

02

CHAPTER

미리 만드는 쓸 만한 프로그램

학습목표

- 변수를 이해한다.
- `print()` 함수의 사용법을 익힌다.
- 긴 프로그램을 작성하고 저장하는 방법을 익힌다.
- 키보드로 숫자를 입력해 계산하는 계산기 프로그램을 만든다.
- 터틀 그래픽을 이용해 간단히 그림을 그리는 프로그램을 만든다.

SECTION 01 이장에서 만들 프로그램

SECTION 02 계산기 프로그램의 기본 기능 구현

SECTION 03 계산기 프로그램 저장

SECTION 04 계산기 프로그램 확장

SECTION 05 터틀 그래픽 프로그램 작성

요약

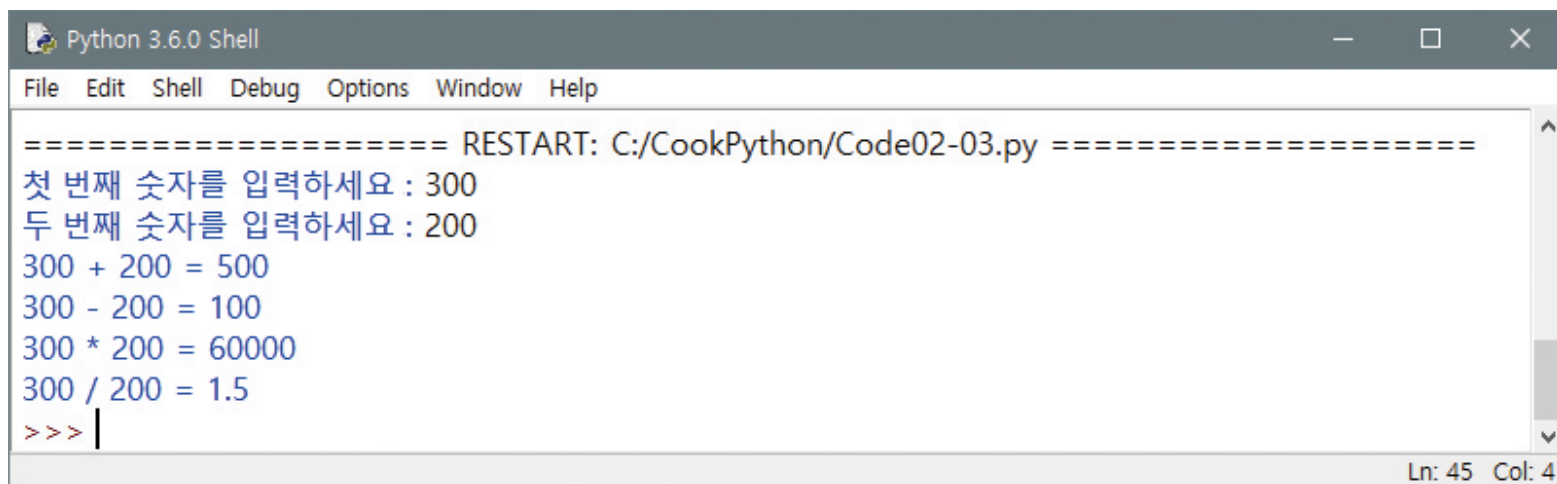
연습문제



Section01 이 장에서 만들 프로그램

■ [프로그램1] 간단 계산기

- 숫자를 2개 입력해 더하기, 빼기, 곱하기, 나누기 등을 계산하는 아주 기본적인 기능

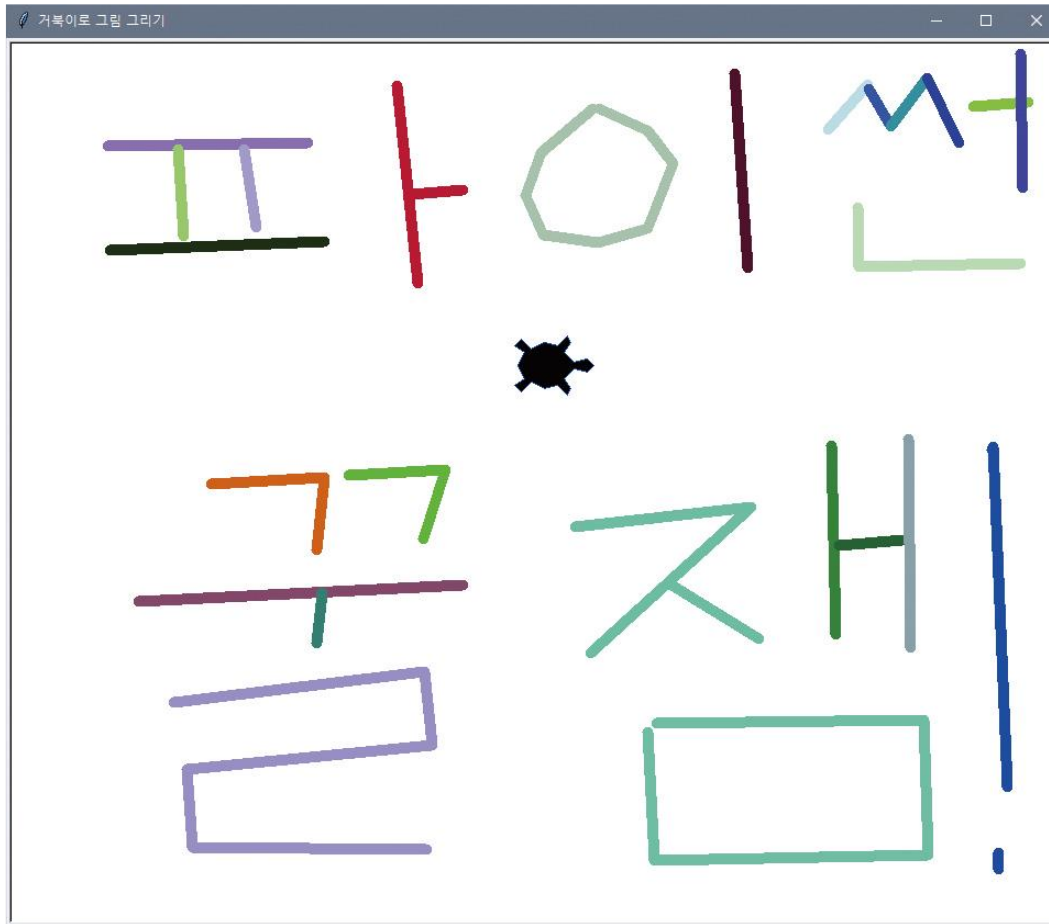


```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:/CookPython/Code02-03.py =====
첫 번째 숫자를 입력하세요 : 300
두 번째 숫자를 입력하세요 : 200
300 + 200 = 500
300 - 200 = 100
300 * 200 = 60000
300 / 200 = 1.5
>>> |
Ln: 45 Col: 4
```

