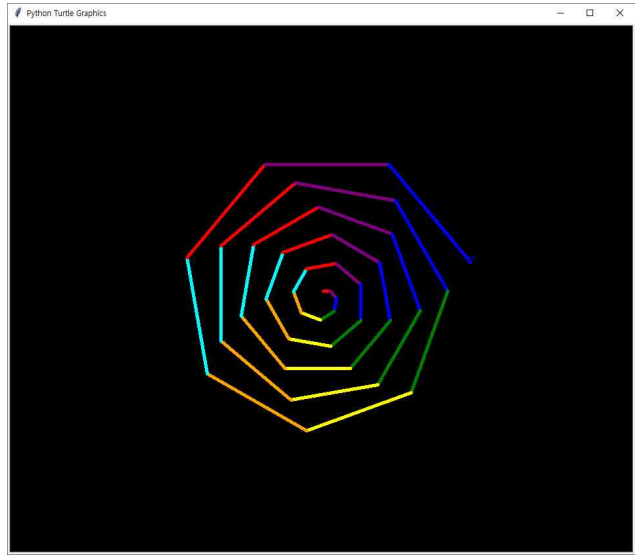
- 입력 데이터의 마지막을 표시하는 특별한 데이터
- 정상적인 데이터 입력에서 사용되지 않는 값을 사용

```
# while-loop

L = list()
while True:
    data = int(input("Input positive integers (-1 to stop) : "))
    if data == -1 :
        break
    L.append(data)
    print("L = ", L)
```

```
Input positive integers (-1 to stop) : 1
L =  [1]
Input positive integers (-1 to stop) : 2
L =  [1, 2]
Input positive integers (-1 to stop) : 3
L =  [1, 2, 3]
Input positive integers (-1 to stop) : 4
L =  [1, 2, 3, 4]
Input positive integers (-1 to stop) : 5
L =  [1, 2, 3, 4, 5]
Input positive integers (-1 to stop) : -1
```

while-loop과 터틀 그래픽

```
#turtle graphic and while-loop
import turtle
colors = ["red", "purple", "blue", "green", "yellow",\
          "orange", "cyan", "white", “violet", "brown"]

t = turtle.Turtle()

turtle.bgcolor("black")
t.width(5)
num_vertices = int(input("input num_vertices = "))
length = 10
count = 0
turn_angle = (360 // num_vertices) - 1
while length < 200:
    t.pencolor(colors[count % num_vertices])
    t.forward(length)
    t.right(turn_angle)
    length += 5
    count += 1
```



for 반복문

for 반복문

◆ for 반복문의 기본 구조

- 변수가 지정된 영역에 있는 경우 반복구문 수행
- 변수는 반복문을 실행할 때 마다 갱신되어야 함
- 변수가 주어진 조건을 만족하지 않을 때 for 반복문을 벗어남

```
for 변수 (variable) in 시퀀스 객체 (range, list, tuple, str, bytes, bytearray 등) :
    반복 구문
    반복 구문
    . . . .
    반복 구분
```

Control loop with for

◆ for loop with range()

```
# control loop with for
n = int(input('Input n to calculate sum of [0..n] : '))
nSum = 0
for i in range(0, n+1): #sSum = sum(range(0, n+1))
    nSum += i
print("Sum of [0..%d] = %d" %(n, nSum))
```

```
================== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1
Input n to calculate sum of [0..n] : 1
Sum of [0..1] = 1
>>>
================== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1
Input n to calculate sum of [0..n] : 10
Sum of [0..10] = 55
```

for-loop with str

```
# for-loop with str
testStr = "Python, 파이선, 12345, *?!!"
nAlphabet = nHangul = nNumber = nSymbol = nOther = 0
print('Test string: ', testStr)

for c in testStr:
    #print(c)
    if 0x41<= ord(c) <=0x5A or 0x61 <= ord(c) <= 0x7A:
        print(c, ' : Alphabet')
        nAlphabet += 1
    elif 0xAC00 <= ord(c) <= 0xD7A3:
        print(c, ' : Hangul')
        nHangul +=1
    elif 0x30 <= ord(c) <= 0x39:
        print(c, ' : Number')
        nNumber += 1
    elif 0x21 <= ord(c) <= 0x2F or 0x3A <= ord(c) <= 0x40:
        print(c, ' : Symbol')
        nSymbol += 1
    else:
        print(c, ' : Other')
        nOther += 1
totalChar = nAlphabet + nHangul + nNumber + nSymbol + nOther

print('Total %d characters'%totalChar)
print('nAlphabet: ', nAlphabet)
print('nHangul: ', nHangul)
print('nNumber: ', nNumber)
print('nSymbol: ', nSymbol)
print('nOther: ', nOther)
```

```
================ RESTART: C:\YTK-Progs\2(
Test string:  Python, 파이선, 12345, *?!!
P  : Alphabet
y  : Alphabet
t  : Alphabet
h  : Alphabet
o  : Alphabet
n  : Alphabet
,  : Symbol
   : Other
파  : Hangul
이  : Hangul
선  : Hangul
,  : Symbol
   : Other
1  : Number
2  : Number
3  : Number
4  : Number
5  : Number
,  : Symbol
   : Other
*  : Symbol
?  : Symbol
!  : Symbol
!  : Symbol
Total 24 characters
nAlphabet:  6
nHangul:  3
nNumber:  5
nSymbol:  7
nOther:  3
>>>
```

Getting Max and Min from list

```
#Find Max and Min from List
L = [70, 85, 15, 55, 30, 90, 45, 10, 5, 60]
Max = Min = L[0]
for n in L:
    if Min > n:
        Min = n
    if Max < n:
        Max = n
print("Data (before data change) : %s"%L)
print("Min: %d, Max: %d"%(Min, Max))

#data change with index of list
for i in range(len(L)):
    L[i] *= 5
Min = min(L)
Max = max(L)
print("Data (after data changes) : %s"%L)
print("Min: %d, Max: %d"%(Min, Max))
```

```
>>>
=========== RESTART: C:/YTK-PythonProg/2_6 Max and Min of List.py ===========
Data (before data change) : [70, 85, 15, 55, 30, 90, 45, 10, 5, 60]
Min: 5, Max: 90
Data (after data changes) : [350, 425, 75, 275, 150, 450, 225, 50, 25, 300]
Min: 25, Max: 450
>>>
```

for-반복문과 break, continue

반복문	설 명
for n in sequence_type_object: statements	▪ 시퀀스 객체에 있는 원소들을 차례로 사용하면서 for 반복 구문을 실행

for n in sequence_type_object:
if condition1:
break
if condition2:
continue
statements
	▪ 시퀀스 객체에 있는 원소들을 차례로 사용하면서 for 반복 구문을 실행
	▪ 만약 condition1이 True이면 for 반복문을 중단하고 빠져 나감
	▪ 만약 conditon2가 True이면 continue 이후 구간을 생략하고 for 반복문을 계속 실행

```
for n in range(100):
    . . . . .
    d = int(input("data : "))
    if d == -1:
        break;
    sum = sum + d
    . . . . .
```

(a) break in for-loop

```
for n in range(100):
    . . . . .
    d = int(input("data : "))
    if d < 0:
        continue;
    sum = sum + d
    . . . . .
```

(b) continue in for-loop

break, continue

◆ Example of for-loop with break and continue

```
#for_loop with break and continue

n = int(input("Input number of data to process: "))
L = list()
sum = 0
print("Input %d non-negative integers"%(n))
for i in range(n):
    d = int(input())
    if d == 0:
        continue
    elif d < 0:
        break
    L.append(d)
    sum += d
print("Input data : ", L)
print("Sum = ", sum)
```

```
====== RESTART: C:/YTK-PythonProg/3_
Input number of data to process: 5
Input 5 non-negative integers
2
3
0
4
5
Input data :  [2, 3, 4, 5]
Sum =  14
>>>
====== RESTART: C:/YTK-PythonProg/3_
Input number of data to process: 5
Input 5 non-negative integers
2
-1
Input data :  [2]
Sum =  2
>>>
```

for-loop with list, set and dict

◆ for-loop with list, set and dict

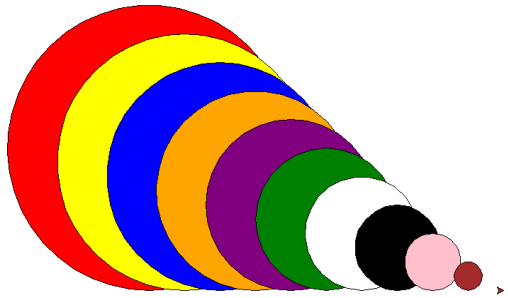
```
# for-loop with list, set and dict
L = [1, 2, 3, 4, 5] #list
print('List L: ', L)
for d in L:
    print("{}".format(d), end=' ')
print()

S = {'A', 'B', 'C', 'D', 'E'} #set
print('Set S: ', S)
for a in S:
    print("{}".format(a), end= ' ')
print()

D = {1:'January', 2:'February', 3:'March', 4:'April', 5:'May'} #dict
print('Dictionary D:\n ', D)
for key, value in D.items():
    print("key = {0}: value = {1}".format(key, value))
```

```
List L:  [1, 2, 3, 4, 5]
1 2 3 4 5
Set S:  {'C', 'E', 'B', 'D', 'A'}
C E B D A
Dictionary D:
  {1: 'January', 2: 'February', 3: 'March', 4: 'April', 5: 'May'}
key = 1: value = January
key = 2: value = February
key = 3: value = March
key = 4: value = April
key = 5: value = May
```

for-loop과 터틀 그래픽

◆ for-loop을 사용한 터틀 그래픽 도형 반복 그리기 실행

```
# drawing with for-loop
import turtle
t = turtle.Turtle()
t.shape("classic")

radius = 200
start_pos = (-250, 0)
t.up()
t.goto(start_pos)
t.down()
color_list = ["red", "yellow", "blue", "orange", "purple",\
              "green", "white", "black", "pink", "brown"]
for color in color_list:
    t.fillcolor(color)
    t.begin_fill()
    t.circle(radius)
    t.end_fill()
    radius -= 20
    t.up()
    t.forward(50)
    t.down()
```



for 반복문과 iter(), next(), enumerate()

◆ for loop with iter(), next(), enumerate() (1)

```
#iter(), next(), enumerate()
Months = ['January', 'February', 'March', 'April', \
          'May', 'June', 'July', 'August',\
          'September', 'October', 'November', 'December']
it = iter(Months) #iterator
for i in range(len(Months)):
    d = next(it)
    print(d, end=' ')
print() #new line
print() #new line

i = 0
for item in enumerate(Months):
    print("{0}".format(item), end=' ')
    i += 1
    if i%4 == 0:
        print()
print() #new line

i = 0
for item in enumerate(Months, 1):
    print(item, end =' ')
    i += 1
    if i%4 == 0:
        print()
print() #new line
```

```
>>>
========== RESTART: C:/YTK-PythonProg/2_9 iter, next, enumerate.py ==========
January February March April May June July August September October November December

(0, 'January') (1, 'February') (2, 'March') (3, 'April')
(4, 'May') (5, 'June') (6, 'July') (7, 'August')
(8, 'September') (9, 'October') (10, 'November') (11, 'December')

(1, 'January') (2, 'February') (3, 'March') (4, 'April')
(5, 'May') (6, 'June') (7, 'July') (8, 'August')
(9, 'September') (10, 'October') (11, 'November') (12, 'December')

>>> |
```

for-loop과 enumerate()

◆ for-loop with enumerate()

```
# for-loop with enumerate()
Month_name = ["January", "February", "March", "April",\
              "May", "June", "July", "August",\
              "September", "October", "November", "December"]

#case A
for item in enumerate(Month_name):
    print(item)
print()

#case B
for item in enumerate(Month_name, 1):
    print(item)
print()

#case C
for i, item in enumerate(Month_name, 1):
    print("%2d"%i, item)
print()
```

(a) Case A

```
(0, 'January')
(1, 'February')
(2, 'March')
(3, 'April')
(4, 'May')
(5, 'June')
(6, 'July')
(7, 'August')
(8, 'September')
(9, 'October')
(10, 'November')
(11, 'December')
```

(b) Case B

```
(1, 'January')
(2, 'February')
(3, 'March')
(4, 'April')
(5, 'May')
(6, 'June')
(7, 'July')
(8, 'August')
(9, 'September')
(10, 'October')
(11, 'November')
(12, 'December')
```

(c) Case C

```
 1 January
 2 February
 3 March
 4 April
 5 May
 6 June
 7 July
 8 August
 9 September
10 October
11 November
12 December
```

