

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



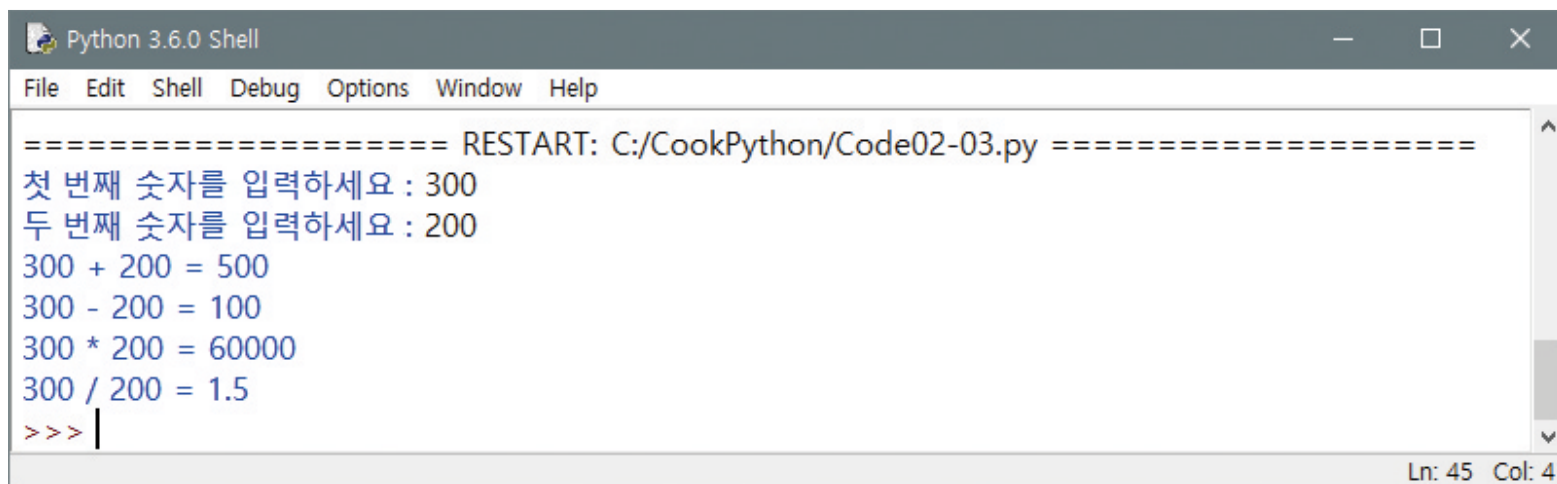
참고문헌

1. 파이썬 for Beginner/ 우재남/ 한빛아카데미 /2017
2. 두근두근 파이썬/ 천인국/생능출판사/2017
3. 파이썬/ Y. Daniel Liang/ 생능출판사/2016

Section01 이 장에서 만들 프로그램

■ [프로그램1] 간단 계산기

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



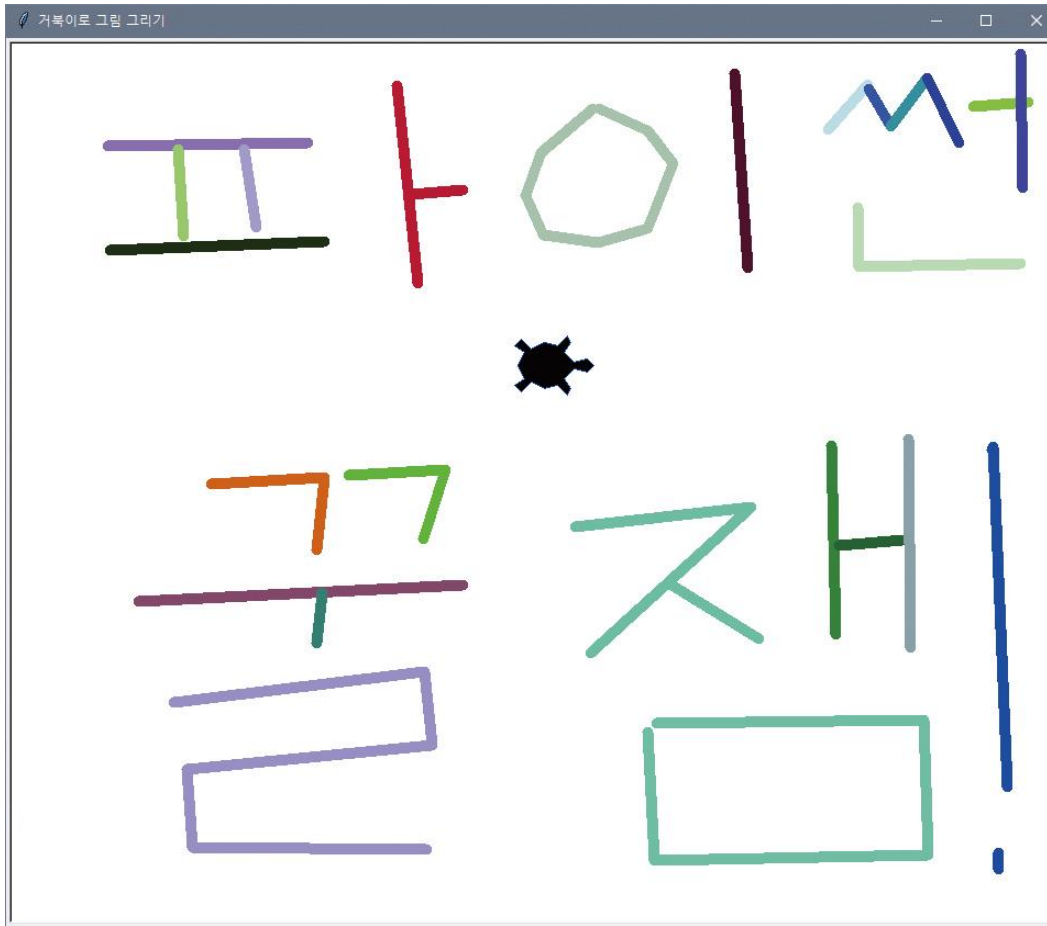
```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:/CookPython/Code02-03.py =====
첫 번째 숫자를 입력하세요 : 300
두 번째 숫자를 입력하세요 : 200
300 + 200 = 500
300 - 200 = 100
300 * 200 = 60000
300 / 200 = 1.5
>>> |
```