Section01 이 장에서 만들 프로그램

■ [프로그램2] 터틀 그래픽

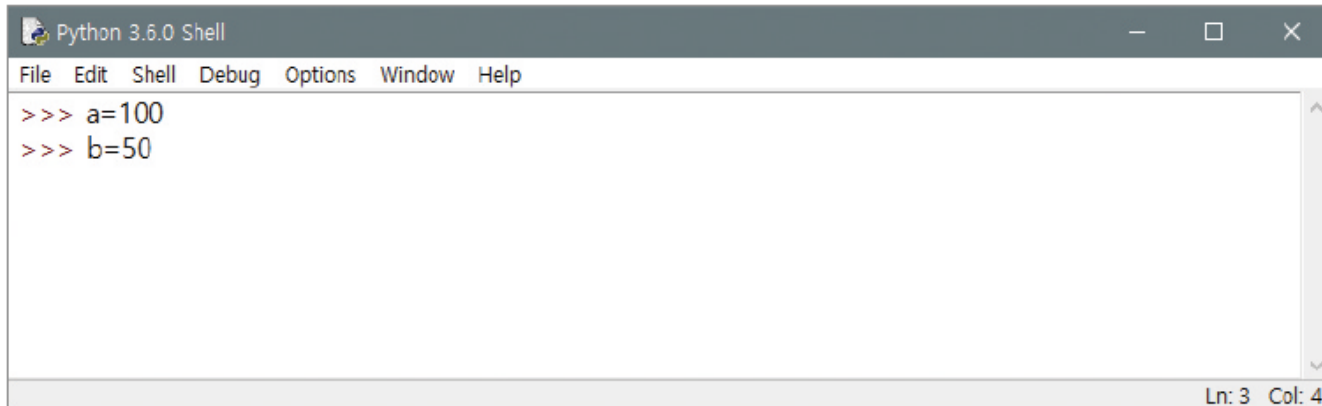
- 마우스로 거북이 커서를 움직여 재미있는 그림을 그리는 프로그램



Section02 계산기 프로그램의 기본 기능 구현

■ 필요한 변수 준비

- = 기호는 같다는 의미가 아니라 '오른쪽의 것을 왼쪽으로 넣어라'는 의미의 대입 연산자
 - $a=100$ 은 $a \leftarrow 100$ 과 같은 개념
 - 내부적으로는 a 와 b 그릇이 생겨 a 그릇에는 100이, b 그릇에는 50이 담긴 상태
- 프로그래밍 언어에서 그릇과 같은 역할을 하는 것이 바로 변수



```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
>>> a=100
>>> b=50
Ln: 3 Col: 4
```

그림 2-1 변수 준비



그릇 이름 : a



그릇 이름 : b

그림 2-2 그릇을 2개 준비

Section02 계산기 프로그램의 기본 기능 구현

■ 더하기 기능 구현

- a 그릇의 100과 b 그릇의 50을 합쳐 새로운 result 그릇에 들어간 상태가 됨
- 변수는 result에 값이 들어가더라도 a, b의 변수값이 그대로 남음

```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
>>> a=100
>>> b=50
>>> result=a+b
```

Ln: 3 Col: 4

그림 2-3 더하기 구현

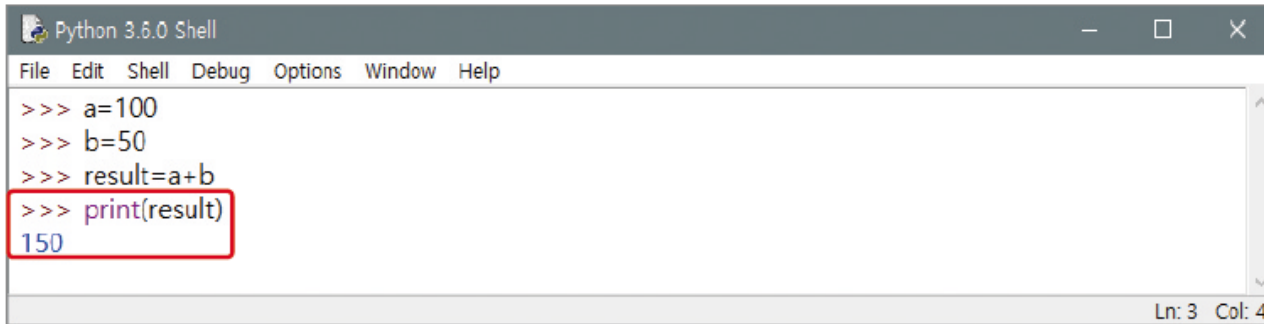


그림 2-4 더하는 작업

Section02 계산기 프로그램의 기본 기능 구현

■ 더한 결과 출력

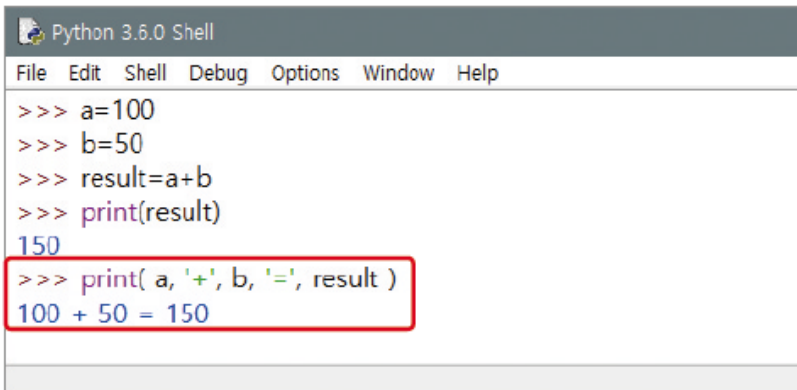
- result 그릇의 내용만 출력



```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
>>> a=100
>>> b=50
>>> result=a+b
>>> print(result)
150
Ln: 3 Col: 4
```

그림 2-5 더한 결과 출력 1

- result 그릇의 내용과 계산식도 출력



```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
>>> a=100
>>> b=50
>>> result=a+b
>>> print(result)
150
>>> print( a, '+', b, '=', result )
100 + 50 = 150
```

그림 2-6 더한 결과 출력 2

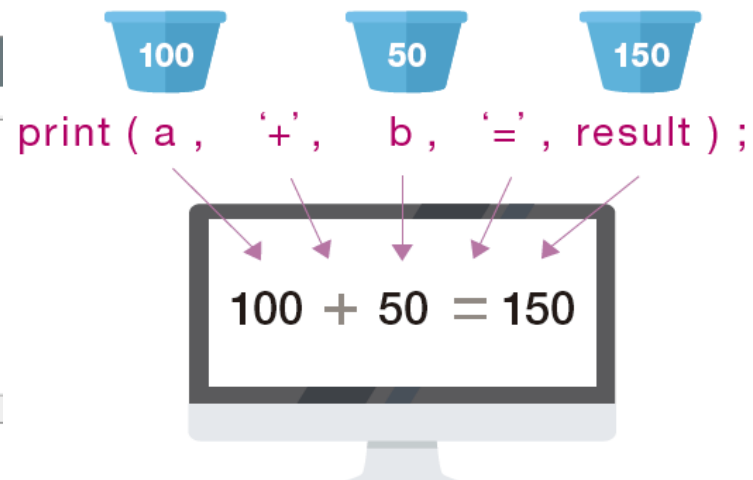
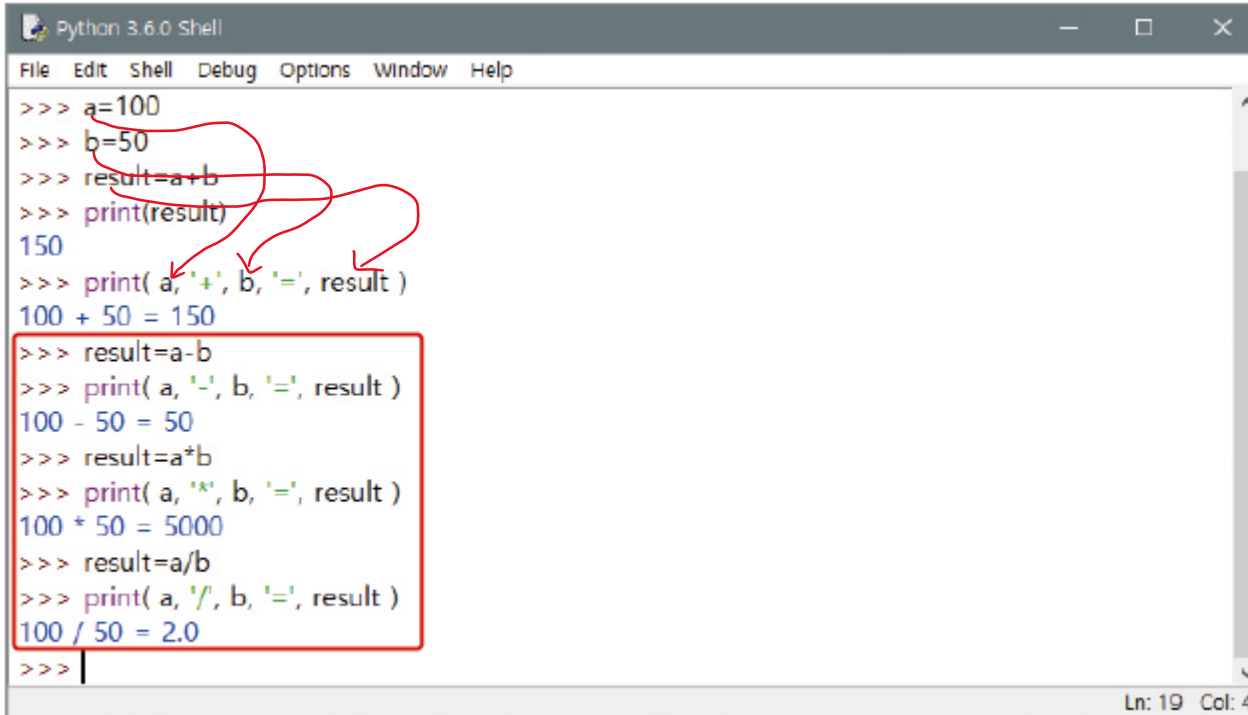