중첩된 for-loop

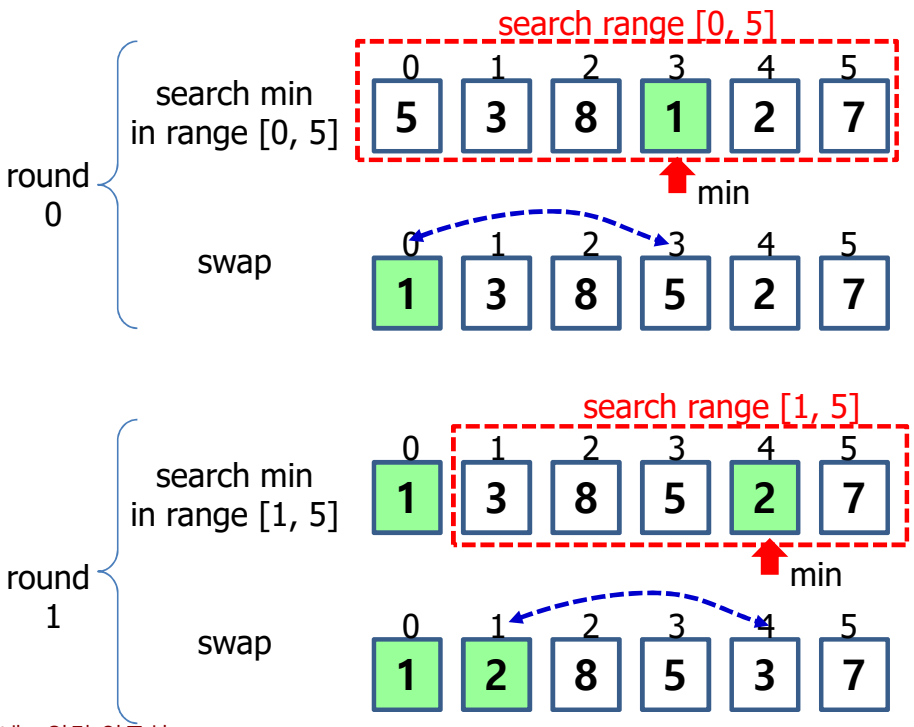
◆ 2중 for-loop

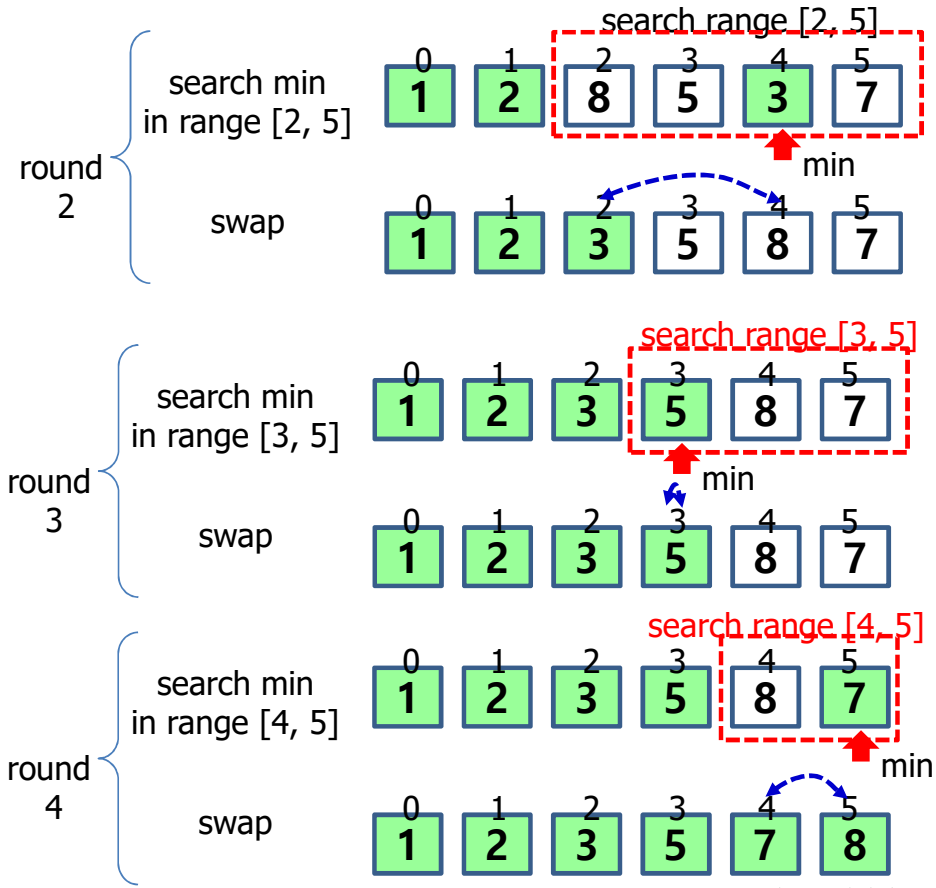
```
# nested for_loop that generates multiplication table

for i in range (1, 10):
    for j in range (1, 10):
        print("{0}x{1} ={2:>3}, ".format(i, j, i*j), end= '')
    print() # new line
```

```
============= RESTART: C:/YTK-PythonProg/2_8 nested for_loop.py =============
1x1 =  1, 1x2 =  2, 1x3 =  3, 1x4 =  4, 1x5 =  5, 1x6 =  6, 1x7 =  7, 1x8 =  8, 1x9 =  9,
2x1 =  2, 2x2 =  4, 2x3 =  6, 2x4 =  8, 2x5 = 10, 2x6 = 12, 2x7 = 14, 2x8 = 16, 2x9 = 18,
3x1 =  3, 3x2 =  6, 3x3 =  9, 3x4 = 12, 3x5 = 15, 3x6 = 18, 3x7 = 21, 3x8 = 24, 3x9 = 27,
4x1 =  4, 4x2 =  8, 4x3 = 12, 4x4 = 16, 4x5 = 20, 4x6 = 24, 4x7 = 28, 4x8 = 32, 4x9 = 36,
5x1 =  5, 5x2 = 10, 5x3 = 15, 5x4 = 20, 5x5 = 25, 5x6 = 30, 5x7 = 35, 5x8 = 40, 5x9 = 45,
6x1 =  6, 6x2 = 12, 6x3 = 18, 6x4 = 24, 6x5 = 30, 6x6 = 36, 6x7 = 42, 6x8 = 48, 6x9 = 54,
7x1 =  7, 7x2 = 14, 7x3 = 21, 7x4 = 28, 7x5 = 35, 7x6 = 42, 7x7 = 49, 7x8 = 56, 7x9 = 63,
8x1 =  8, 8x2 = 16, 8x3 = 24, 8x4 = 32, 8x5 = 40, 8x6 = 48, 8x7 = 56, 8x8 = 64, 8x9 = 72,
9x1 =  9, 9x2 = 18, 9x3 = 27, 9x4 = 36, 9x5 = 45, 9x6 = 54, 9x7 = 63, 9x8 = 72, 9x9 = 81,
>>> |
```

선택정렬(selection sort)

- 선택정렬(selection sort): 정렬이 안된 숫자들 중에서 최소값을 선택하여 배열의 첫 번째 요소와 교환
- 몇 개의 단계만 살펴보자.





for-loop 기반 선택 정렬 (Selection Sorting)

```
# selection sort with for-loop

L = [5, 3, 8, 1, 2, 7]
size = len(L)
print("L (initial) = ", L)
for i in range(size-1):
    min_idx = i;
    for j in range(i+1, size):
        if L[min_idx ] > L[j]:
            min_idx = j
    if (min_idx != i):
        L[min_idx], L[i] = L[i], L[min_idx]
    print("round{:2} - L : {}".format(i, L))
```

```
L(initial) =  [5, 3, 8, 1, 2, 7]
round 0 - L : [1, 3, 8, 5, 2, 7]
round 1 - L : [1, 2, 8, 5, 3, 7]
round 2 - L : [1, 2, 3, 5, 8, 7]
round 3 - L : [1, 2, 3, 5, 8, 7]
round 4 - L : [1, 2, 3, 5, 7, 8]
```

for-loop 기반 행렬 (Matrix) 연산

◆ 행렬의 덧셈과 뺄셈

- $C_{(3x3)} = A_{(3x3)} + B_{(3x3)}$
- $D_{(3x3)} = A_{(3x3)} - B_{(3x3)}$

$$\begin{bmatrix} c_{00}=a_{00}+b_{00} & c_{01}=a_{01}+b_{01} & c_{02}=a_{02}+b_{02} \\ c_{10}=a_{10}+b_{10} & c_{11}=a_{11}+b_{11} & c_{12}=a_{12}+b_{12} \\ c_{20}=a_{20}+b_{20} & c_{21}=a_{21}+b_{21} & c_{22}=a_{22}+b_{22} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix} + \begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix}$$

for-loop 기반 행렬 (Matrix) 연산

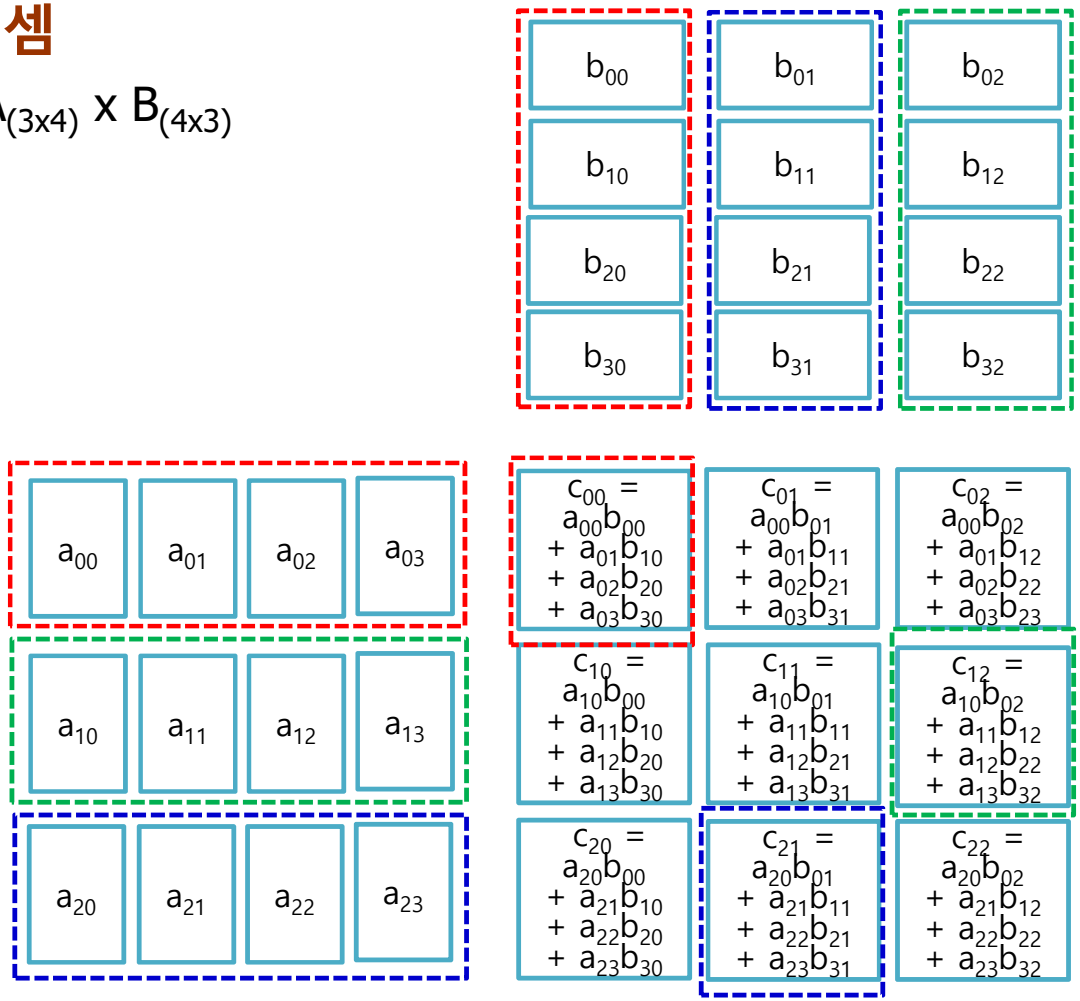
```
# nested for_loop for 2-dimensional lists' addition, subtraction

A = [[1,2,3], [4,5,6], [7,8,9]]
B = [[1,0,0], [0,1,0], [0,0,1]]
C = [[0,0,0], [0,0,0], [0,0,0]]
D = [[0,0,0], [0,0,0], [0,0,0]]

print("A : ", A)
print("B : ", B)
for i in range (3):
    for j in range (3):
        C[i][j] = A[i][j] + B[i][j]
print("A+B => ", C)

for i in range (3):
    for j in range (3):
        D[i][j] = A[i][j] - B[i][j]
print("A-B => ", D)
```

```
A :  [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
B :  [[1, 0, 0], [0, 1, 0], [0, 0, 1]]
A+B =>  [[2, 2, 3], [4, 6, 6], [7, 8, 10]]
A-B =>  [[0, 2, 3], [4, 4, 6], [7, 8, 8]]
```

for-loop 기반 행렬 (Matrix) 연산

◆ 행렬의 곱셈

- $C_{(3x3)} = A_{(3x4)} \times B_{(4x3)}$



for-loop 기반 행렬 (Matrix) 연산

```
# nested for_loop for Matrix multiplication

A = [[1, 2, 3, 4], [5, 6, 7, 8], [9, 0, 1, 2]] # 3 x 4
B = [[1, 0, 0], [0, 1, 0], [0, 0, 1], [1, 0, 0]] # 4 x 3
R = [[0,0,0], [0,0,0], [0,0,0]] # 3 x 3


print("A = ", A)
print("B = ", B)

for i in range (3):
    for j in range (3):
        R[i][j] = 0
        for k in range (4):
            R[i][j] += A[i][k] * B[k][j]
print("R = A*B = ", R)
```

```
A =  [[1, 2, 3, 4], [5, 6, 7, 8], [9, 0, 1, 2]]
B =  [[1, 0, 0], [0, 1, 0], [0, 0, 1], [1, 0, 0]]
R = A*B =  [[5, 2, 3], [13, 6, 7], [11, 0, 1]]
```

파이썬 예외 처리 (Exception Handling)

Exceptions in Python

◆ Exceptions in Python

- ArithmeticError
- AssertionError
- IndexError
- IOError
- NameError
- RuntimeError
- StopIteration
- SyntaxError
- SystemExit
- TypeError
- ZeroDivisionError

Exception Handling with try, except, finally

◆ Exception Error

```
# Exception Handling with try - finally

try:
    z = 10 / 0
finally:
    print("At final")
```

```
========== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 3-2
At final
Traceback (most recent call last):
  File "C:\YTK-Progs\2018 Book (Python)\ch 3-2 Control for,
    z = 10 / 0
ZeroDivisionError: division by zero
```

Exception Handling with try, except

◆ Example of try - except

```
# Exception handling with try, except, and finally

try:
    z = 10 / 0
except ZeroDivisionError as e:
    print("ZeroDivisionError:", e.args)
finally:
    print("At final.")
```

```
========= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 3-2
ZeroDivisionError: ('division by zero',)
At final.
>>> |
```

ArithmeticError, ZeroDivisionError

◆ ArithmeticError, ZeroDivisionError

```
# Exception handling with try, except, ArithmeticError, ZeroDivisionError

try:
    z = 10 / 0
except ArithmeticError as e:
    print("ArithmeticError:", e.args)
except ZeroDivisionError as e:
    print("ZeroDivisionError:", e.args)
finally:
    print("At final.")
```

```
======= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 3-2
ArithmeticError: ('division by zero',)
At final.
>>> |
```

Exception Handling with try, else

◆ try, else

```
# Exception handling with try, except, else and finally

try:
    z = 10 / 10
except ArithmeticError as e:
    print("ArithmeticError:", e.args)
except ZeroDivisionError as e:
    print("ZeroDivisionError:", e.args)
else:
    print("No exceptions")
finally:
    print("At final.")
```

```
...
========== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 3-2
No exceptions
At final.
>>> |
```