Ln: 45 Col: 4

Section01 이 장에서 만들 프로그램

■ [프로그램2] 간단 계산기

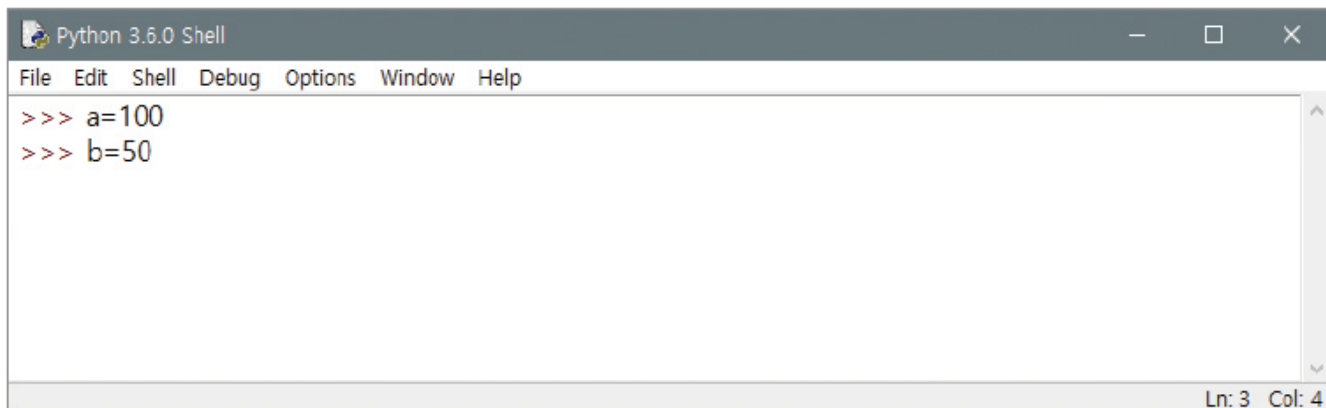
- 마우스로 거북이 커서를 움직여 재미있는 그림을 그리는 프로그램



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

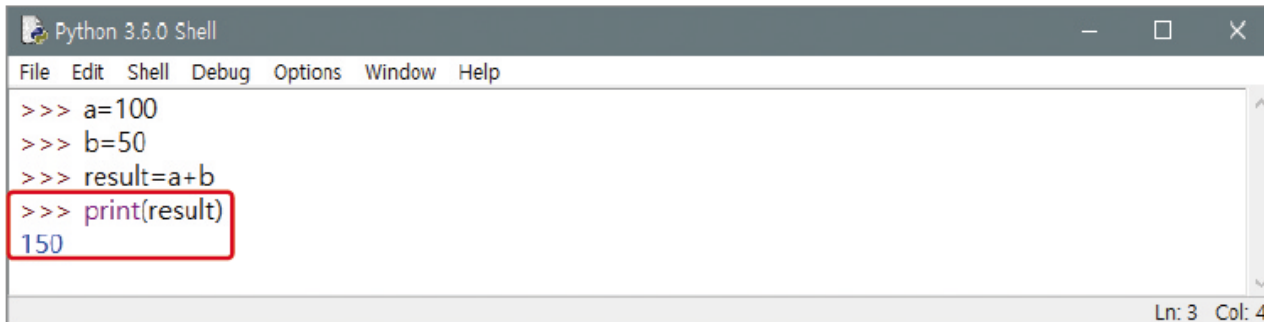


그림 2-4 더하는 작업

Section02 계산기 프로그램의 기본 기능 구현

■ 더한 결과 출력

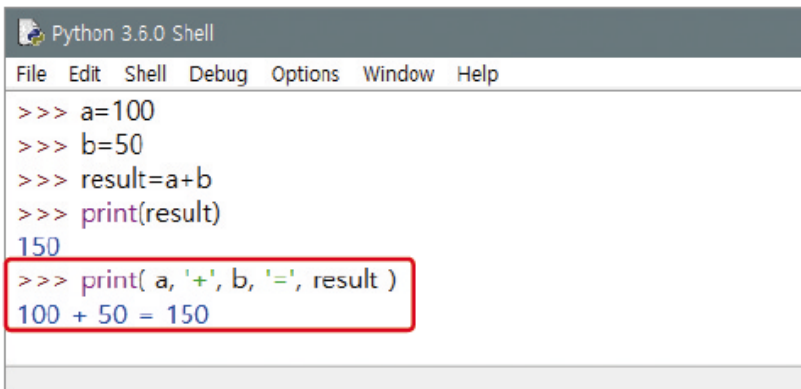
- result 그릇의 내용만 출력



```
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

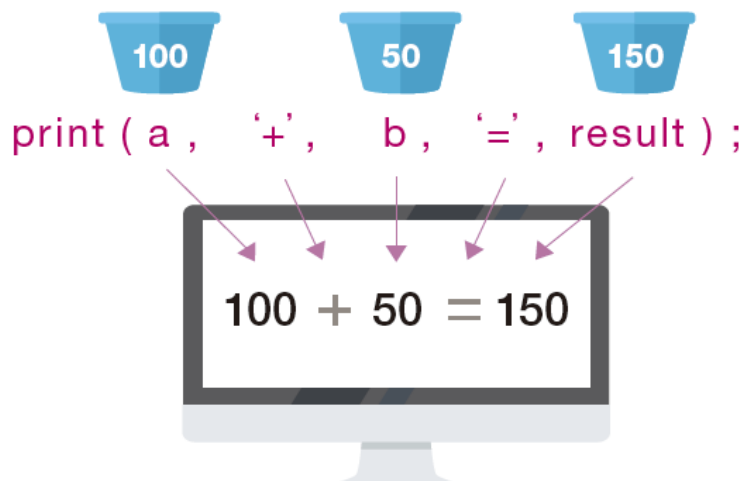


그림 2-7 print() 함수로 모두 출력

여기서 잠깐



함수(Function)

함수는 특정 기능을 하도록 미리 만들어져 있거나 새로 만드는 것을 말한다. 함수는 뒤에 괄호가 붙는다. `print()`는 괄호 안의 내용을 출력해 함수로 파이썬이 미리 만들어서 제공하는 것이다. 이 외에도 파이썬은 매우 많은 함수를 제공하지만 프로그래머가 원하는 모든 함수를 파이썬에서 제공할 수는 없어 직접 만들어서 사용하기도 한다. 이 장 후반부에서 함수를 만드는 방법을 간단히 배우고, 더 자세한 내용은 9장에서 다룬다.