그림 2-7 print() 함수로 모두 출력

Section03 계산기 프로그램 저장

■ 빼기, 곱하기, 나누기 기능 구현



A screenshot of a Python 3.6.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main area shows a series of Python commands and their outputs. Red annotations are present: a red box highlights the subtraction, multiplication, and division code blocks; red arrows point from the first two lines of code to the output '150'; and red circles highlight the '+' and '-' operators in the print statements.

```
>>> a=100
>>> b=50
>>> result=a+b
>>> print(result)
150
>>> print( a, '+', b, '=', result )
100 + 50 = 150
>>> result=a-b
>>> print( a, '-', b, '=', result )
100 - 50 = 50
>>> result=a*b
>>> print( a, '*', b, '=', result )
100 * 50 = 5000
>>> result=a/b
>>> print( a, '/', b, '=', result )
100 / 50 = 2.0
>>> |
```

Ln: 19 Col: 4

그림 2-8 실습 최종 결과

■ IDLE 종료

- `exit()` 코드를 입력한 후 Kill? 메시지에서 <확인> 버튼 클릭(또는 [File]-[Exit] 메뉴 선택)

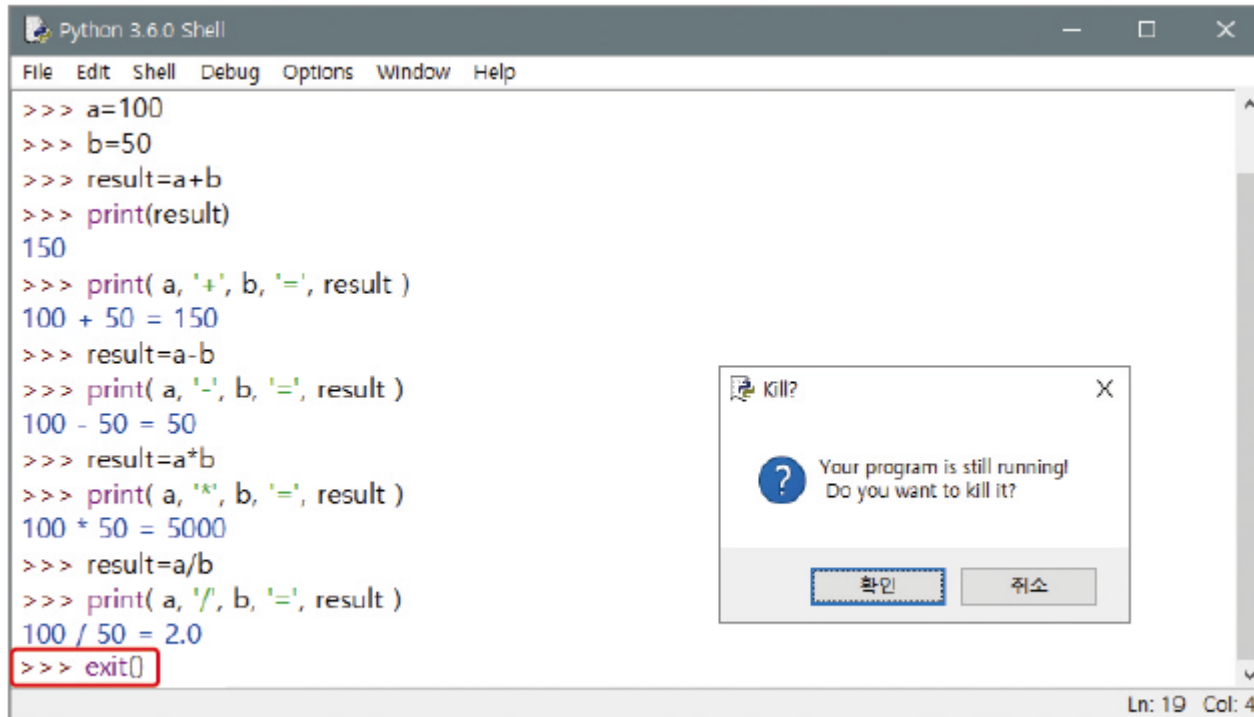
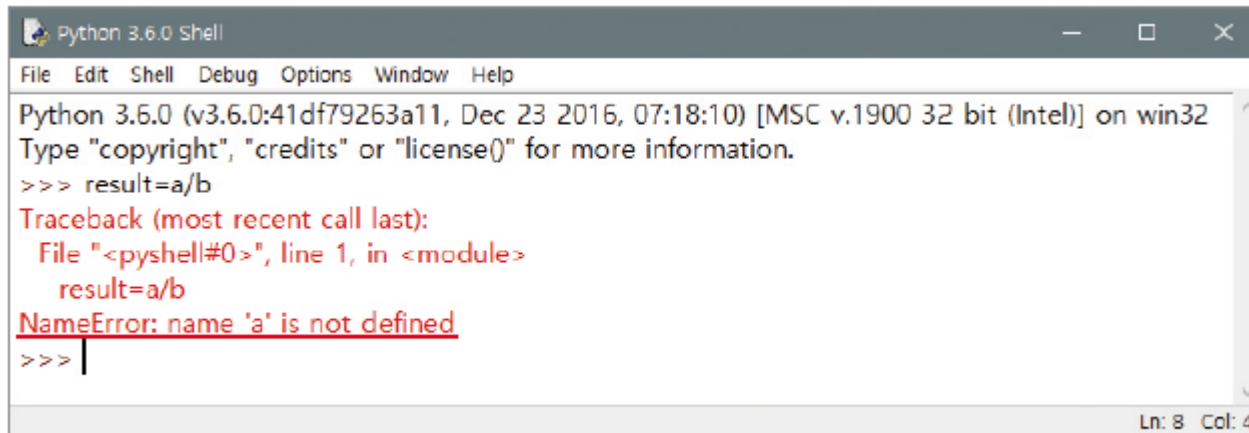


그림 2-9 IDLE 종료

■ 프로그램 저장의 필요성

- IDLE을 실행한 후 앞에서 입력한 나누기를 다시 실행
- 메모리에 저장된 것은 IDLE을 종료하면 모두 사라져 오류 발생. 처음부터 다시 입력해야 함
- 오류 발생



The screenshot shows a Python 3.6.0 Shell window with a menu bar (File, Edit, Shell, Debug, Options, Window, Help). The text inside the window reads: "Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 07:18:10) [MSC v.1900 32 bit (Intel)] on win32", "Type 'copyright', 'credits' or 'license()' for more information.", ">>> result=a/b", "Traceback (most recent call last):", "File "<pyshell#0>", line 1, in <module>", "result=a/b", "NameError: name 'a' is not defined", ">>> |". The status bar at the bottom right shows "Ln: 8 Col: 4".