Exception Handling with raise, traceback

◆ raise, traceback

```
#Exception handling with raise, traceback

import sys
import traceback
try:
    raise Exception("Raised Exception")
except:
    exc_type, exc_value, exc_traceback = sys.exc_info()
    lines = traceback.format_exception(exc_type, exc_value, exc_traceback)
    print(line for line in lines)
finally:
    print("At final")
```

```
======= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 3-2
<generator object <genexpr> at 0x02B3B450>
At final
>>> |
```

assert

◆ try, assert

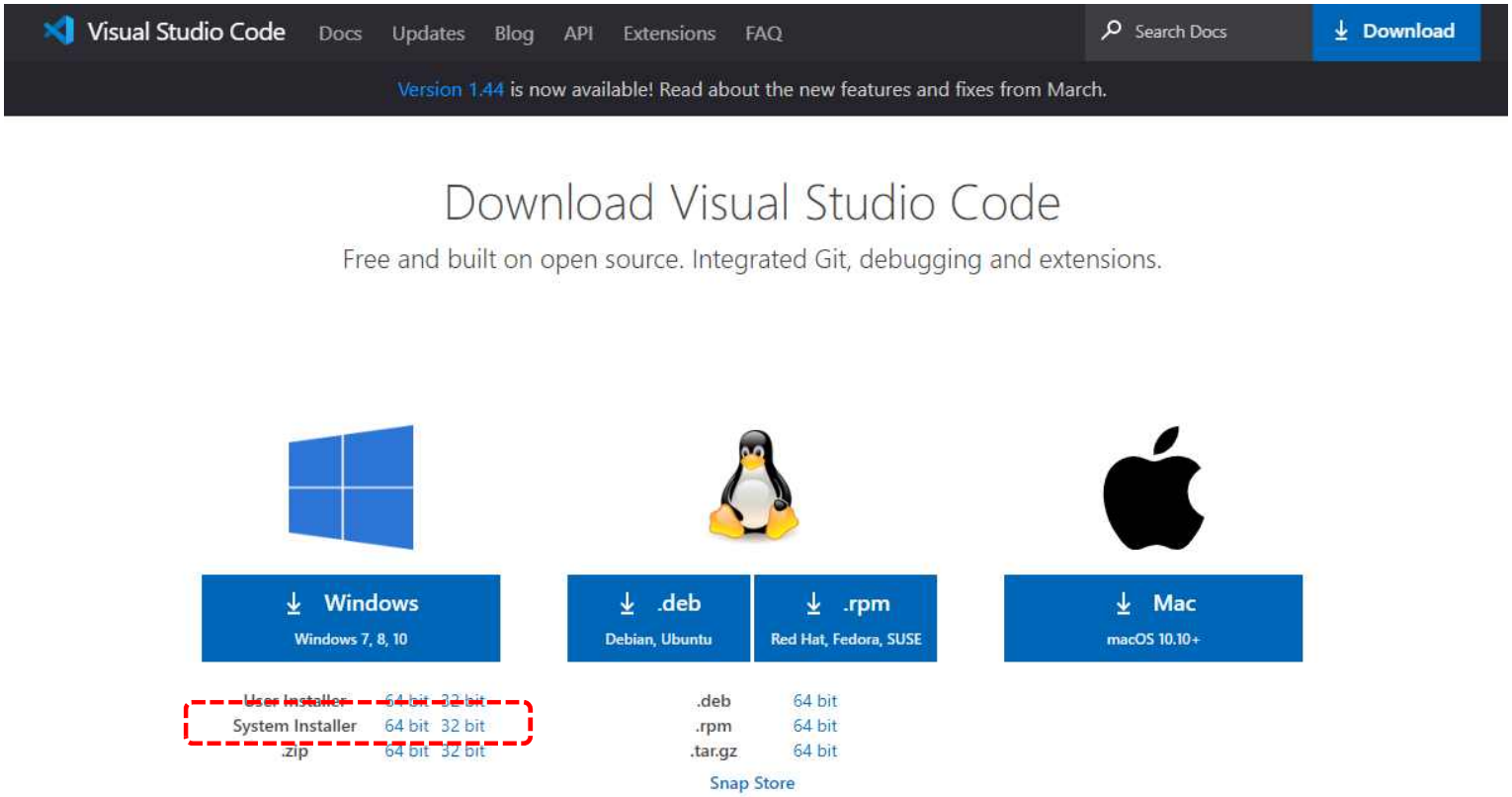
```
# Exception handling with try, assert

try:
    divisor = 0
    assert divisor != 0, "divisor must not be zero"
    z = 10 / divisor
except AssertionError as e:
    print('AssertionError : ', e.args)
finally:
    print("At final.")
```

```
========= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 3-2
AssertionError :  ('divisor must not be zero',)
At final.
>>> |
```

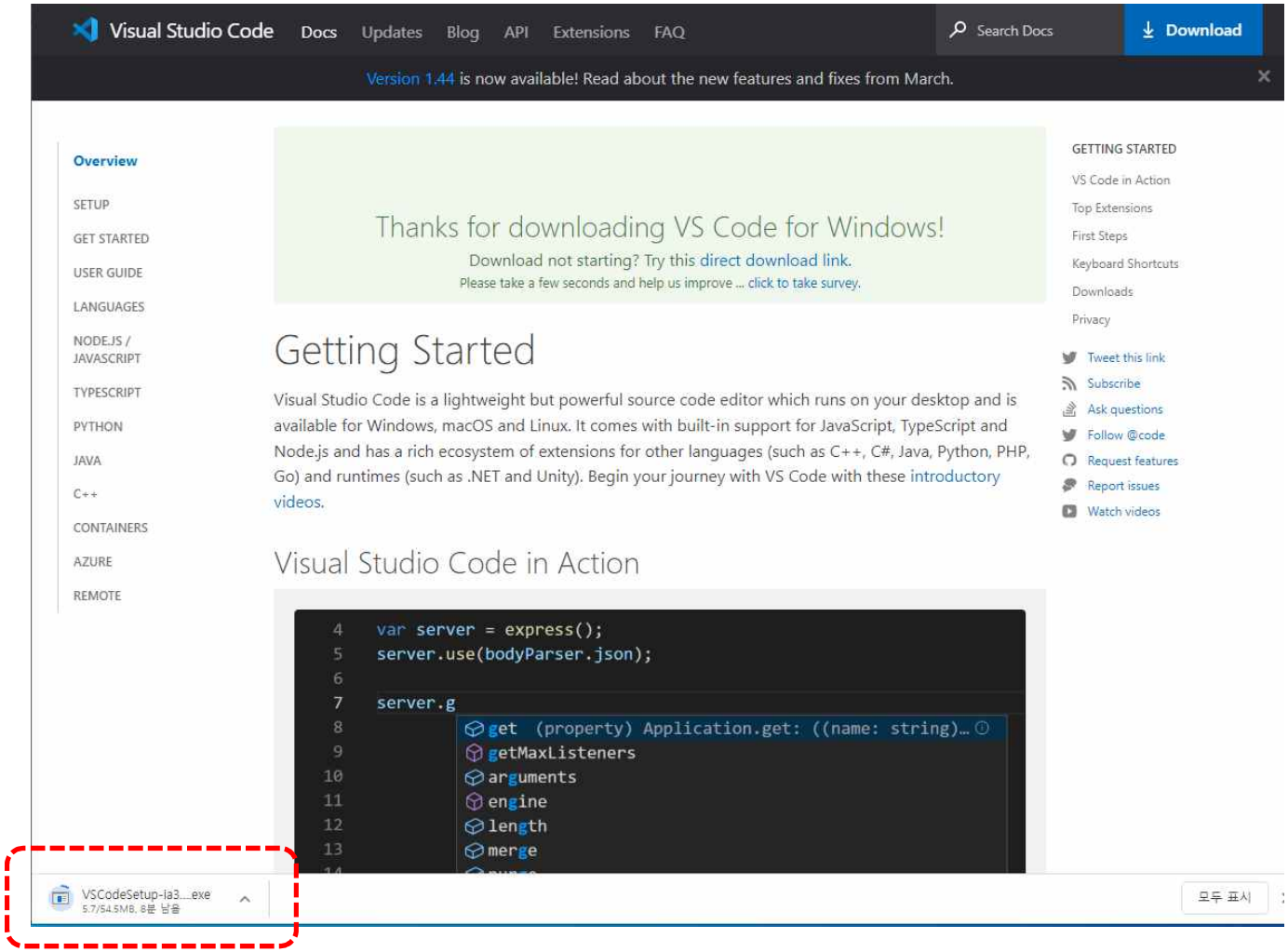
Visual Studio Code (VS Code)를 사용한 파이썬 프로그램 작성 및 디버깅

Python Program Debugging with Visual Studio Code (VS Code)

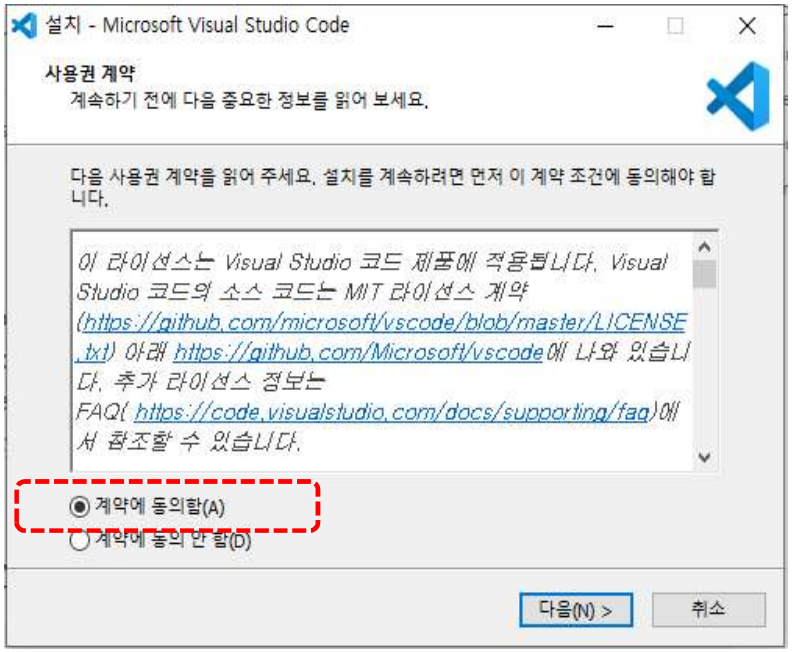
◆ https://code.visualstudio.com/download

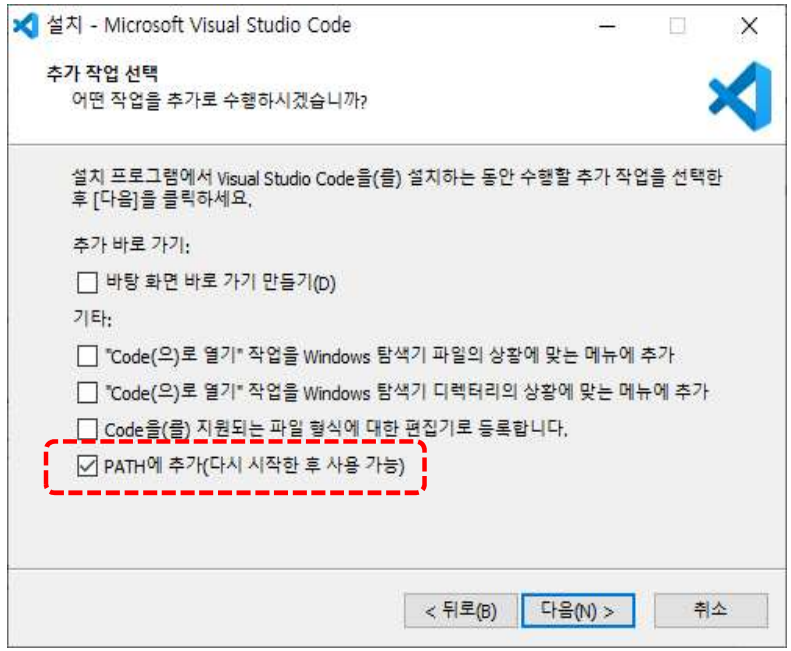


Download Visual Studio Code (System Installer 32-bit)

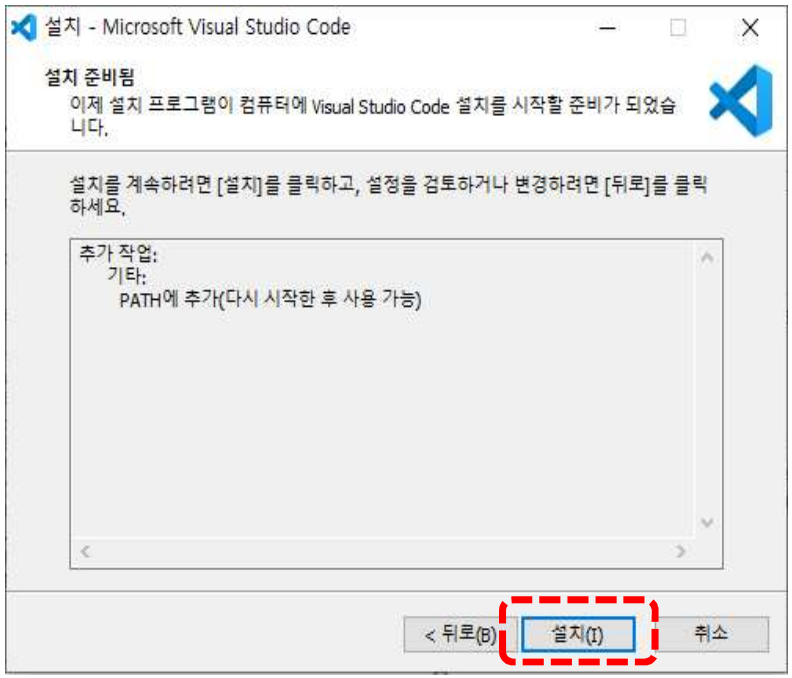


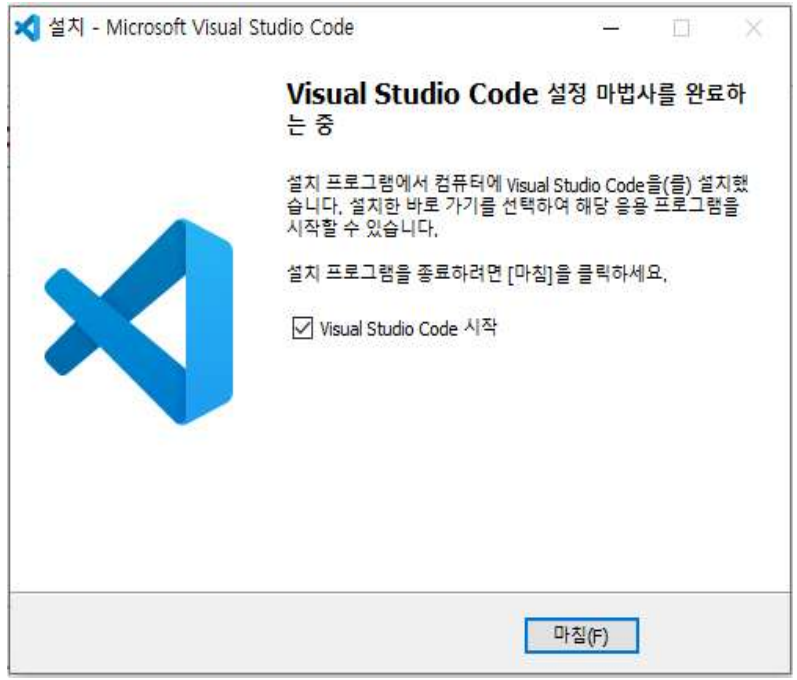
VS Code 설치


설치 - Microsoft Visual Studio Code
사용권 계약
계속하기 전에 다음 중요한 정보를 읽어 보세요.
다음 사용권 계약을 읽어 주세요. 설치를 계속하려면 먼저 이 계약 조건에 동의해야 합니다.
이 라이선스는 Visual Studio 코드 제품에 적용됩니다. Visual Studio 코드의 소스 코드는 MIT 라이선스 계약 (https://github.com/microsoft/vscode/blob/master/LICENSE.txt) 아래 https://github.com/Microsoft/vscode에 나와 있습니다. 추가 라이선스 정보는 FAQ(https://code.visualstudio.com/docs/supporting/faq)에서 참조할 수 있습니다.
계약에 동의함(A)
계약에 동의 안 함(D)
다음(N) >
취소