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. The commands are: a=100, b=50, result=a+b, print(result) (output: 150), print(a, '+', b, '=', result) (output: 100 + 50 = 150), result=a-b, print(a, '-', b, '=', result) (output: 100 - 50 = 50), result=a*b, print(a, '*', b, '=', result) (output: 100 * 50 = 5000), result=a/b, print(a, '/', b, '=', result) (output: 100 / 50 = 2.0). The last four lines of code and their outputs are enclosed in a red rectangular box. The status bar at the bottom right shows 'Ln: 19 Col: 4'.

```
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
>>> 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

Section03 계산기 프로그램 저장

■ IDLE 종료

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

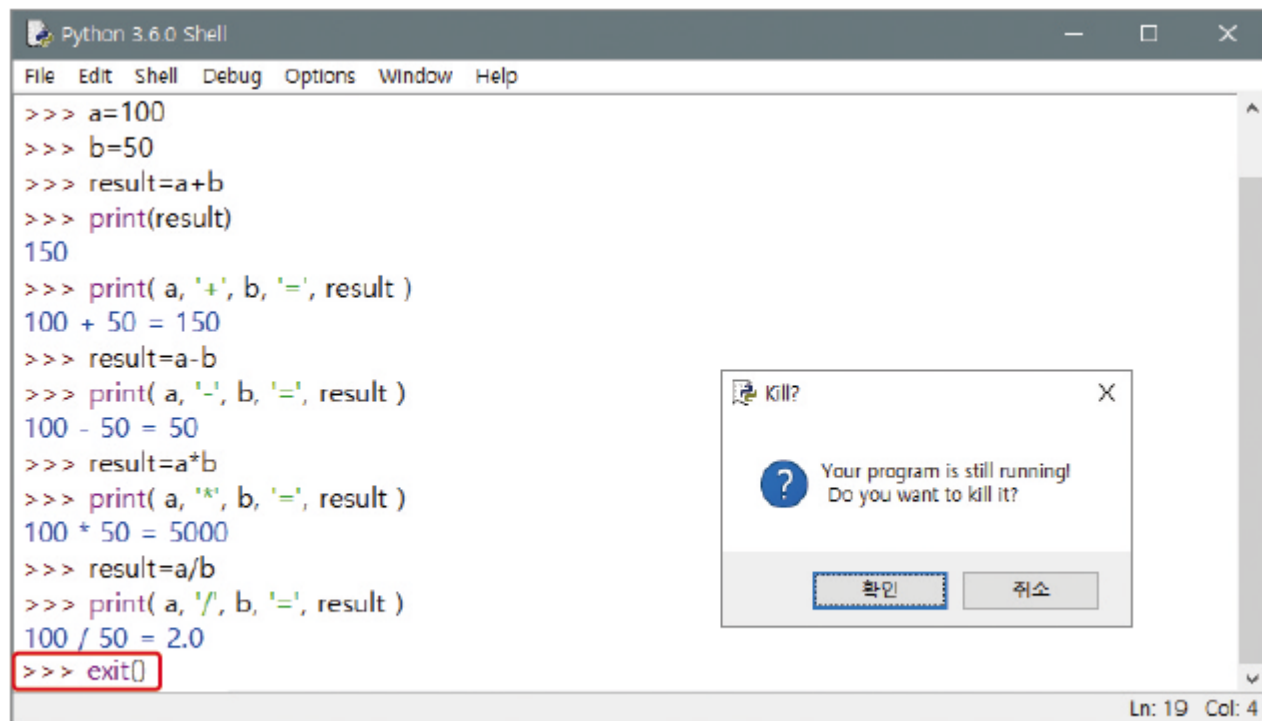


그림 2-9 IDLE 종료

Section03 계산기 프로그램 저장

■ 프로그램 저장의 필요성

- IDLE을 실행한 후 앞에서 입력한 나누기를 다시 실행
- 메모리에 저장된 것은 IDLE을 종료하면 모두 사라져 오류 발생. 처음부터 다시 입력해야 함
- 파이썬 코드는 하드디스크나 USB에 저장해 놓는 방법 사용