그림 2-10 오류 발생

■ 파이썬 파일 저장(1)

- 코드가 수실했을 경우는 스크립트 모드 사용(IDLE에서 [File]-[New File] 메뉴 선택)
- 메모장 같은 창인 스크립트 모드에서 코드를 여러 줄 입력 가능. 단, 실행은 되지 않음

Code02-01.py

```
1 a = 100
2 b = 50
3 result = a + b
4 print(a, "+", b, "=", result)
5 result = a - b
6 print(a, "-", b, "=", result)
7 result = a * b
8 print(a, "*", b, "=", result)
9 result = a / b
10 print(a, "/", b, "=", result)
```

Section03 계산기 프로그램 저장

■ 파이썬 파일 저장(2)

- 스크립트 모드에서 [File]-[Save] 메뉴를 선택해 C:\CookPython\ 폴더에 Code02-01 이름으로 저장(확장명 *.py가 자동으로 붙음)

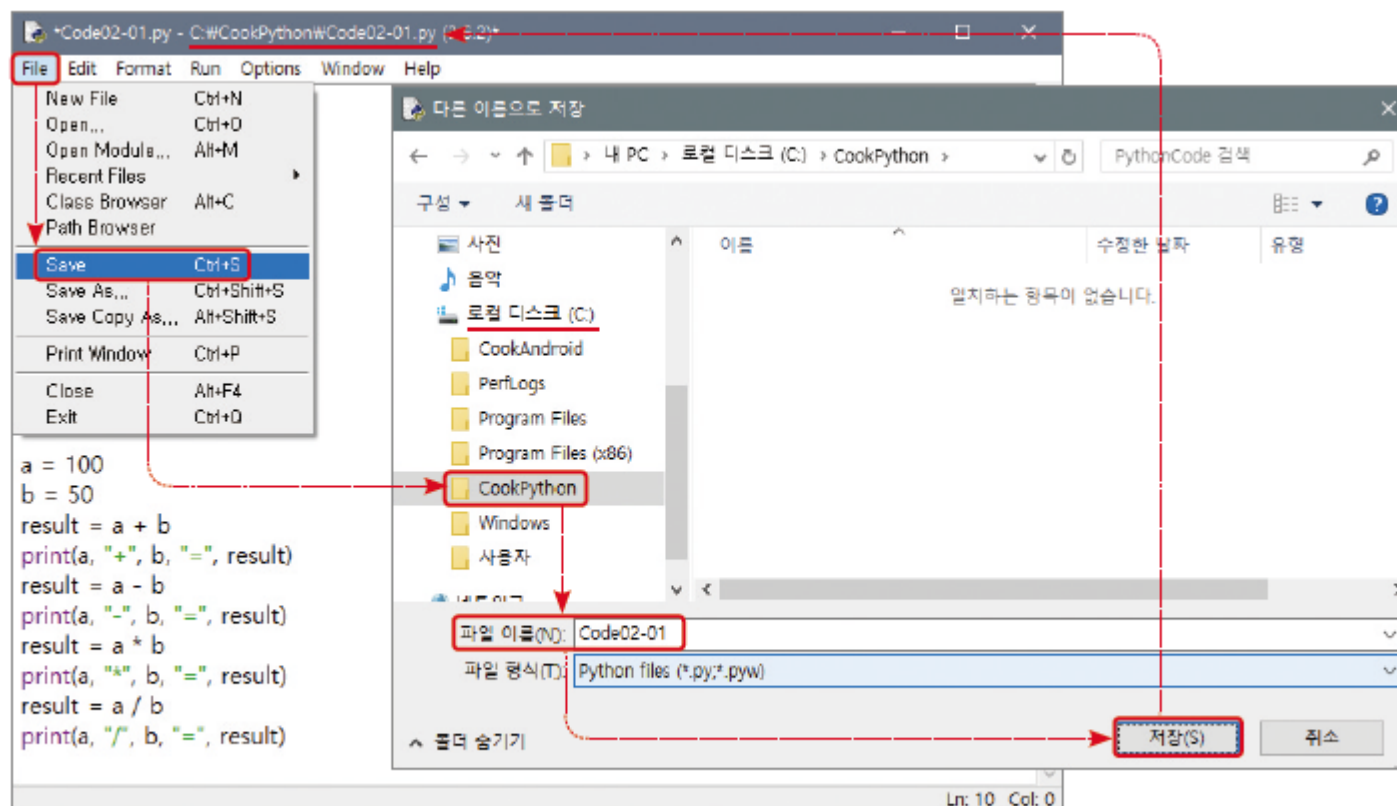


그림 2-12 파이썬 코드를 파일로 저장

Section03 계산기 프로그램 저장

■ 파이썬 파일 실행

- 스크립트 모드에서 [Run]-[Run Module] 메뉴 선택(또는 [F5])

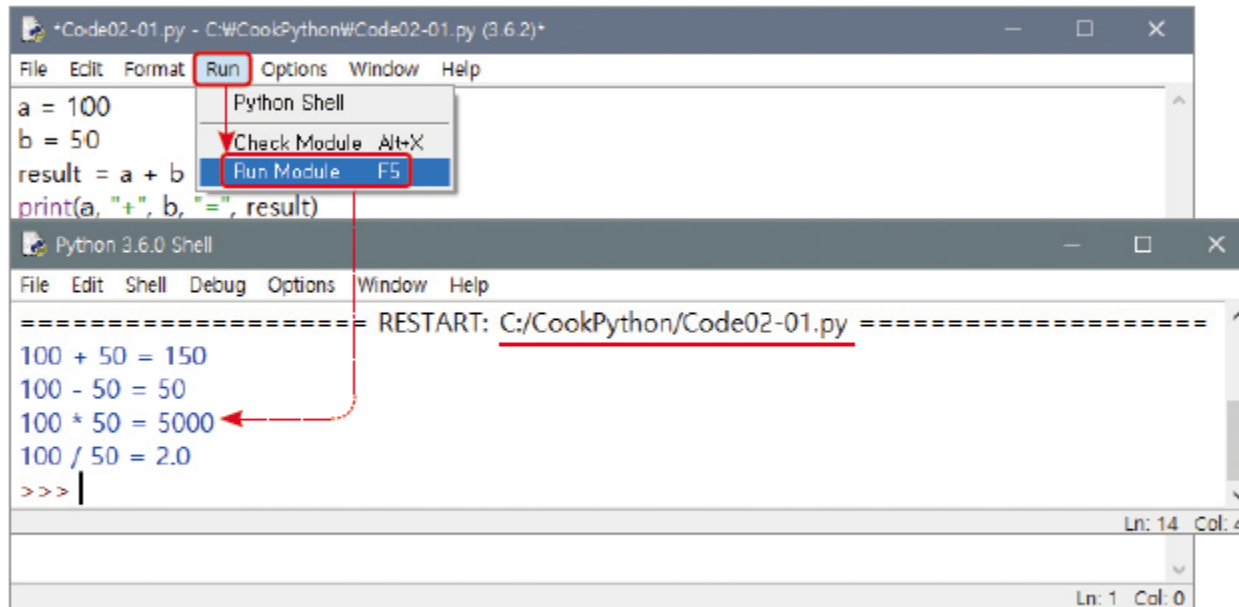


그림 2-13 파이썬 코드를 일괄 실행

Section03 계산기 프로그램 저장

■ 파이썬 파일 열기(1)