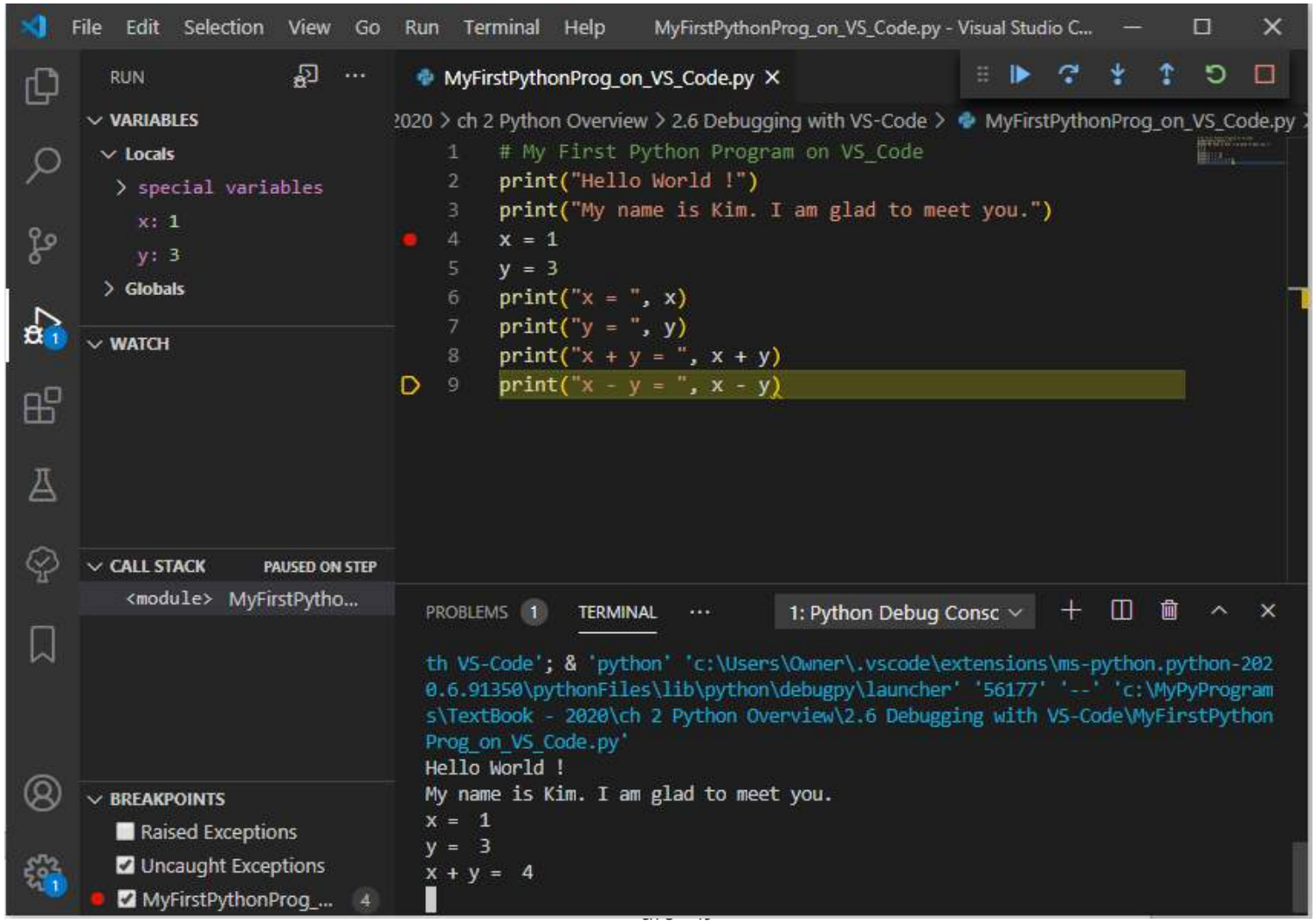

설치 - Microsoft Visual Studio Code
추가 작업 선택
어떤 작업을 추가로 수행하시겠습니까?
설치 프로그램에서 Visual Studio Code을(를) 설치하는 동안 수행할 추가 작업을 선택한 후 [다음]을 클릭하세요.
추가 바로 가기:
바탕 화면 바로 가기 만들기(D)
기타:
"Code(으)로 열기" 작업을 Windows 탐색기 파일의 상황에 맞는 메뉴에 추가
"Code(으)로 열기" 작업을 Windows 탐색기 디렉터리의 상황에 맞는 메뉴에 추가
Code을(를) 지원되는 파일 형식에 대한 편집기로 등록합니다.
PATH에 추가(다시 시작한 후 사용 가능)
< 뒤로(B)
다음(N) >
취소

VS Code 설치

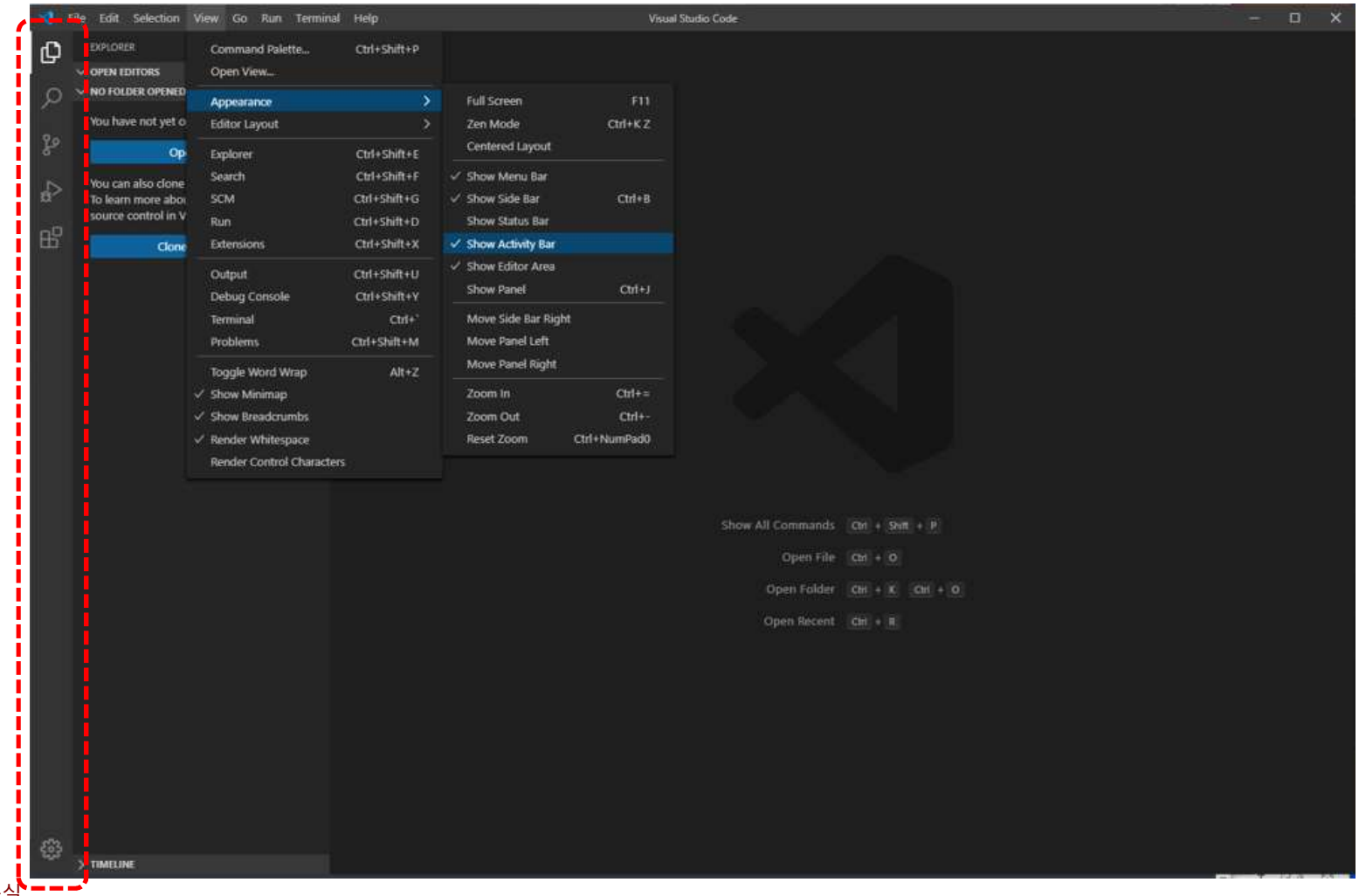

설치 - Microsoft Visual Studio Code
설치 준비됨
이제 설치 프로그램이 컴퓨터에 Visual Studio Code 설치를 시작할 준비가 되었습니다.
설치를 계속하려면 [설치]를 클릭하고, 설정을 검토하거나 변경하려면 [뒤로]를 클릭하세요.
추가 작업:
기타:
PATH에 추가(다시 시작한 후 사용 가능)
< 뒤로(B)
설치(I)
취소


설치 - Microsoft Visual Studio Code
Visual Studio Code 설정 마법사를 완료하는 중
설치 프로그램에서 컴퓨터에 Visual Studio Code을(를) 설치했습니다. 설치한 바로 가기를 선택하여 해당 응용 프로그램을 시작할 수 있습니다.
설치 프로그램을 종료하려면 [마침]을 클릭하세요.
Visual Studio Code 시작
마침(F)