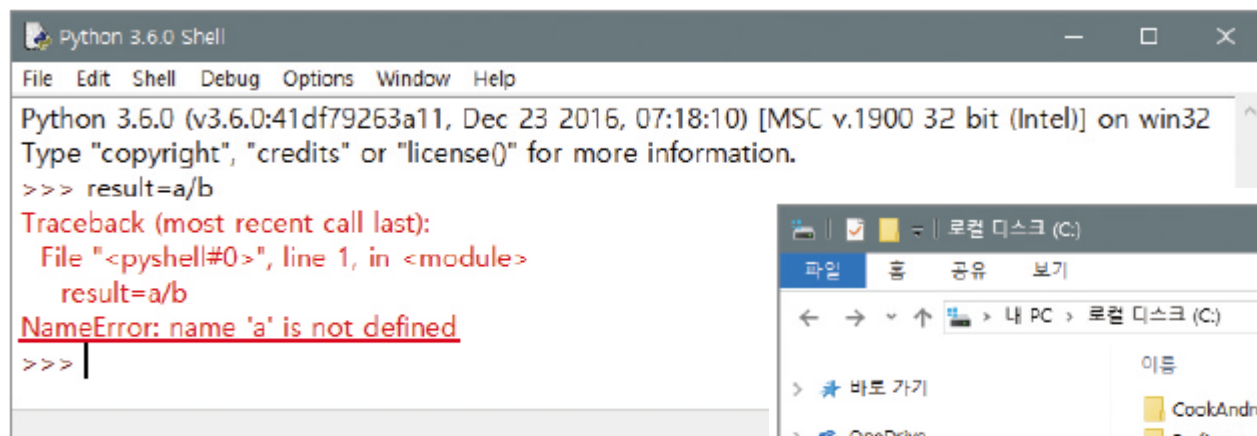


그림 2-10 오류 발생

- 이 책은 C 드라이브(C:\) 바로 아래에 CookPython 폴더를 만들어 사용

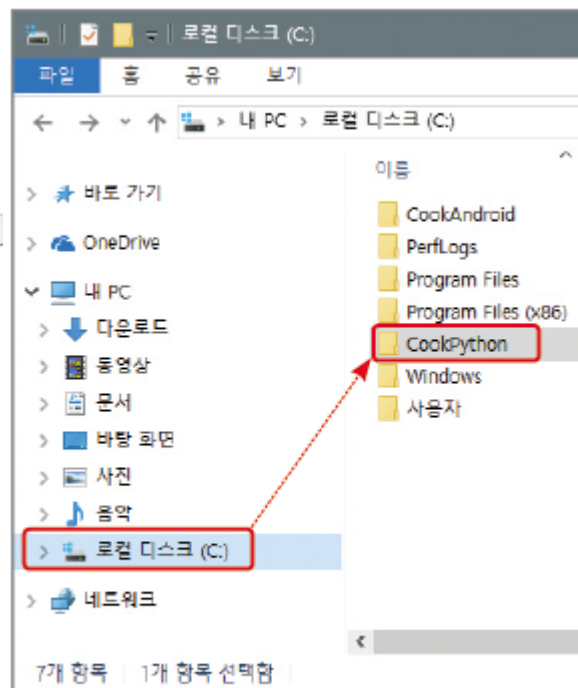


그림 2-11 실습용 폴더 생성

■ 파이썬 파일 저장(1)

- 코드가 수싵 줄인 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)