- IDLE에서 [File]-[Open] 메뉴를 선택한 후 Code02-01.py 열기

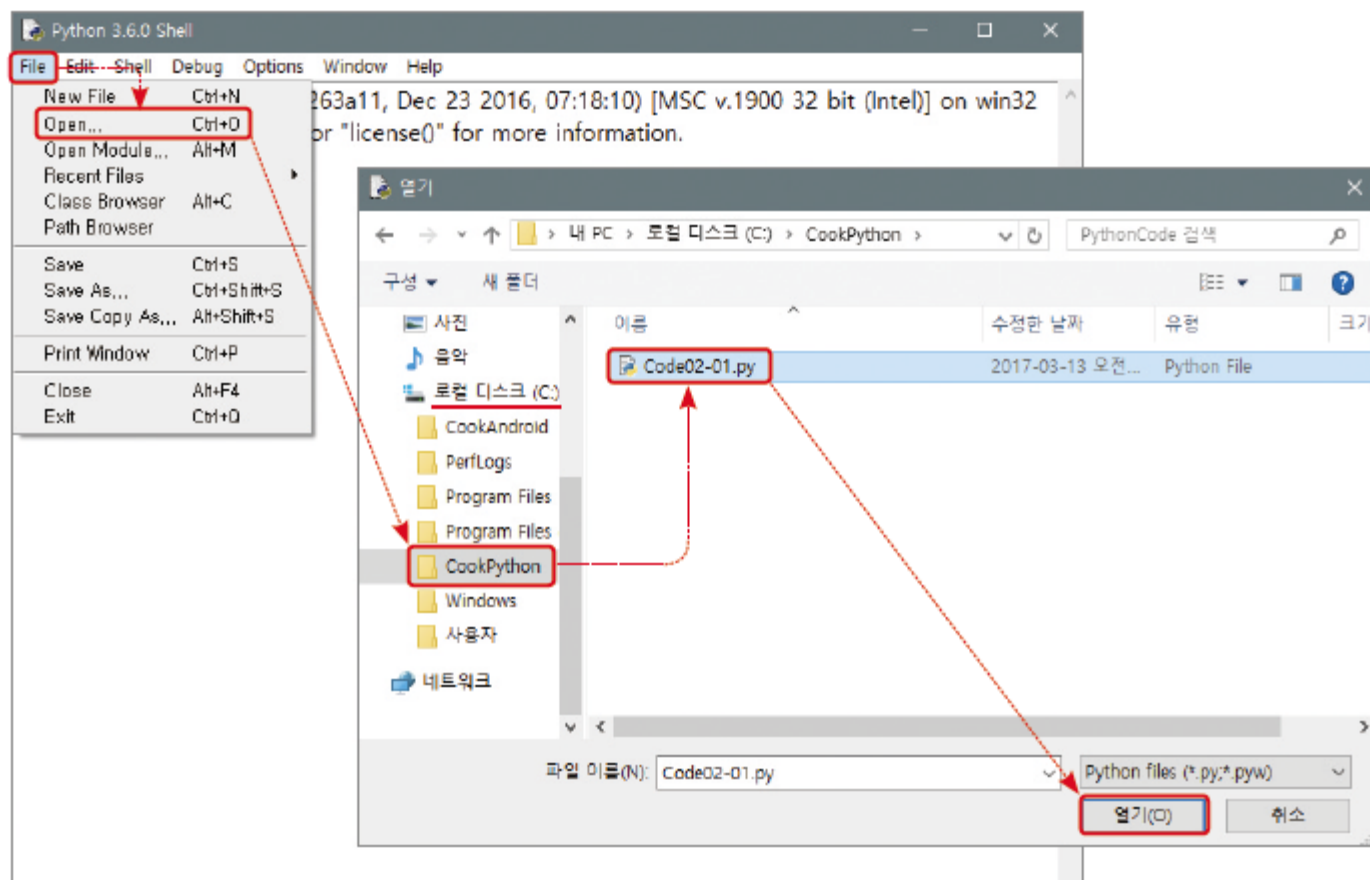


그림 2-16 파이썬 파일 열기

Section03 계산기 프로그램 저장

■ 파이썬 파일 열기(2)

- IDLE에서 [File]-[Open] 메뉴를 선택한 후 Code02-01.py 열어 a와 b의 값을 300과 200으로 수정한 후 [File]-[Save] 메뉴 선택(또는 [Ctrl+S])해 저장 후 [Run]-[Run Module] 메뉴(또는 [F5])로 다시 실행