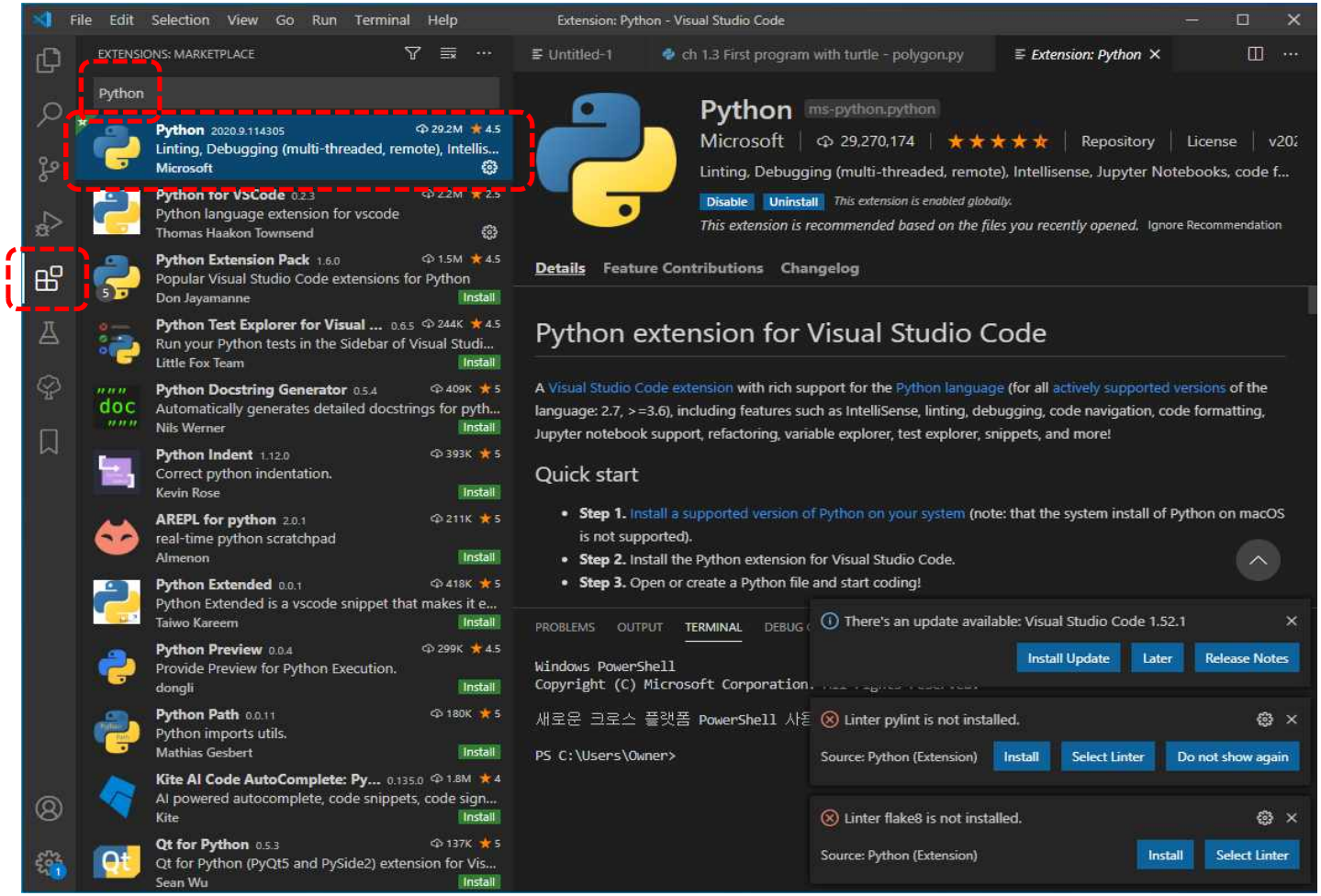
MyFirstPythonProg_on_VS_Code



View -> appearance -> show activity bar

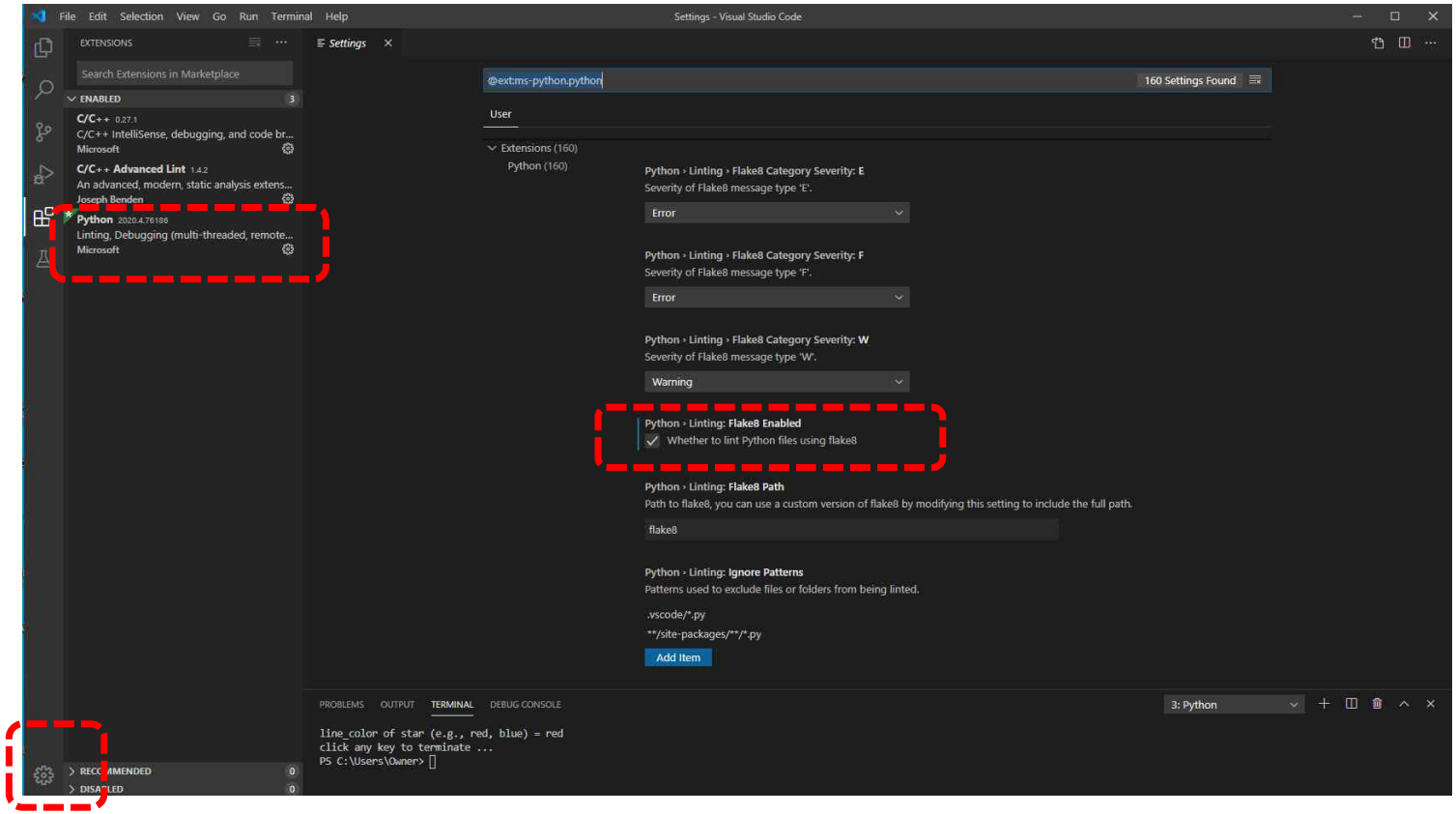


Extensions (control – shift – x)
=> Python 추가



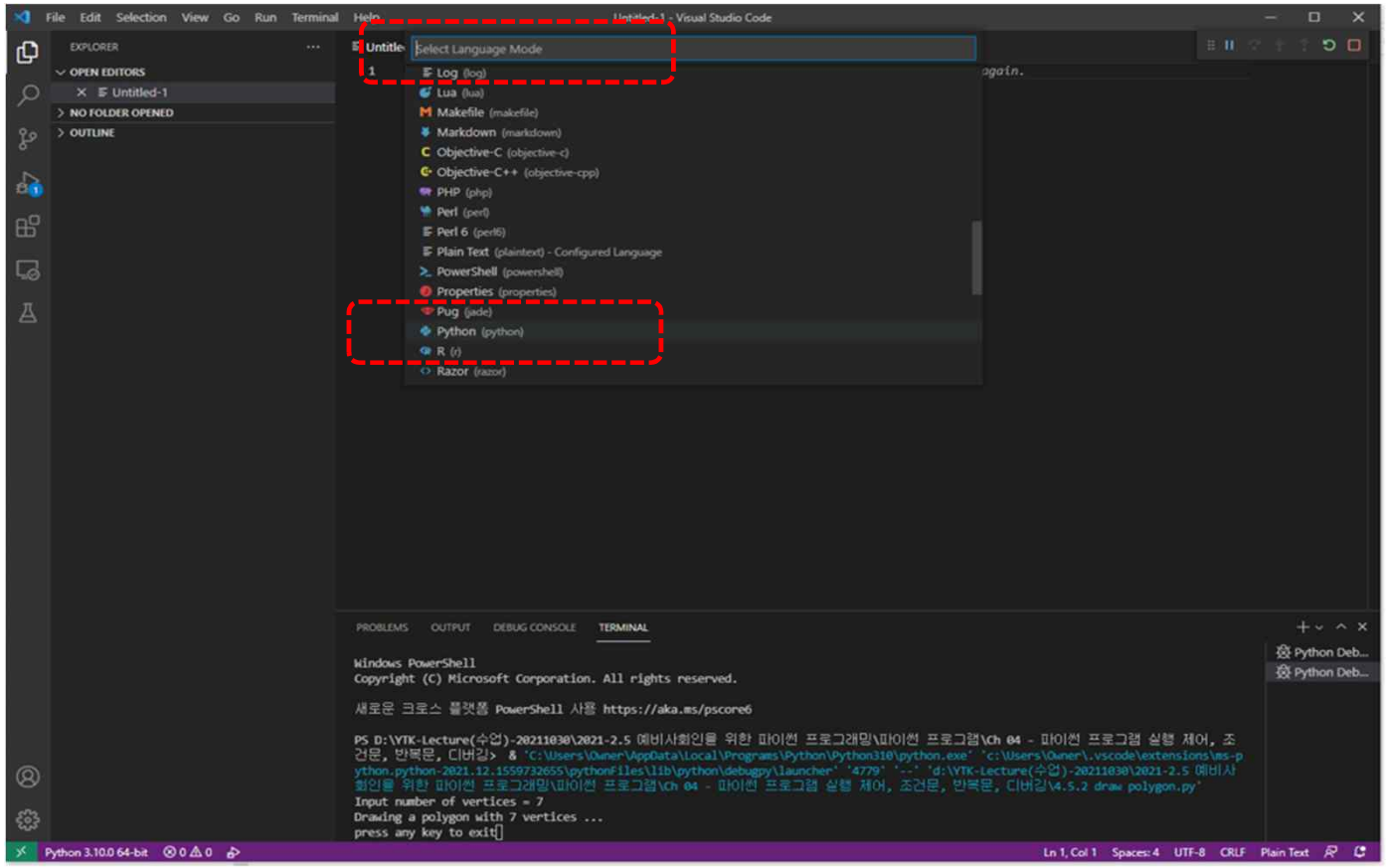
Visual Studio Code에 Turtle Graphic관련 모듈 설치

◆ **Manage -> Python -> Extension Settings -> Linting : Flake8 Enabled**



Visual Studio에서 Python 프로그램 작성

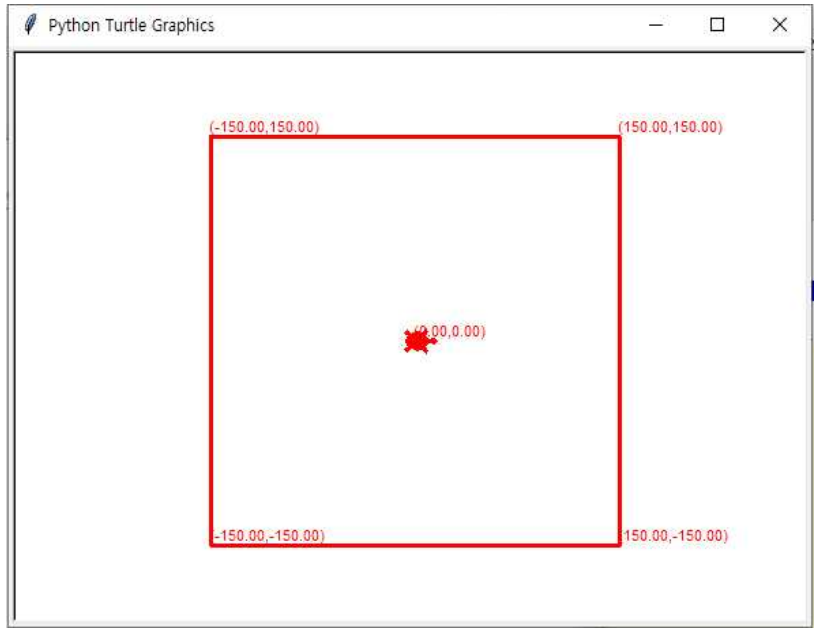
◆ Select Language Mode -> Python



터틀 프로그래밍 예제 – draw_square.py

```
# draw_square with Turtle graphic and loop
import turtle

t = turtle.Turtle()
t.shape("turtle")
t.color("red")
t.width(3)
width = int(input("input width of square = "))
turn_angle = 90
loop_count = 0
t.home()
t.write(t.position())
start_x = - width//2
start_y = - width//2
t.up()
t.goto((start_x, start_y))
t.down()
while loop_count < 4:
    t.write(t.position())
    t.forward(width)
    t.left(turn_angle)
    loop_count = loop_count + 1
t.up(); t.home(); t.down()
input("press any key to exit")
```

```
input width of square = 300
press any key to exit
```



Run without Debugging (Ctrl + F5)

File Edit Selection View Go Run Terminal Help — 1.3 draw_square.py - Visual Studio Code

RUN

Run and Debug

To customize Run and Debug, open a folder and create a launch.json file.

Show all automatic debug configurations.

Untitled-1 | Extension: Python | 1.3 draw_square.py

D: > YTK-Lecture(수업)-20210101 > 2020-2 파이썬 프로그래밍 (예비사회인 IT 및 SW 교육프로그램) > PythonProg > ch 1 > 1.3 draw_square.py

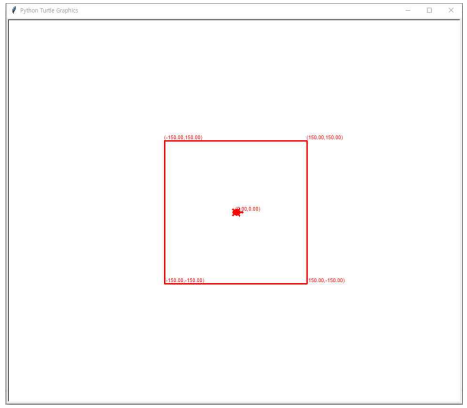
```python
# draw_square with Turtle graphic and loop
import turtle

t = turtle.Turtle()
t.shape("turtle")
t.color("red")
t.width(3)
width = int(input("input width of square = "))
turn_angle = 90
loop_count = 0
t.home()
t.write(t.position())
start_x = - width//2
start_y = - width//2
t.up()
t.goto((start_x, start_y))
t.down()
while loop_count < 4:
    t.write(t.position())
    t.forward(width)
    t.left(turn_angle)
    loop_count = loop_count + 1
t.up(); t.home(); t.down()
```

BREAKPOINTS

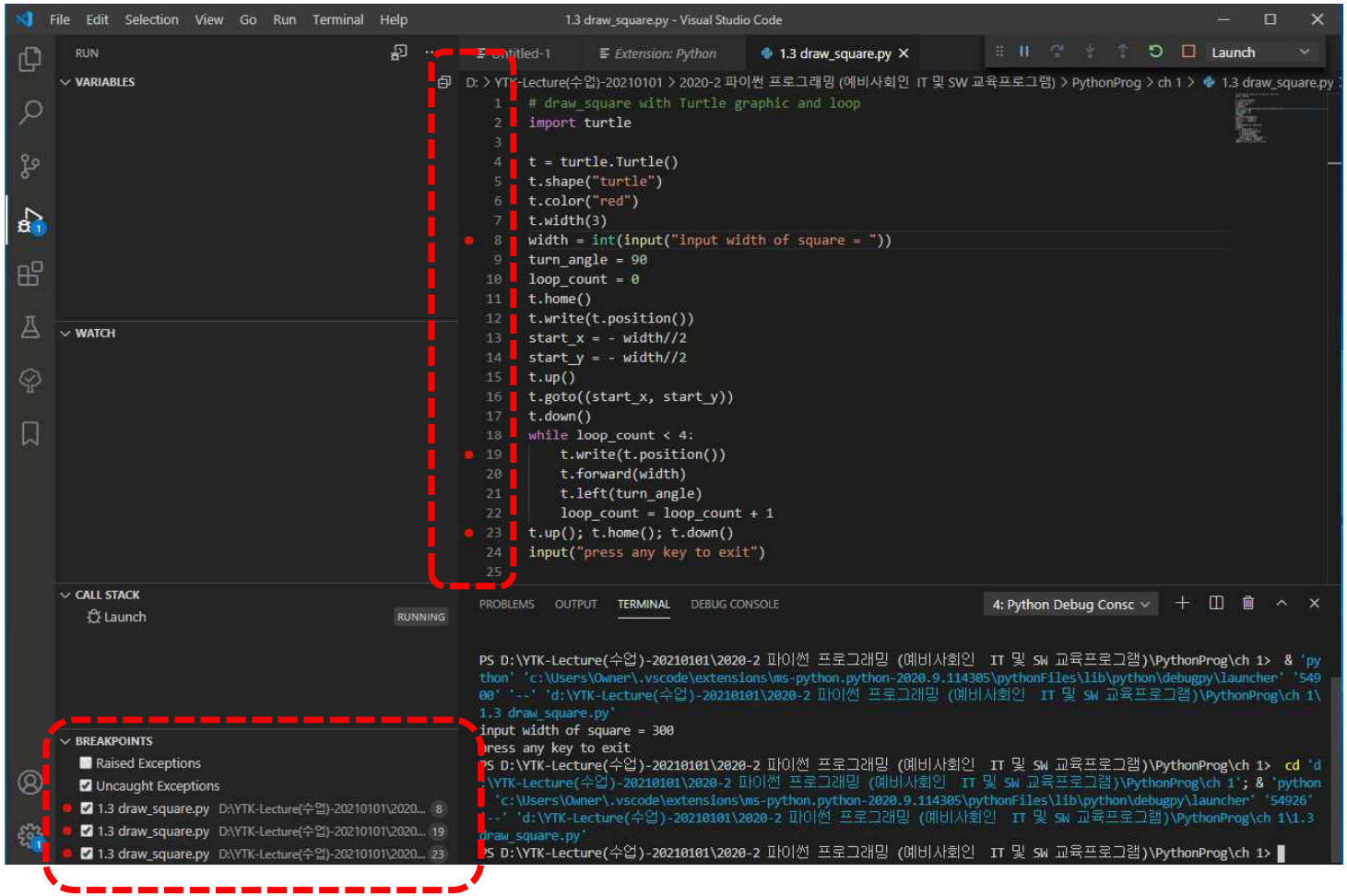
- Raised Exceptions
- Uncaught Exceptions
- 1.3 draw_square.py D:\YTK-Lecture(수업)-20210101\2020... 8
- 1.3 draw_square.py D:\YTK-Lecture(수업)-20210101\2020... 19
- 1.3 draw_square.py D:\YTK-Lecture(수업)-20210101\2020... 23