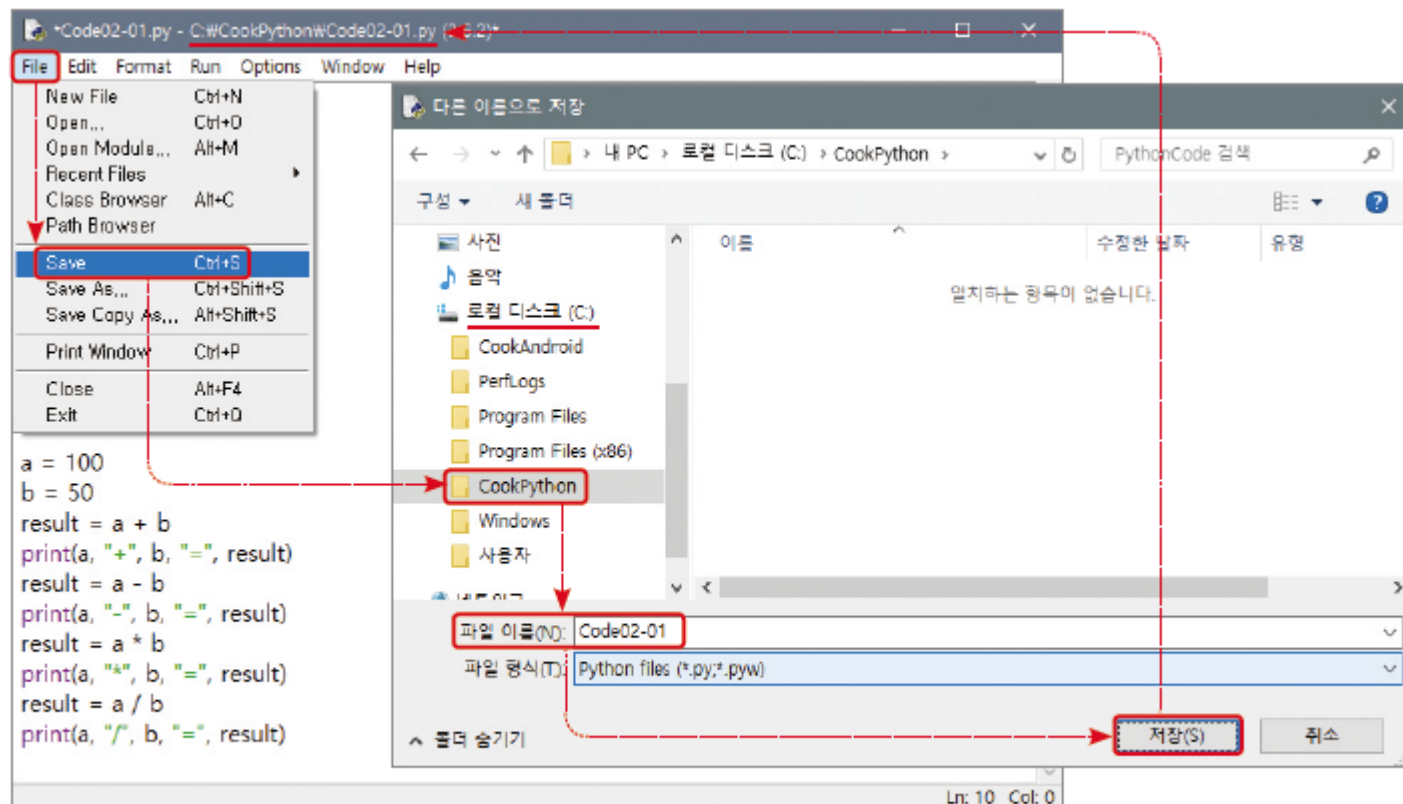


그림 2-12 파이썬 코드를 파일로 저장

Section03 계산기 프로그램 저장

■ 파이썬 파일 실행

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

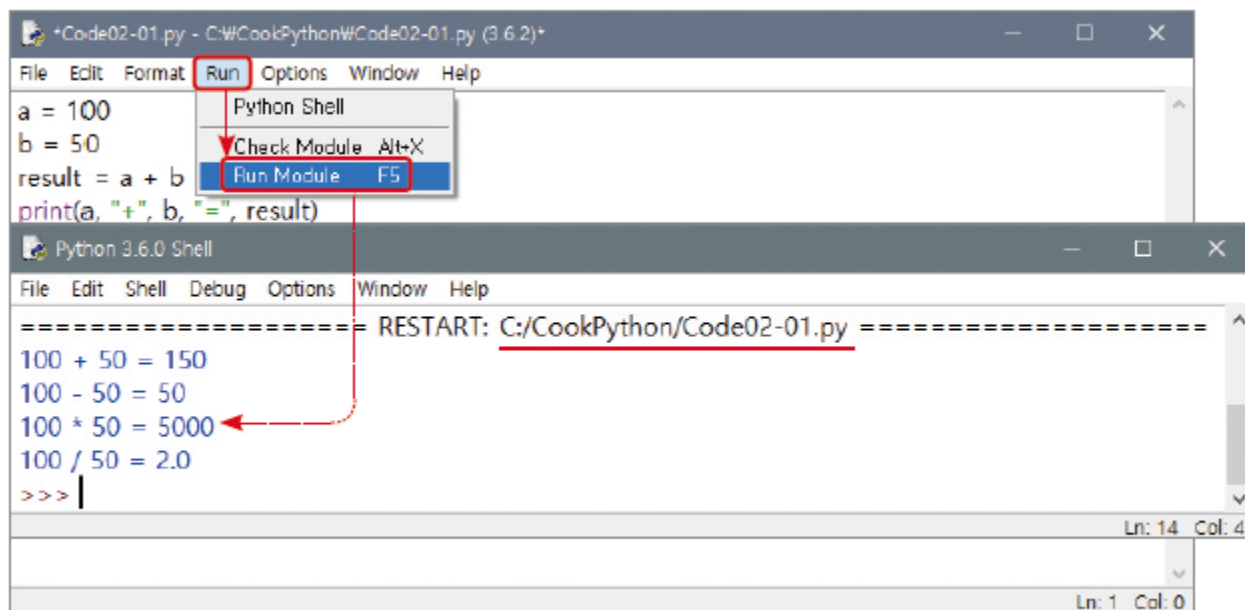


그림 2-13 파이썬 코드를 일괄 실행

- 파이썬 코드를 파일 탐색기에서 바로 실행하는 방법은 교재 41쪽의 [여기서 잠깐] 참고

Section03 계산기 프로그램 저장

■ 파이썬 파일 열기(1)