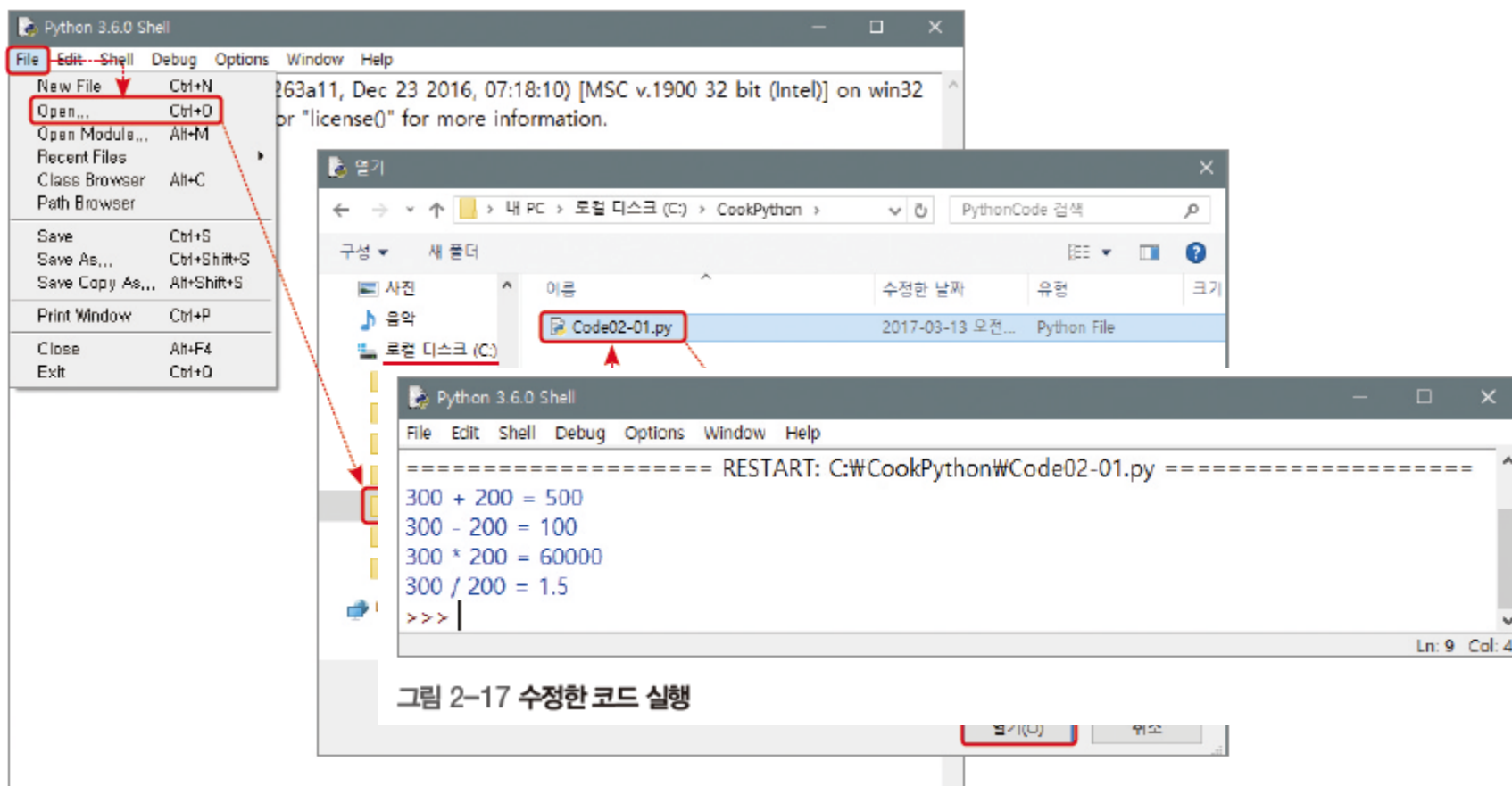


그림 2-16 파이썬 파일 열기

Section03 계산기 프로그램 저장

■ 긴 프로그램을 코딩하는 순서

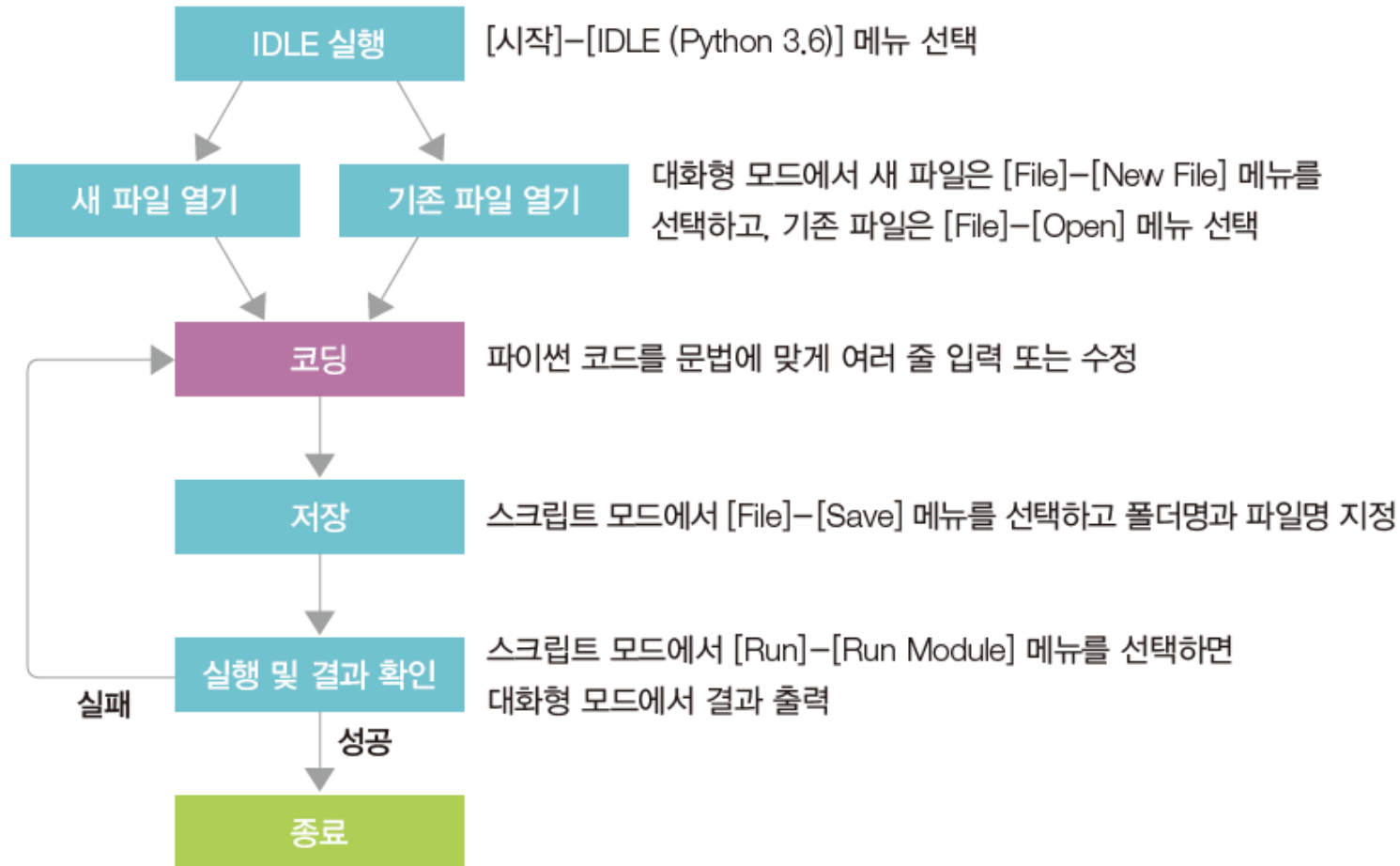
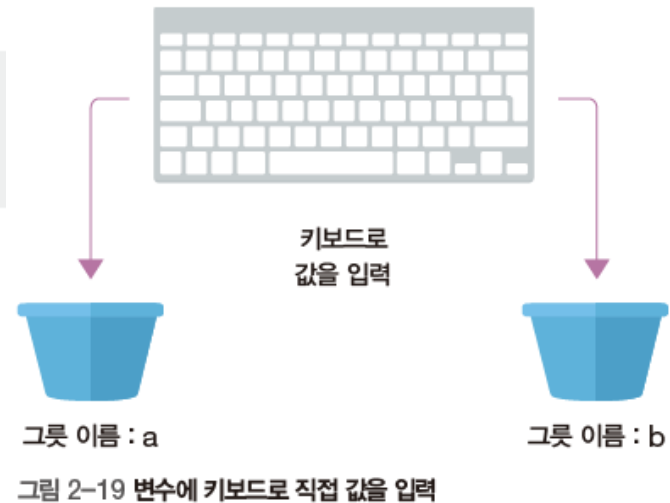


그림 2-18 긴 프로그램을 코딩하는 순서

■ 파이썬 파일 만들어 저장

- 100과 50을 고정적으로 계산하는 것이 아니라 직접 입력한 두 숫자의 사칙 연산을 수행하도록 프로그램 수정

변수 a에 100을 넣는다. → 변수 a에 입력할 값을 키보드로 입력받는다.
변수 b에 50을 넣는다. → 변수 b에 입력할 값을 키보드로 입력받는다.



- 대화형 모드에서 [File]-[New File] 메뉴를 선택해 새 파일을 연 후 스크립트 모드에서 [File]-[Save] 메뉴를 선택해 C:\CookPython\Code02-02.py로 저장

Section04 계산기 프로그램 확장

■ input() 함수를 사용해 값 입력

- Code02-01.py의 1~2행을 input() 함수를 사용하도록 수정 → [F5]를 눌러 실행
→ 숫자 하나를 입력하고 [Enter] → 다시 숫자 하나를 입력하고 [Enter]

Code02-02.py

```
1 a = input()
2 b = input()
3 result = a + b
4 print(a, "+", b, "=", result)
5 result = a - b
6 print(a, "-", b, "=", result)
7 result = a * b
8 print(a, "*", b, "=", result)
9 result = a / b
10 print(a, "/", b, "=", result)
```

← 오류 발생 ?

계산 결과가 틀리거나 오류 발생
input() 함수는 값을 입력받지만 모두
문자열로 취급하기 때문

```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:/CookPython/Code02-02.py =====
100
50
100 + 50 = 10050
Traceback (most recent call last):
  File "C:/CookPython/Code02-02.py", line 5, in <module>
    result=a-b
TypeError: unsupported operand type(s) for -: 'str' and 'str'
>>>
```

Section04 계산기 프로그램 확장

■ input() 함수를 사용해 정수로 변환

- 오른쪽 예처럼 int() 함수를 사용해 정수로 변환

```
int("100")      # 결과는 정수 100
int(100.123)    # 결과는 정수 100
```

- Code02-02.py의 1~2행을 다음과 같이 수정 후 다시 [F5]를 눌러 실행

↓
형변환

Code02-03.py

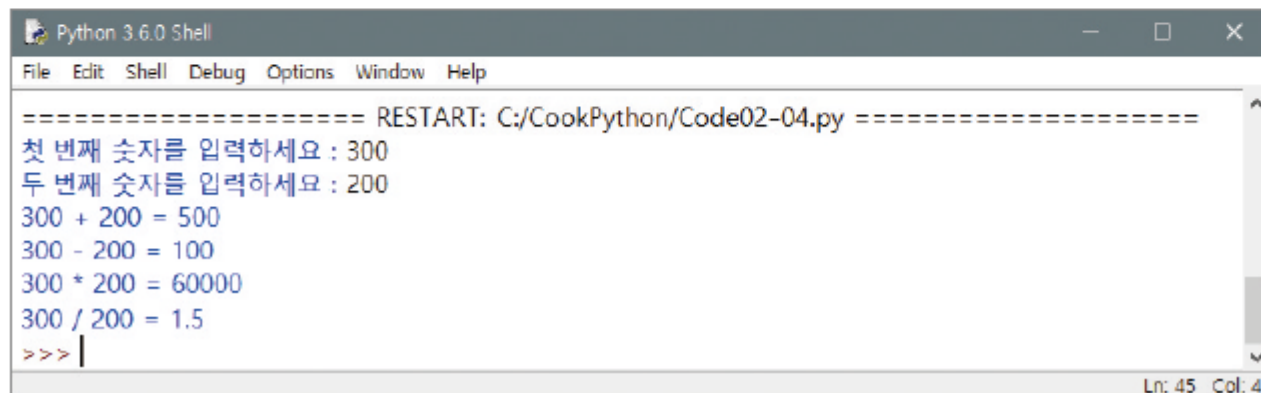
```
1 a = int(input())
2 b = int(input())
```

```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:/CookPython/Code02-03.py =====
100
50
100 + 50 = 150
100 - 50 = 50
100 * 50 = 5000
100 / 50 = 2.0
>>>
```