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE — 2: Python Debug Consc

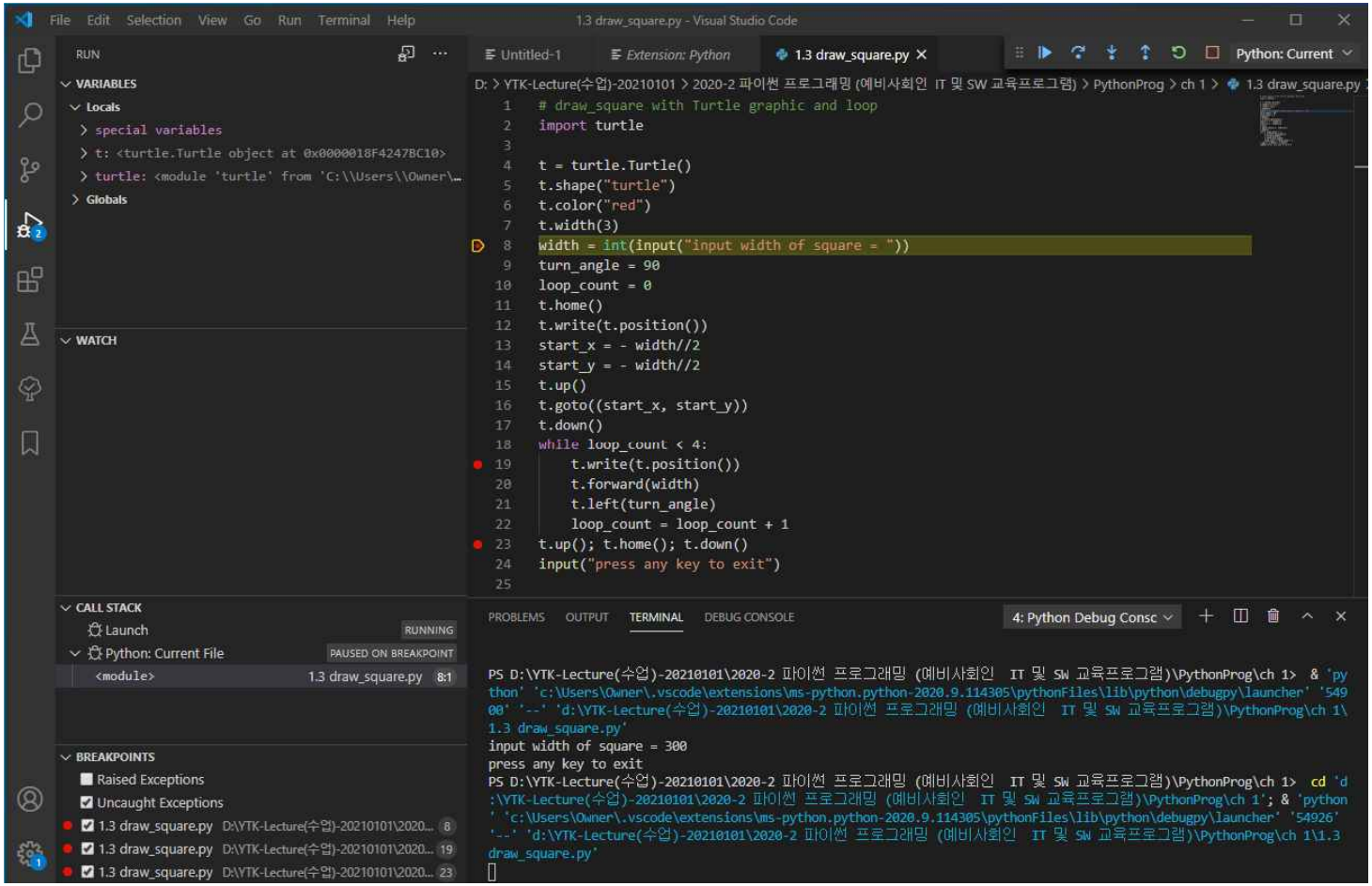
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/pscore6

PS D:\YTK-Lecture(수업)-20210101\2020-2 파이썬 프로그래밍 (예비사회인  IT 및 SW 교육프로그램)\PythonProg\ch 1>  & 'py
thon' 'c:\Users\Owner\.vscode\extensions\ms-python.python-2020.9.114305\pythonFiles\lib\python\debugpy\launcher' '548
54' '--' 'd:\YTK-Lecture(수업)-20210101\2020-2 파이썬 프로그래밍 (예비사회인  IT 및 SW 교육프로그램)\PythonProg\ch 1\
1.3 draw_square.py'
input width of square = 300
PS D:\YTK-Lecture(수업)-20210101\2020-2 파이썬 프로그래밍 (예비사회인  IT 및 SW 교육프로그램)\PythonProg\ch 1>
```



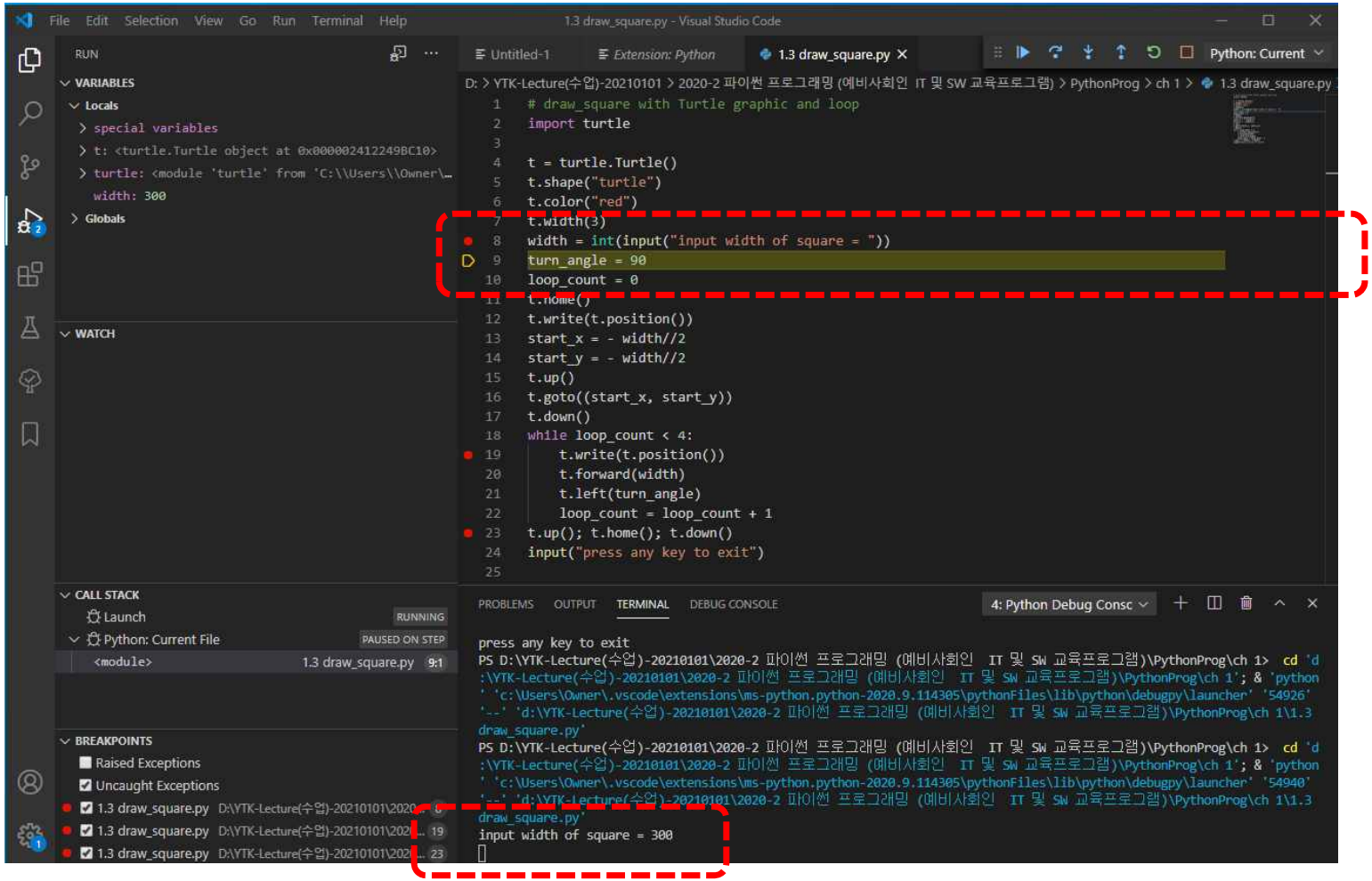
Break Points 설정



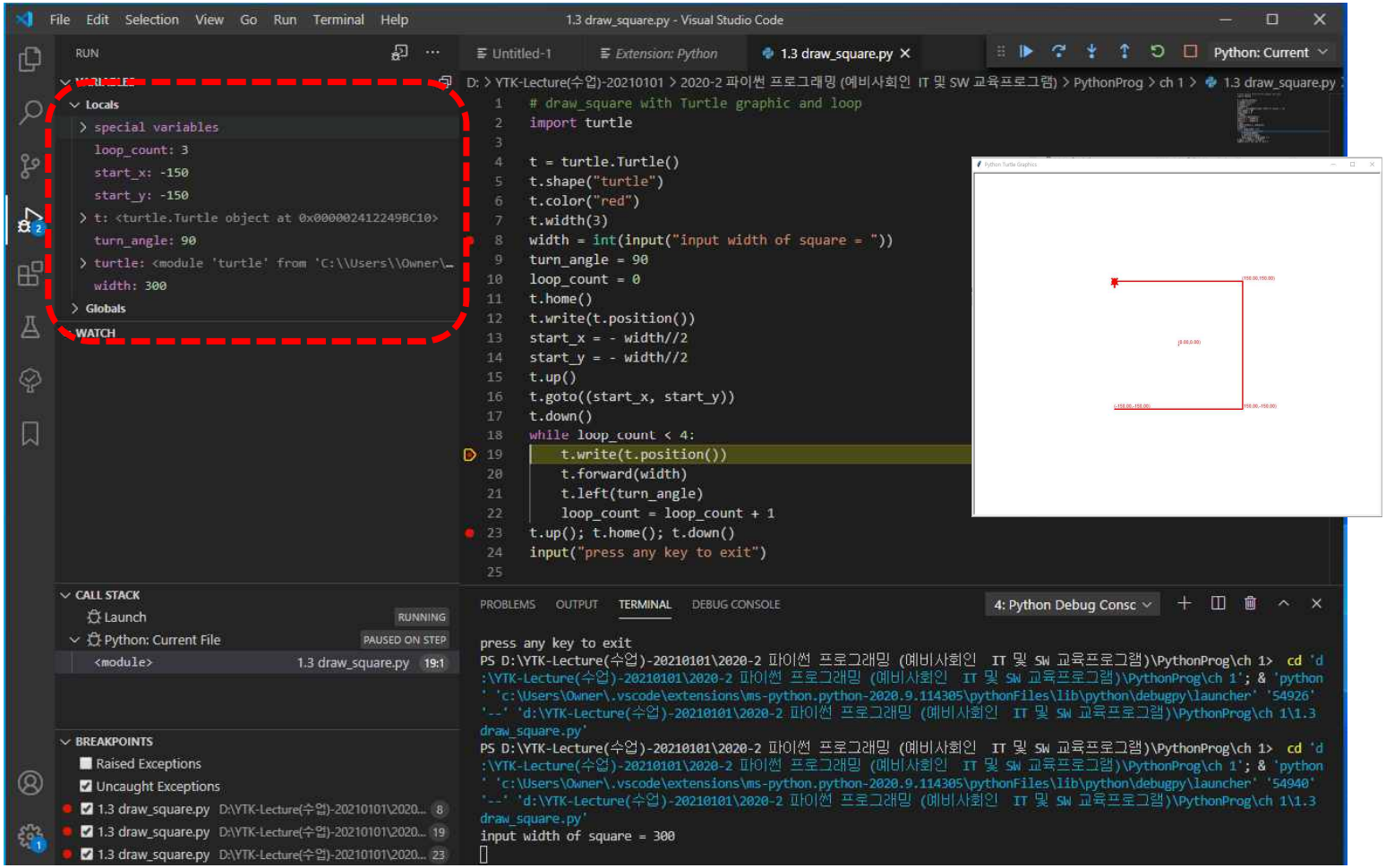
Run -> Start Debugging (F5) -> Python File



Step-over (F10)을 사용한 프로그램 실행



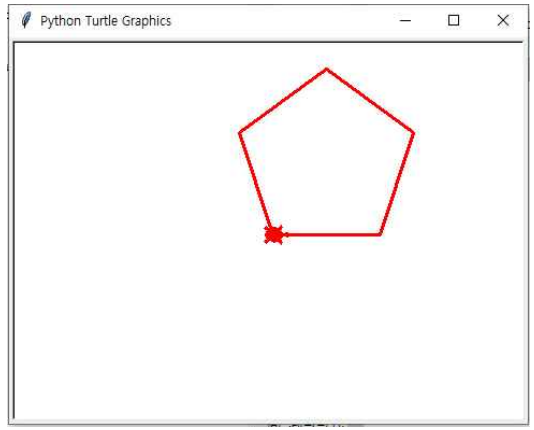
Step-over (F10)을 단계별 실행 결과 확인



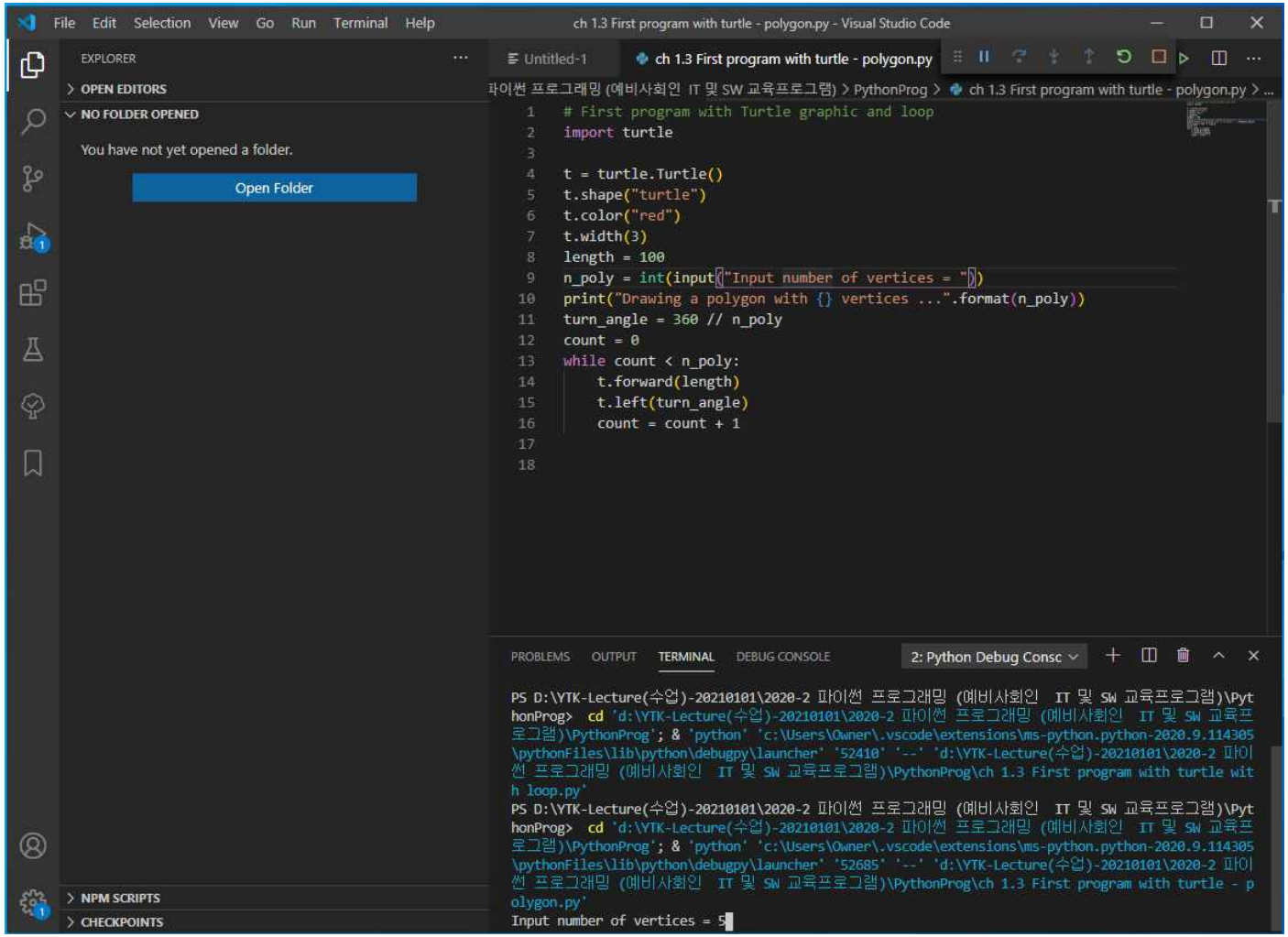
Sample Python Source – turtle_polygon.py

◆ 다각형 그리기

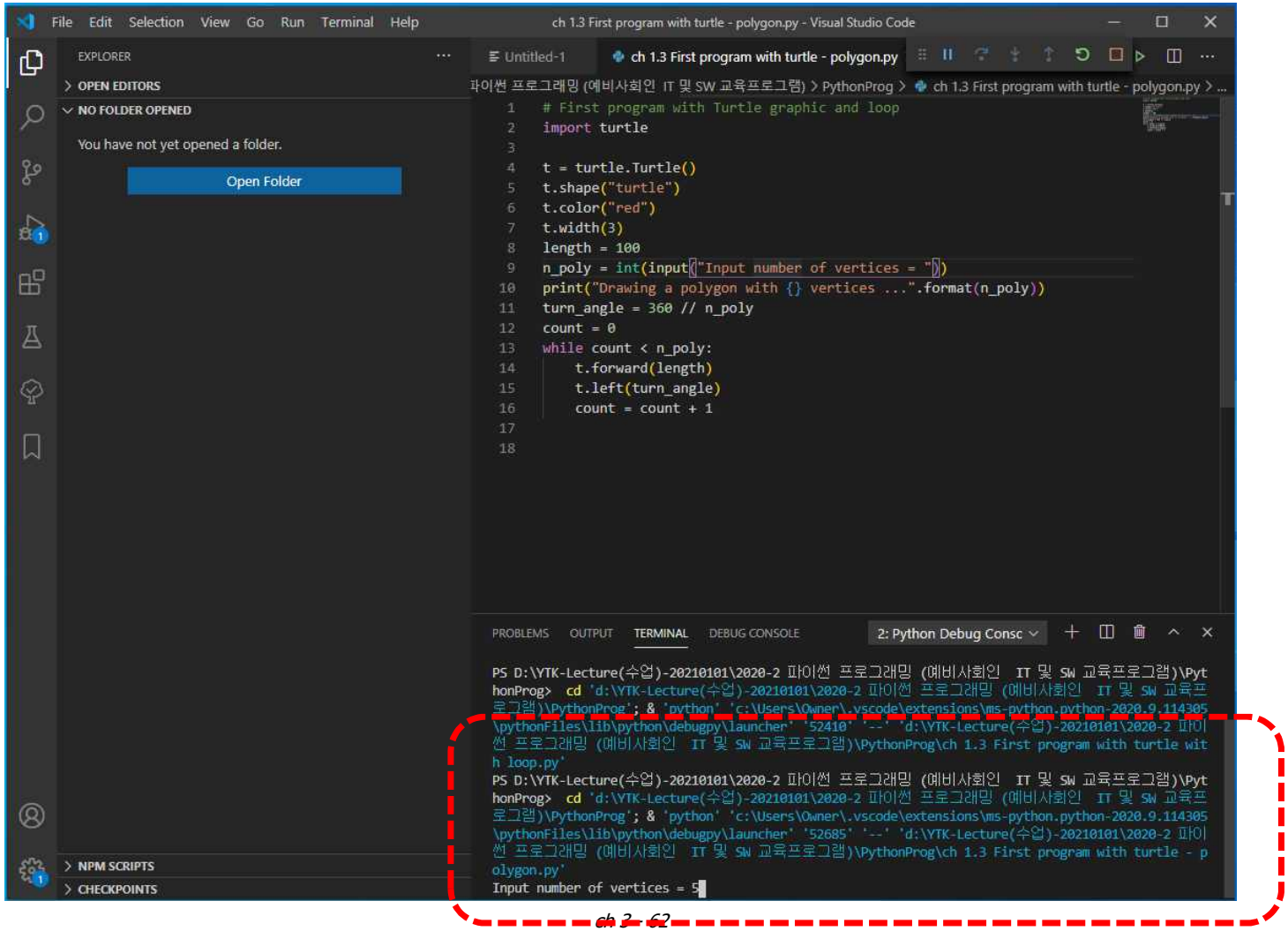
```
# Drawing Polygon with Turtle graphic and loop
import turtle