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

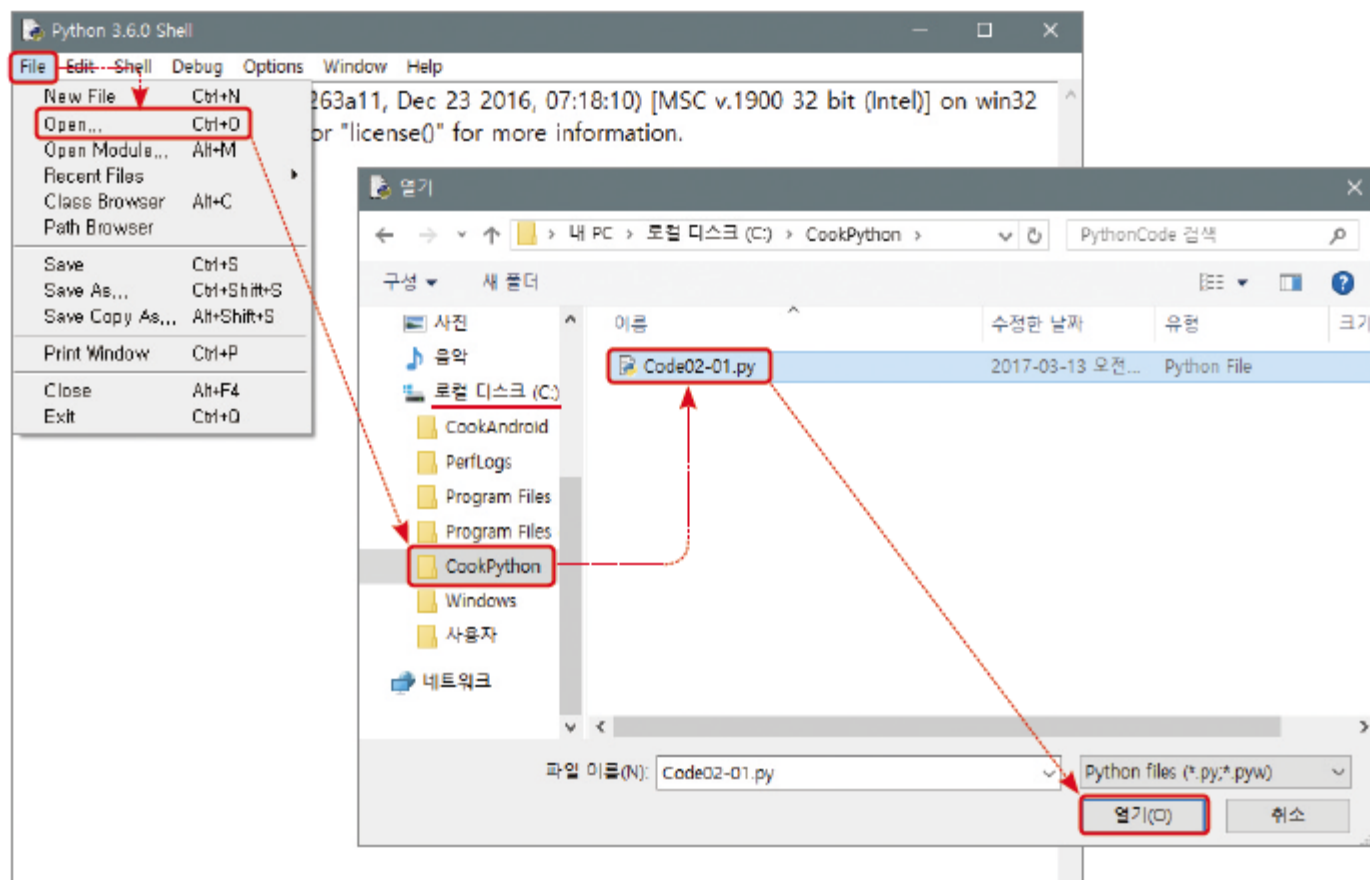


그림 2-16 파이썬 파일 열기

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