■ 계산기의 최종 버전

Code02-04.py

```
1 a = int(input("첫 번째 숫자를 입력하세요 : "))
2 b = int(input("두 번째 숫자를 입력하세요 : "))
3 result = a + b
4 print(a, "+", b, "=", result)
5 result = a - b
6 print(a, "-", b, "=", result)
7 result = a * b
8 print(a, "*", b, "=", result)
9 result = a / b
10 print(a, "/", b, "=", result)
```



```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:/CookPython/Code02-04.py =====
첫 번째 숫자를 입력하세요 : 300
두 번째 숫자를 입력하세요 : 200
300 + 200 = 500
300 - 200 = 100
300 * 200 = 60000
300 / 200 = 1.5
>>> |
Ln: 45 Col: 4
```

실습 : 계산기 프로그램을 함수로 만들어 보자

■ 더하기 함수

```
def add(a, b):  
    c = a + b  
    return c
```

← 합트 정의

```
sum = add(10, 20)
print("sum = ", sum)
```

← $\frac{1}{\sigma} \frac{1}{\sigma} \frac{1}{\sigma} \frac{1}{\sigma}$

■ 빼기, 곱하기, 나누기 함수

```
def minus(a, b):
```

```
def multi(a, b):
```

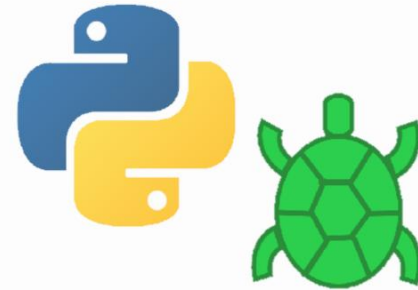
```
def divide(a, b):
```

Section05 터틀 그래픽 프로그램 작성

■ 터틀 프로그램 기본 틀 만들기

- 긴 프로그램은 다음과 같이 세 부분으로 나누어 작성

```
## 함수 선언 부분 ##  
  
## 변수 선언 부분 ##  
  
## 메인(main) 코드 부분 ##
```



- # 기호는 주석(Remark)으로 프로그램의 설명에 해당하고 여러 줄의 주석은 작은따옴표 3개 사용. 뒤에 \를 붙이면 줄을 바꾸어 써도 한 줄로 인식

```
'''  
여러 줄 주석  
입니다.  
'''
```

```
data = '안녕' + \  
        '하세요?' + \  
        '파이썬!'  
print(data)
```

- 터틀 그래픽스 학습 참고 : <https://bit.ly/3tcKUuP>

■ 터틀 프로그램 기본 틀 만들기

■ 함수 선언 부분

- 프로그램에서 사용될 함수들을 만들어 둠
- 함수를 만드는 형식

```
def 함수명(매개변수) :  
    global 사용할_전역_변수  
    # 이 부분에 함수 내용을 코딩
```

■ 변수 선언 부분

- 프로그램 전체에서 사용될 전역 변수를 미리 선언

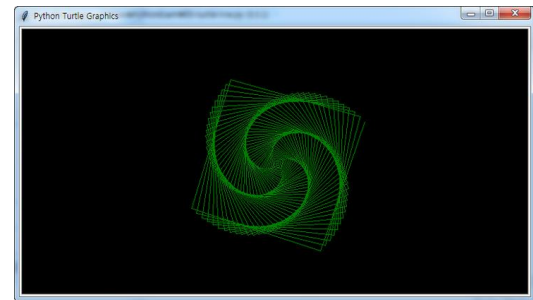
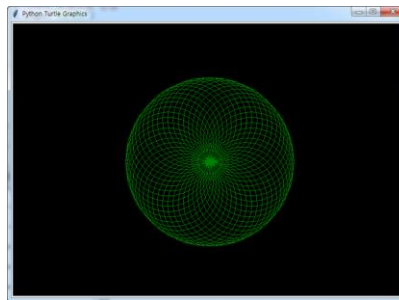
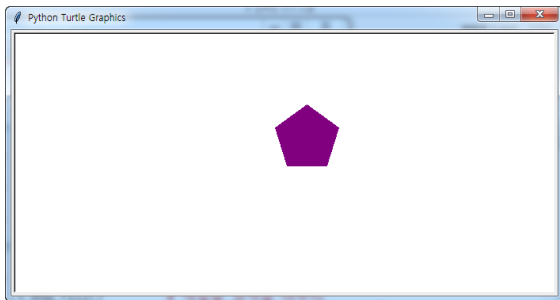
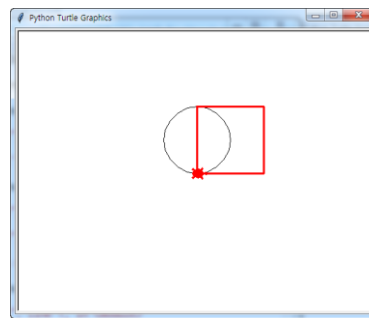
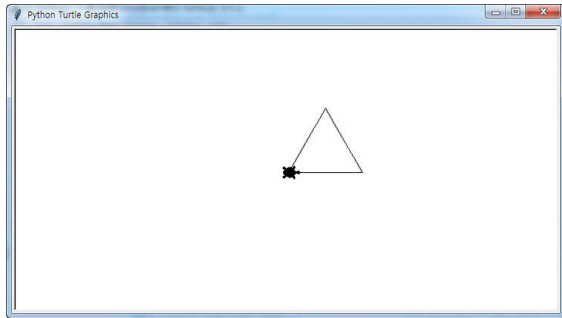
■ 메인(main) 코드 부분

- 프로그램이 실제로 처리되는 주요한 부분(Code02-04의 3~10행)

Section05 터틀 그래픽 프로그램 작성

- 참고 사이트

- <https://bit.ly/3slyl5V>
- <https://docs.python.org/ko/3.10/library/turtle.html>



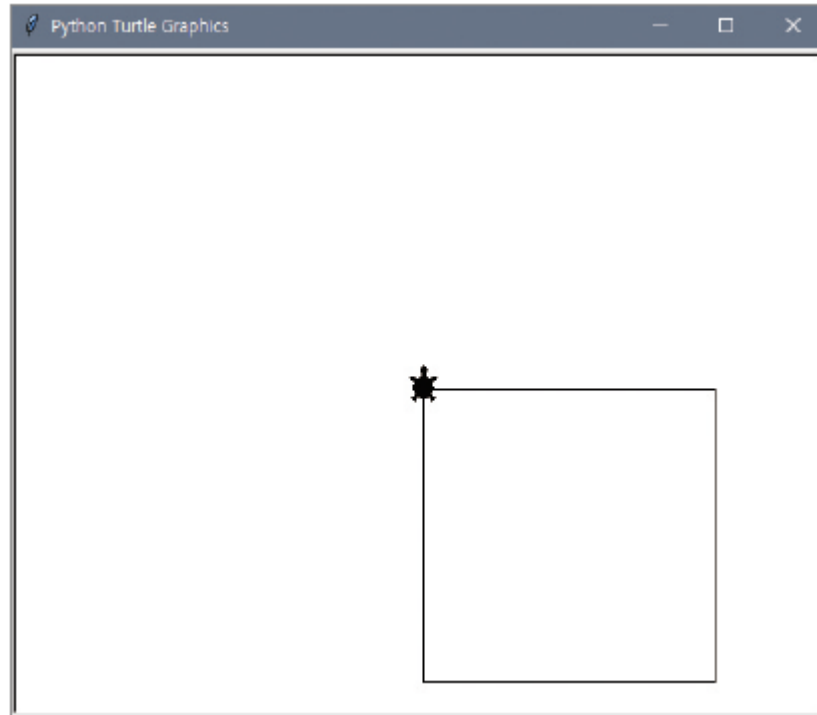
Section05 터틀 그래픽 프로그램 작성

■ 터틀 프로그램 기본 틀 만들기

- 윈도우창에 거북이가 나오는 간단한 프로그램 : 거북이가 나와서 정사각형을 그림

Code02-05.py

```
1 import turtle
2
3 turtle.shape('turtle')
4
5 turtle.forward(200)
6 turtle.right(90)
7 turtle.forward(200)
8 turtle.right(90)
9 turtle.forward(200)
10 turtle.right(90)
11 turtle.forward(200)
12
13 turtle.done()
```