t = turtle.Turtle()
t.shape("turtle")
t.color("red")
t.width(3)
length = 100
n_poly = int(input("Input number of vertices = "))
print("Drawing a polygon with {} vertices ...".format(n_poly))
turn_angle = 360 // n_poly
loop_count = 0
while loop_count < n_poly:
    t.forward(length)
    t.left(turn_angle)
    loop_count = loop_count + 1
```



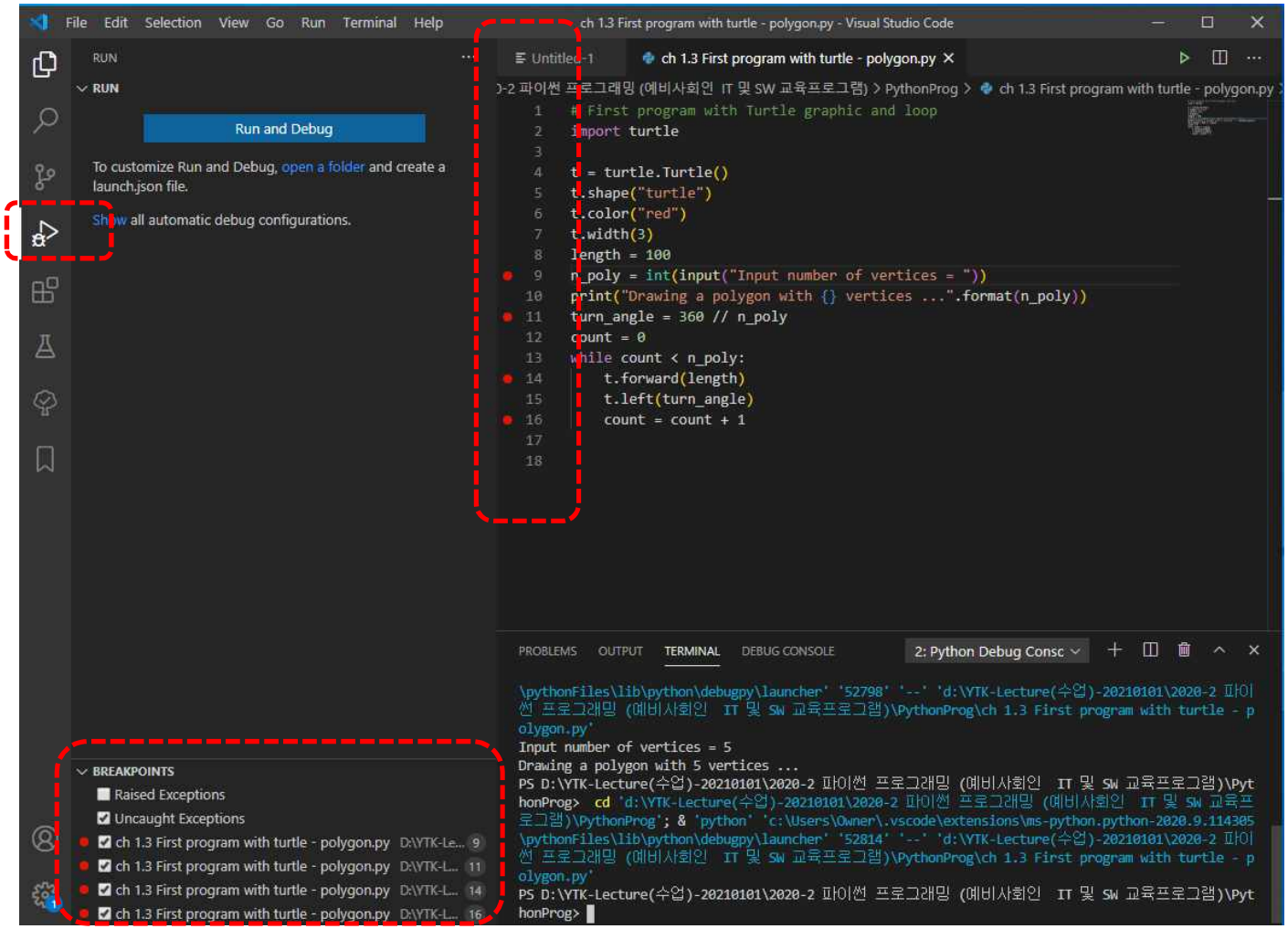
File Open on Visual Studio Code (File -> Open File (Ctrl+O))



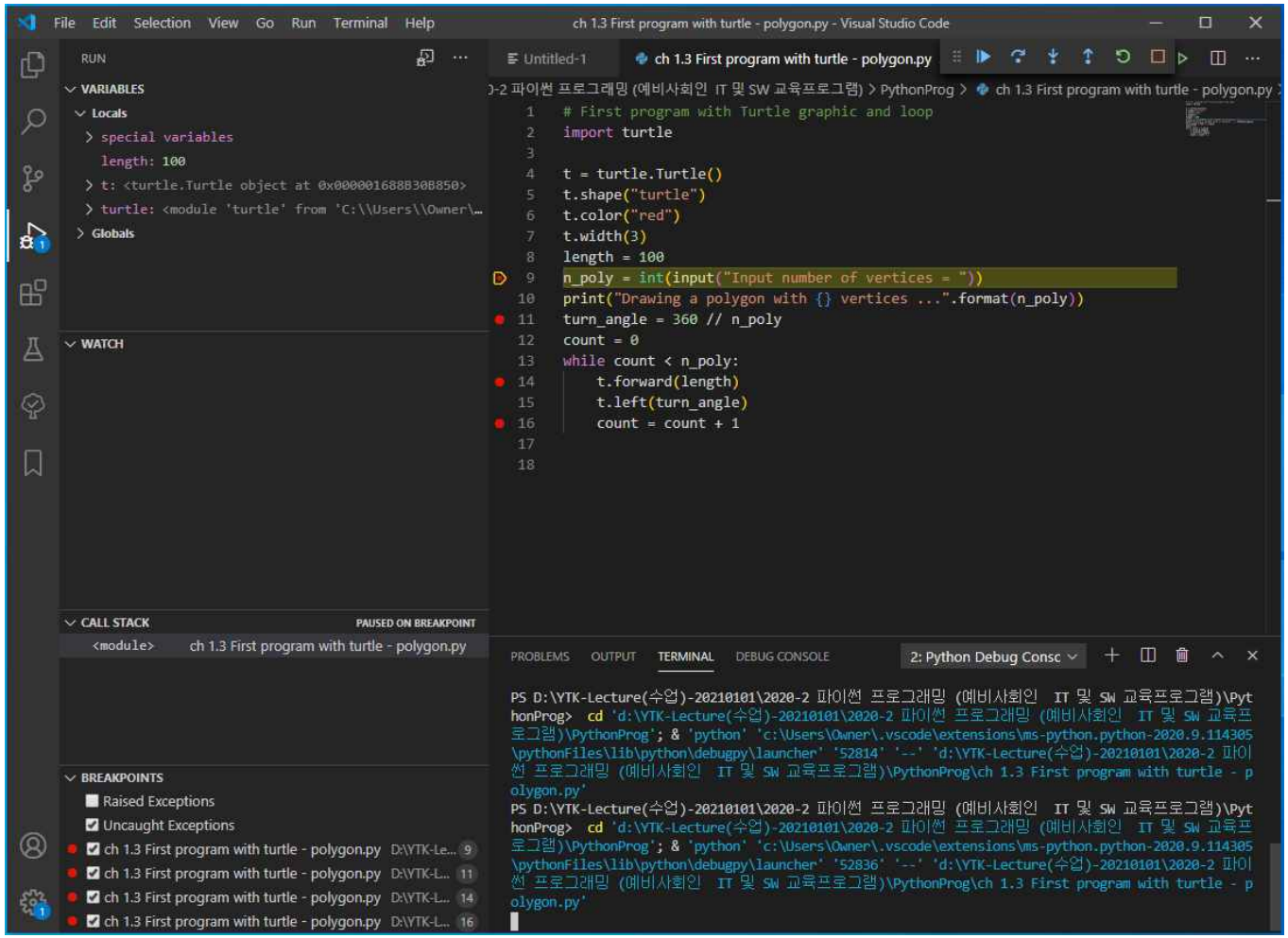
Run without Debugging on Visual Studio Code (Ctrl + F5)



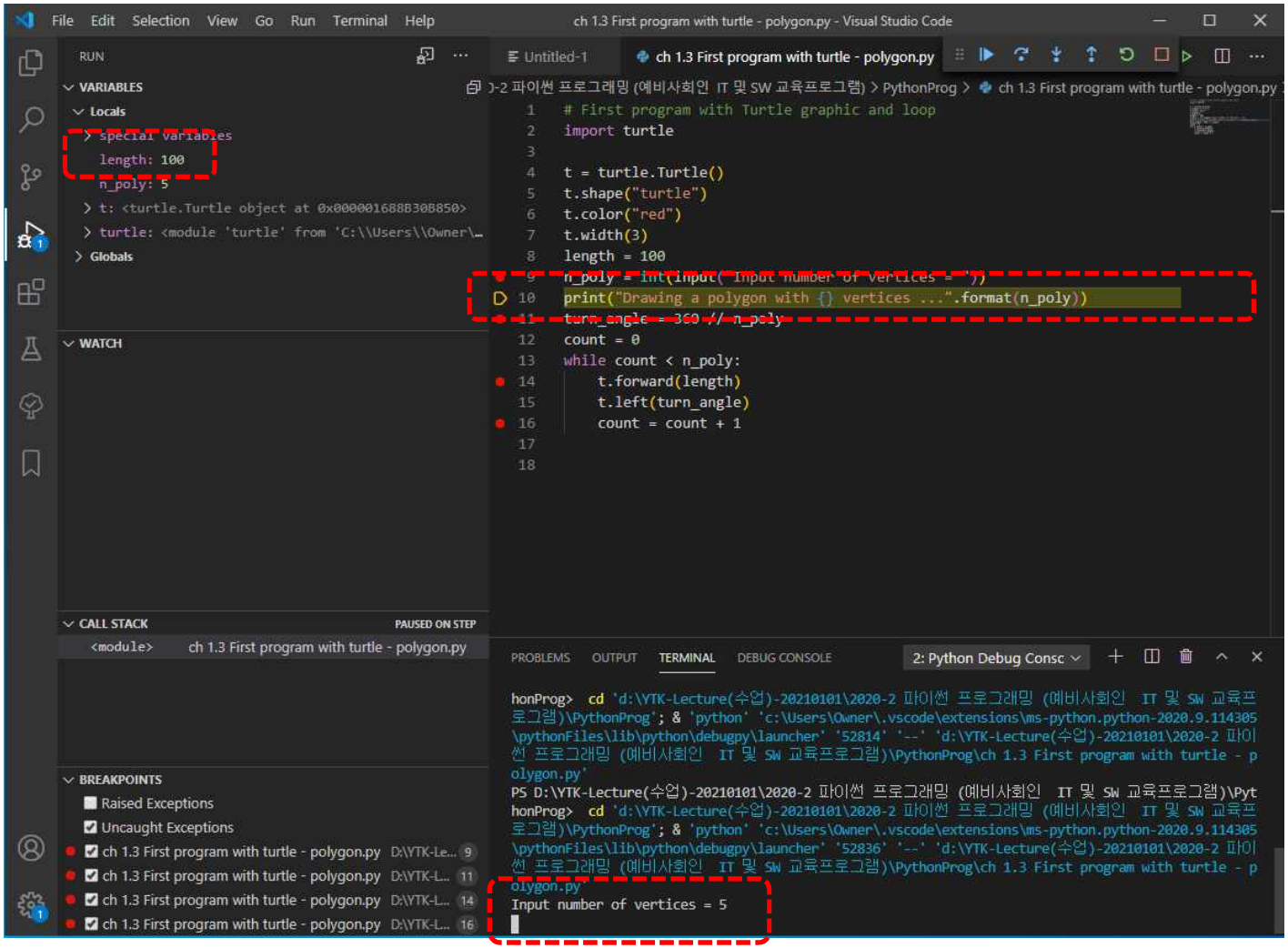
Setting Breakpoint on VS-Code



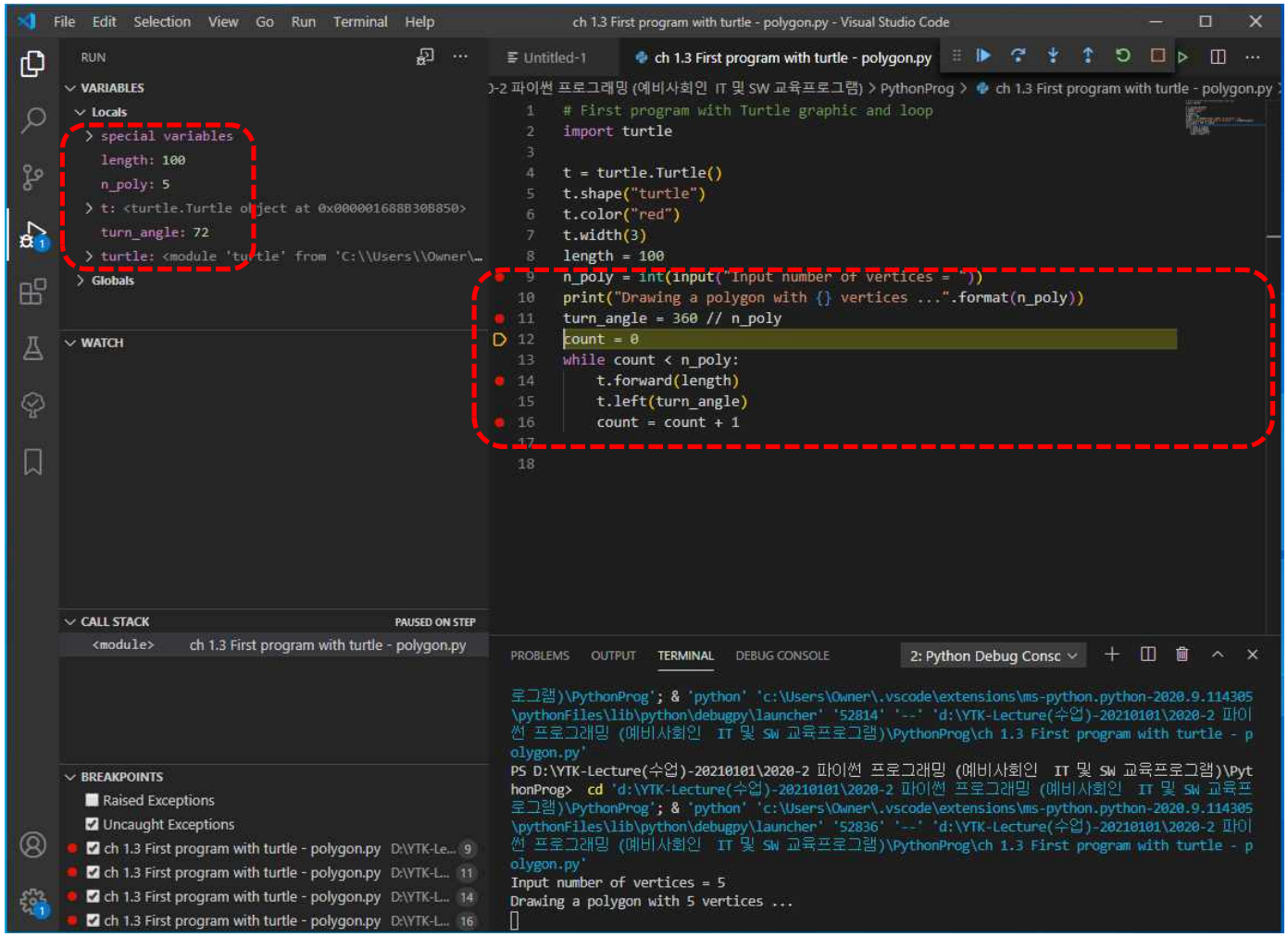
(Re-)Start Debugging (F5, Cntrl+Shift+F5) with Tracing (F10- Step over, F11-Step into)



Tracing with F10 (step-over)



Tracing with F10 (step-over)



Homework 3

Homework 3.1

3.1 본인의 생일을 연(year), 월(month), 일(day)의 3개 정수로 입력 받고, 이 날이 서기 1년 1월 1일부터 몇 번째 날짜인지를 계산하며, 이 날이 무슨 요일인지 계산하여 출력하며, 생일의 월별 탄생석 (birth stone)을 출력하는 파이썬 프로그램을 작성하라.
참고로 서기 1년 1월 1일은 월요일이다. 프로그램은 0 0 0이 입력될 때 까지 반복하도록 할 것.

(실행 예)