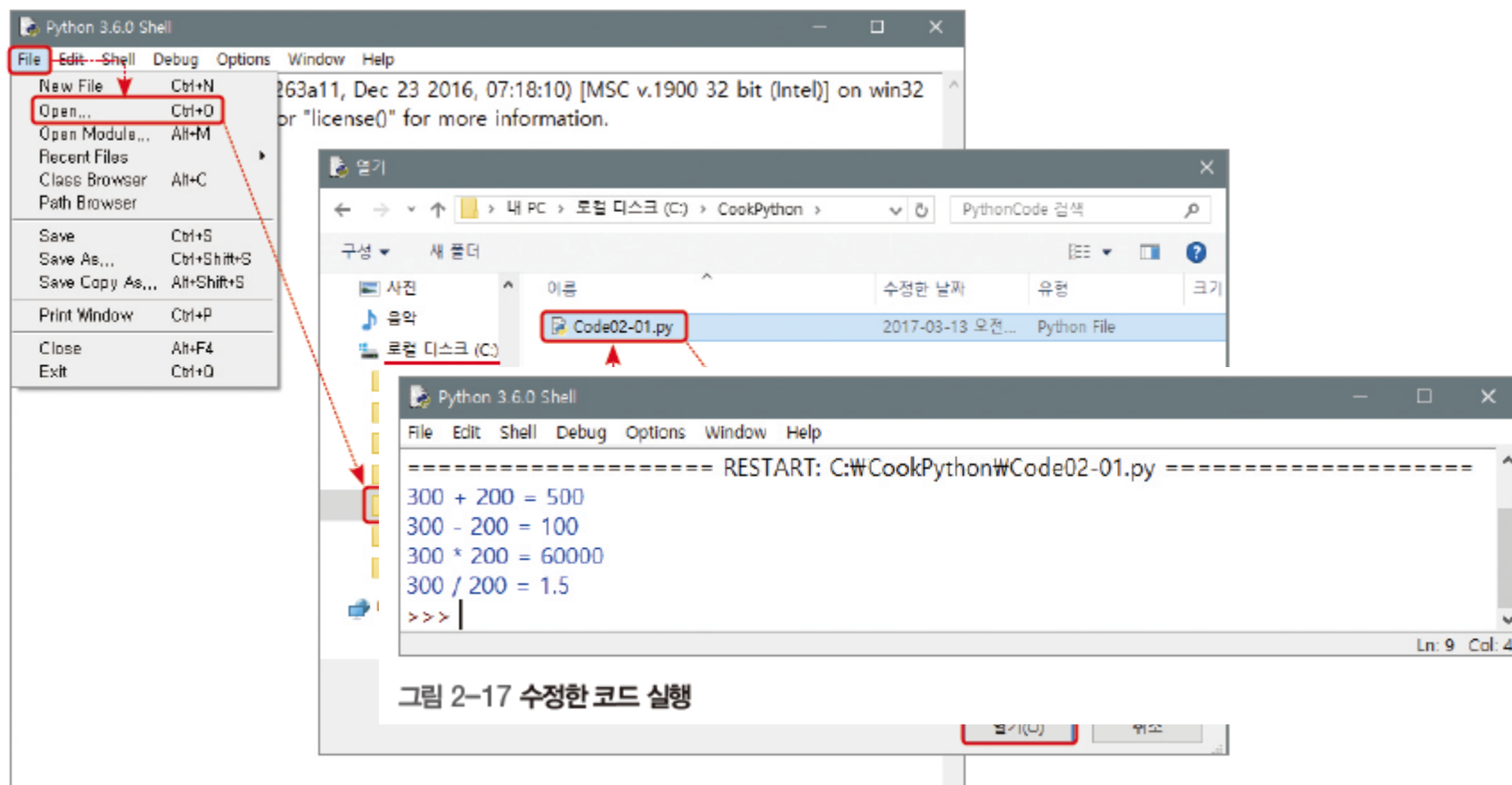


그림 2-16 파이썬 파일 열기

Section03 계산기 프로그램 저장

■ 긴 프로그램을 코딩하는 순서

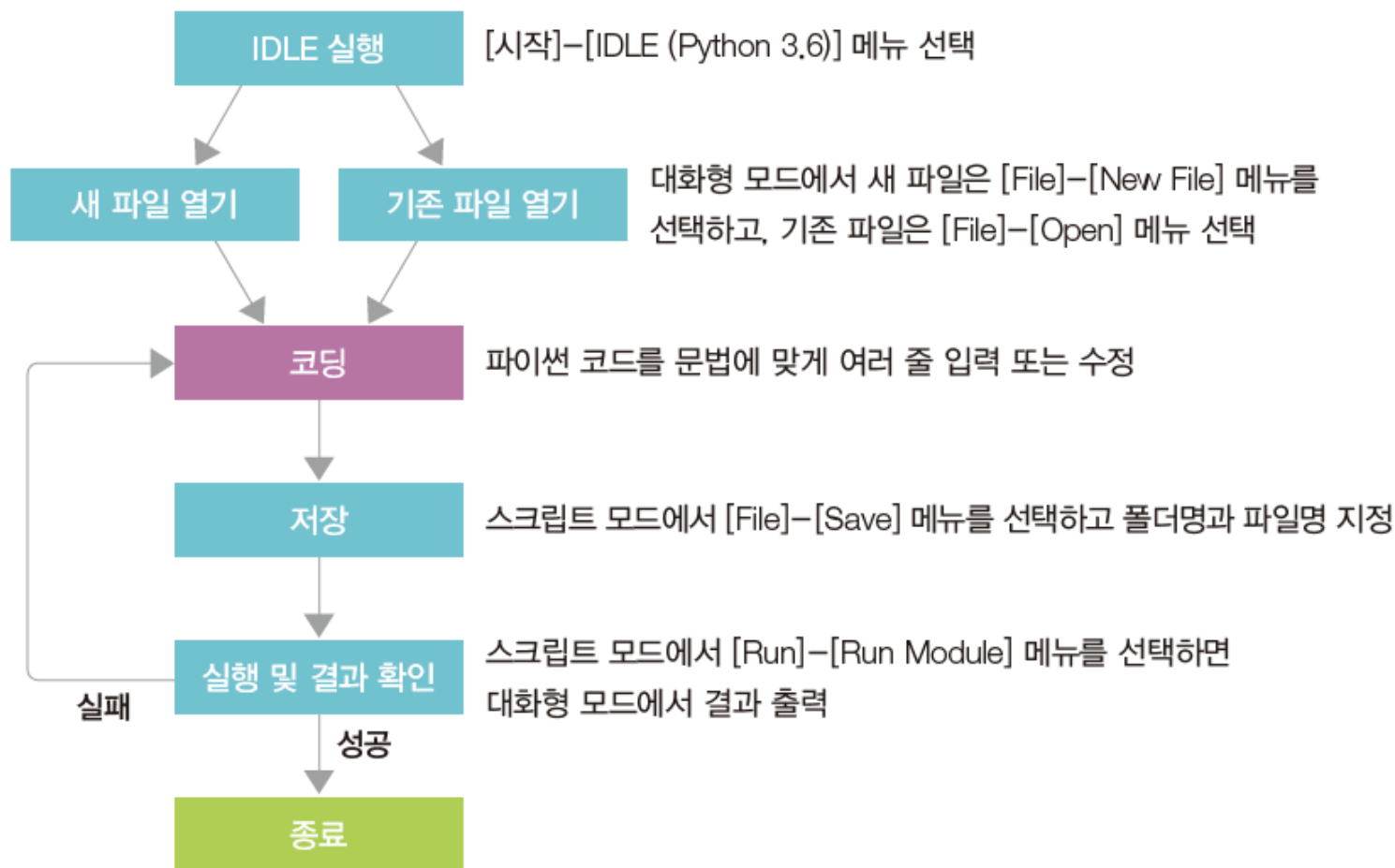
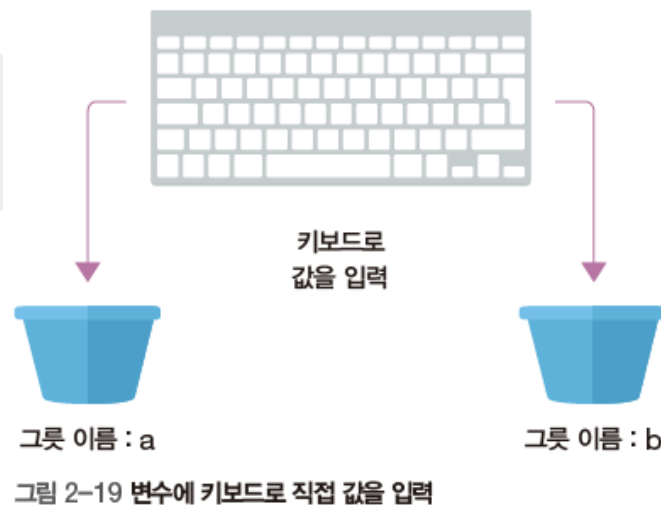