■ 터틀 프로그램 기본 틀 만들기

- Code02-05.py를 프로그램 틀 형태로 수정

Code02-06.py

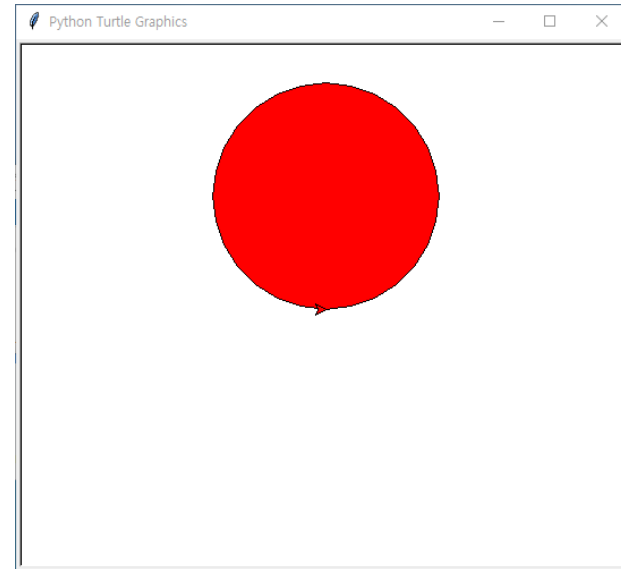
```
1 import turtle
2
3 ## 함수 선언 부분 ##
4
5 ## 변수 선언 부분 ##
6 myT = None
7
8 ## 메인 코드 부분 ##
9 myT = turtle.Turtle()
10 myT.shape('turtle')
11
12 for i in range(0, 4) :
13     myT.forward(200)
14     myT.right(90)
15
16 myT.done()
```

실습 : 터틀 원 그리기

```
import turtle as t

t.fillcolor('red')
t.begin_fill()
t.circle(100)
t.end_fill()

# 그래픽 창이 열린 상태로 유지
t.done()
```



실습 : 터틀 원 그리기

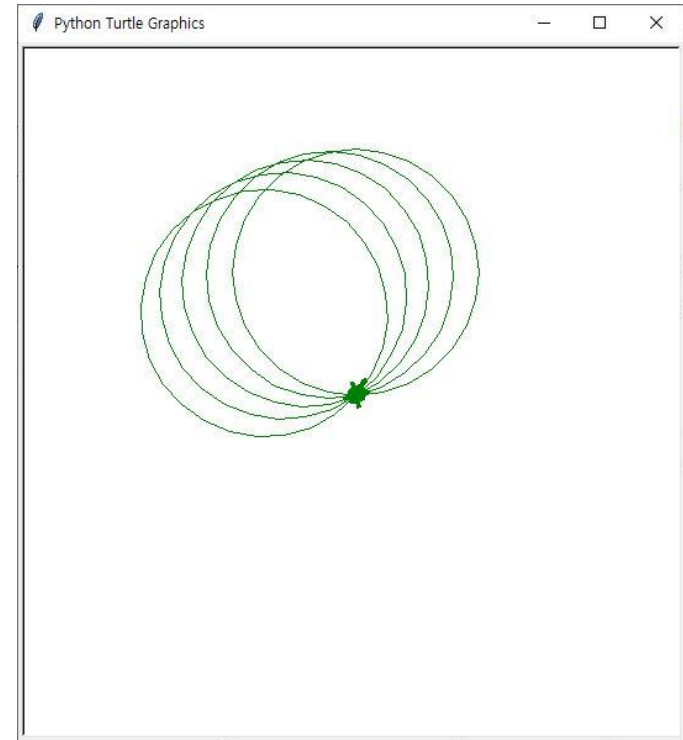
```
import turtle as t

t.shape("turtle")
t.color("green")
t.speed(5)

r = int(input("몇)개의 원을 그릴까요?"))

for i in range(5):
    # 반지름 100으로 원 그리기
    t.circle(100)
    # 360/r 각도 이동
    t.left(360/r)

t.done()
```



■ 구현할 기능 계획



그림 2-20 터틀 그래픽 프로그램의 구현 기능 계획

- 기능 1 : 마우스 왼쪽 버튼을 누르면 거북이가 클릭한 지점까지 임의의 색상으로 선을 그리면서 따라오도록 한다.
- 기능 2 : 마우스 오른쪽 버튼을 누르면 거북이가 클릭한 지점까지 선을 그리지 않고 이동만 하도록 한다.
- 기능 3 : 마우스 가운데 버튼을 누르면 거북이가 임의로 크기를 확대 또는 축소한다.

■ 필요한 변수 준비

- 거북이로 그릴 선의 두께(pSize)와 거북이의 크기(tSize) 변수
- 색상을 표현하는 빨강(r), 초록(g), 파랑(b) 변수

```
pSize, tSize = 10, 0  
r, g, b = 0.0, 0.0, 0.0
```

또는

```
r = 0.0  
g = 0.0  
b = 0.0
```

■ 기능1 구현

- 마우스 왼쪽 버튼을 누르면 거북이가 클릭한 지점까지 임의의 색상으로 선을 그리면서 따라오도록 하는 함수

```
def screenLeftClick(x, y) :  
    global r, g, b  
    turtle.pencolor((r, g, b))  
    turtle.pendown()  
    turtle.goto(x, y)
```

■ 기능2 구현

- 마우스 오른쪽 버튼을 누르면 거북이가 클릭한 지점까지 선을 그리지 않고 이동만 하도록 하는 함수

```
def screenRightClick(x, y) :  
    turtle.penup()  
    turtle.goto(x, y)
```

■ 기능3 구현

- 마우스 가운데 버튼을 누르면 거북이가 임의로 크기를 확대 또는 축소하는 함수
- 마우스 가운데 버튼을 누를 때는 선의 색상이 임의로 선택되도록 하고 거북이의 크기도 1 부터 9까지 임의로 설정되도록 함

```
def screenMidClick(x, y) :  
    global r, g, b  
    tSize = random.randrange(1, 10)  
    turtle.shapesize(tSize)  
    r = random.random()  
    g = random.random()  
    b = random.random()
```

Section05 터틀 그래픽 프로그램 작성

■ 터틀 그래픽 프로그램 완성

Code02-07.py

```
1 import turtle
2 import random
3
4 ## 함수 선언 부분 ##
5 def screenLeftClick(x, y):
6     global r, g, b
7     turtle.pencolor((r, g, b))
8     turtle.pendown()
9     turtle.goto(x,y)
10
11 def screenRightClick(x, y):
12     turtle.penup()
13     turtle.goto(x, y)
14
15 def screenMidClick(x, y):
16     global r, g, b
17     tSize = random.randrange(1, 10)
18     turtle.shapesize(tSize)
19     r = random.random()
20     g = random.random()
21     b = random.random()
22
23 ## 변수 선언 부분 ##
24 pSize = 10
25 r, g, b = 0.0, 0.0, 0.0
26
27 ## 메인 코드 부분 ##
28 turtle.title('거북이로 그림 그리기')
29 turtle.shape('turtle')
30 turtle.pensize(pSize)
31
32 turtle.onscreenclick(screenLeftClick, 1)
33 turtle.onscreenclick(screenMidClick, 2)
34 turtle.onscreenclick(screenRightClick, 3)
35
36 turtle.done()
```



Section05 터틀 그래픽 프로그램 작성

표 2-1 Code02-07.py에 사용된 기타 함수

함수	설명
<code>turtle.title(제목)</code>	윈도창의 제목을 설정한다.
<code>turtle.pensize(펜 두께)</code>	그릴 선의 두께를 설정한다.
<code>turtle.onscreenclick(함수명, 번호)</code>	윈도창을 마우스로 클릭하면 '함수명' 함수가 작동한다. 1은 마우스 왼쪽 버튼, 2는 마우스 가운데 버튼, 3은 마우스 오른쪽 버튼을 지정한다.

SELF STUDY 2-1

터틀 그래픽 프로그램을 수정해서 마우스 왼쪽 버튼과 마우스 가운데 버튼의 기능을 통합해 보자. 즉 마우스 왼쪽 버튼만 눌러도 임의의 색상이 지정되고 거북이의 크기가 변경되면서 선이 그려지도록 하자.



Thank You