```
input your birth date (year month day) : 1 1 1
Your birth date = year(1), month(1), day(1) :
 elapsed 1 days from Jan01AD01, week day = Monday, birth_stone = Gernet
input your birth date (year month day) : 2023 1 1
Your birth date = year(2023), month(1), day(1) :
 elapsed 738521 days from Jan01AD01, week day = Sunday, birth_stone = Gernet
input your birth date (year month day) : 2023 2 6
Your birth date = year(2023), month(2), day(6) :
 elapsed 738557 days from Jan01AD01, week day = Monday, birth_stone = Amethyst
input your birth date (year month day) : 0 0 0
```

Homework 3.2

3.2 날짜를 나타내는 연(year), 월(month), 일(day)의 3개 정수를 입력 받고, 이 달의 달력을 출력하는 파이썬 프로그램을 작성하라. 달력의 첫 줄은 요일을 의미하는 영문 약자 (SUN, MON, TUE, WED, THR, FRI, SAT)를 출력하고, 이 달의 1일 부터 요일에 맞추어 출력되도록 할 것.
(실행 예)

```
input year month day : 2023 2 6
Input yr_mn_dy_strings :  ['2023', '2', '6']

February of Year 2023
============================
 SUN MON TUE WED THR FRI SAT
----------------------------
                  1   2   3   4
   5   6   7   8   9  10  11
  12  13  14  15  16  17  18
  19  20  21  22  23  24  25
  26  27  28
============================
```

```
input year month day : 2023 3 1
Input yr_mn_dy_strings :  ['2023', '3', '1']

March of Year 2023
============================
 SUN MON TUE WED THR FRI SAT
----------------------------
                  1   2   3   4
   5   6   7   8   9  10  11
  12  13  14  15  16  17  18
  19  20  21  22  23  24  25
  26  27  28  29  30  31
============================
```

Homework 3.3

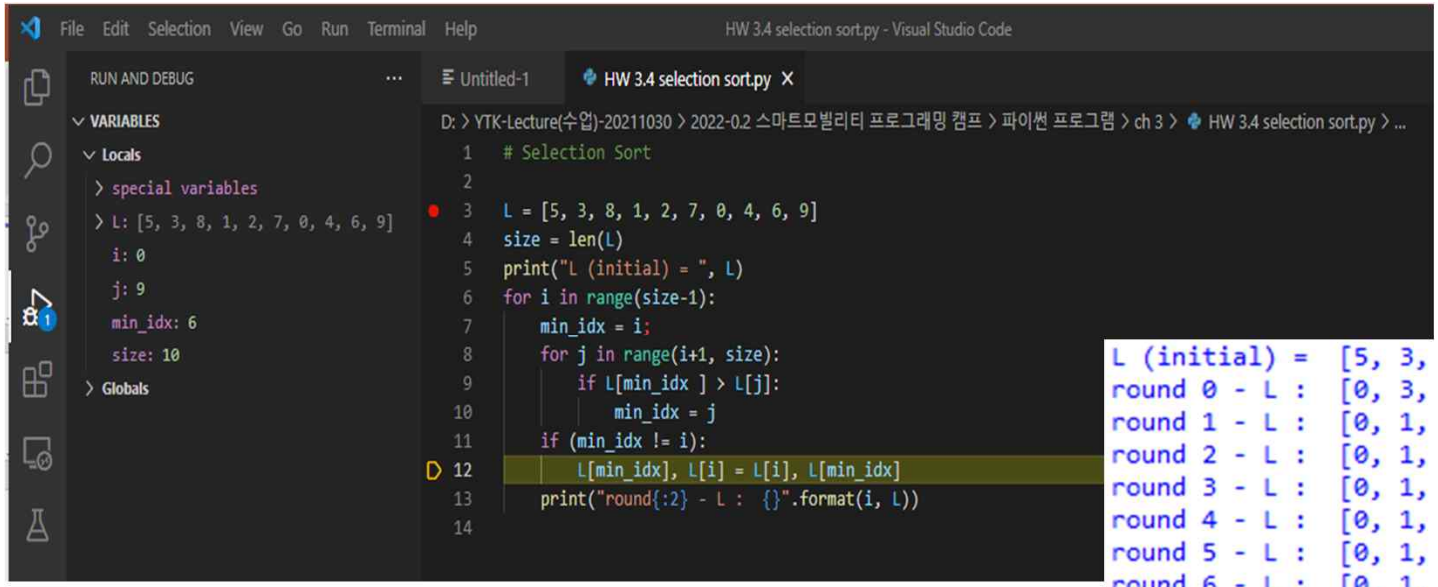
3.3 시간을 나타내는 시 (hour), 분(min), 초(sec)의 3개 정수를 한 줄로 입력 받고, 이 시간이 그날의 0시 0분 0초로 부터 몇 초가 경과되었는지 계산하고, 그날의 자정 (24:00:00)까지 몇 초가 남았는지 계산하여 출력하는 파이썬 프로그램을 작성하라. 시간의 출력 양식은 (00:00:00) ~ (23:59:59)로 표시할 것.
(실행 예)

```
input hour min sec : 0 0 0
Input time : (00:00:00)
Elapsed seconds from last midnight =  0
Remaining seconds to next-midnight =  86400
input hour min sec : 0 0 1
Input time : (00:00:01)
Elapsed seconds from last midnight =  1
Remaining seconds to next-midnight =  86399
input hour min sec : 1 0 0
Input time : (01:00:00)
Elapsed seconds from last midnight =  3600
Remaining seconds to next-midnight =  82800
input hour min sec : 23 59 59
Input time : (23:59:59)
Elapsed seconds from last midnight =  86399
Remaining seconds to next-midnight =  1
input hour min sec : 23 0 0
Input time : (23:00:00)
Elapsed seconds from last midnight =  82800
Remaining seconds to next-midnight =  3600
input hour min sec :
```

Homework 3.4

3.4 선택 정렬 (Selection Sort) 구현 및 VS code로 디버깅

- 2중 for-loop 구조로 주어진 리스트 L에 포함된 10개의 정수 원소들을 정렬하는 파이썬 프로그램을 작성하라.
- 이 선택정렬 프로그램의 실행 과정을 VS code로 확인하라.
- 선택 정렬의 각 단계에서 리스트 L에 포함된 원소들이 어떻게 정렬되어 가는지 확인할 수 있도록 중간 상태를 출력하도록 하고, VS Code의 local variables에서 확인 할 것.
- (실행 예)




```
L (initial) =  [5, 3, 8, 1, 2, 7, 0, 4, 6, 9]
round 0 - L :  [0, 3, 8, 1, 2, 7, 5, 4, 6, 9]
round 1 - L :  [0, 1, 8, 3, 2, 7, 5, 4, 6, 9]
round 2 - L :  [0, 1, 2, 3, 8, 7, 5, 4, 6, 9]
round 3 - L :  [0, 1, 2, 3, 8, 7, 5, 4, 6, 9]
round 4 - L :  [0, 1, 2, 3, 4, 7, 5, 8, 6, 9]
round 5 - L :  [0, 1, 2, 3, 4, 5, 7, 8, 6, 9]
round 6 - L :  [0, 1, 2, 3, 4, 5, 6, 8, 7, 9]
round 7 - L :  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
round 8 - L :  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Homework 3.5

3.5 터틀 그래픽 기반 별 그리기 (drawing a star) 프로그램 작성 및 VS Code를 사용한 디버깅

- 터틀 그래픽 기반 별 (star) 그리기에서 선의 길이 (length)와 도형 중심 좌표 (x0, y0)를 입력 받고, 도형 중심 좌표 (x0, y0)로 이동한 후, 지정된 별을 그리는 파이썬 프로그램을 작성하라.
- 이 별 그리기 프로그램의 실행 과정을 VS code로 확인하라.
- 별 그리기 각 단계에서 어떤 동작이 차례대로 실행되는지 확인할 수 있도록 VS Code의 local variables에서 확인 하고, 실제 화면에 그려지는 내용을 확인 할 것.
- (실행 예)