그림 2-18 긴 프로그램을 코딩하는 순서

■ 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)
```

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 ← 사용자가 입력한 값
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

Code02-03.py

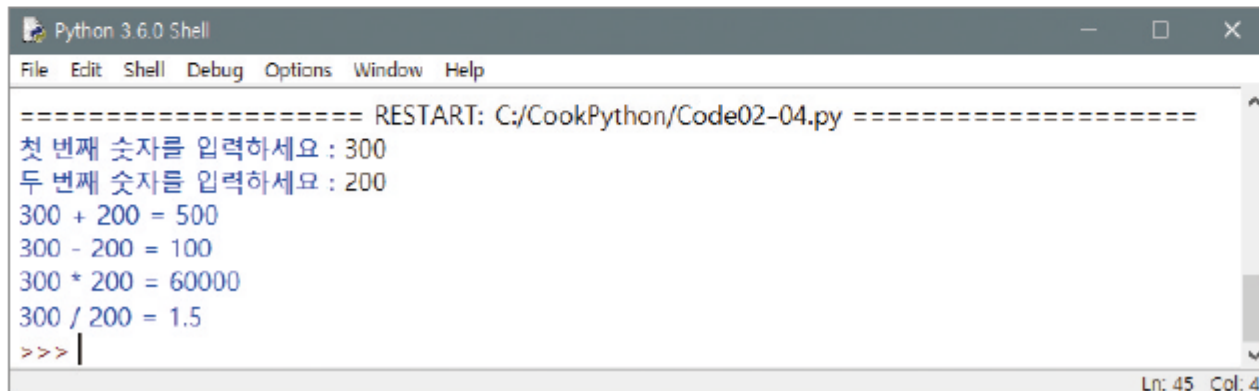
```
1 a = int(input())
2 b = int(input())
```

```
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

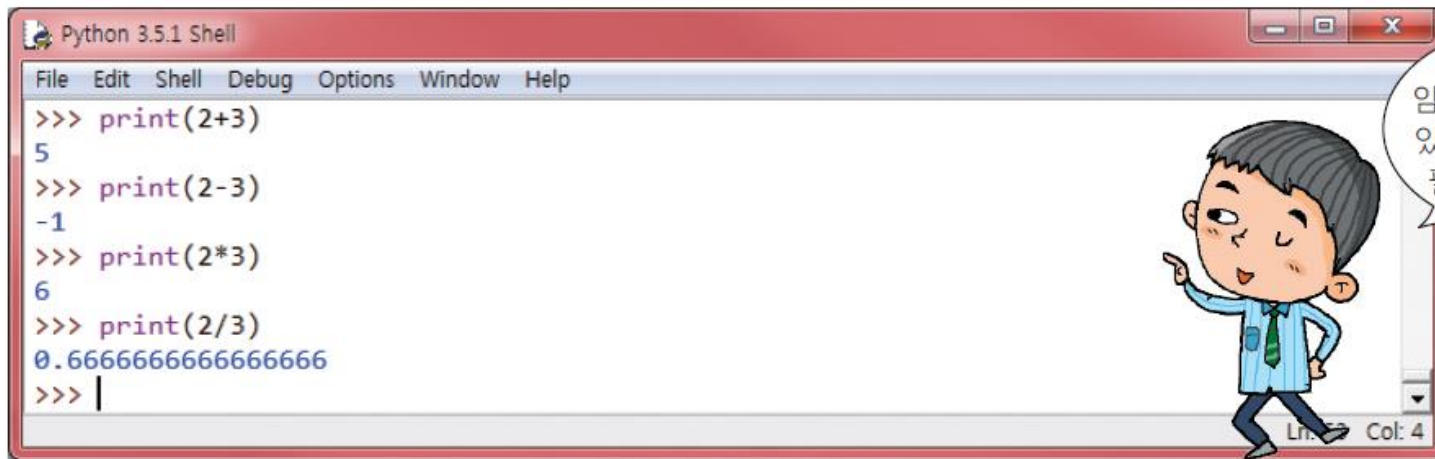
Code02-04.py

```
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
```

계산하기 #1

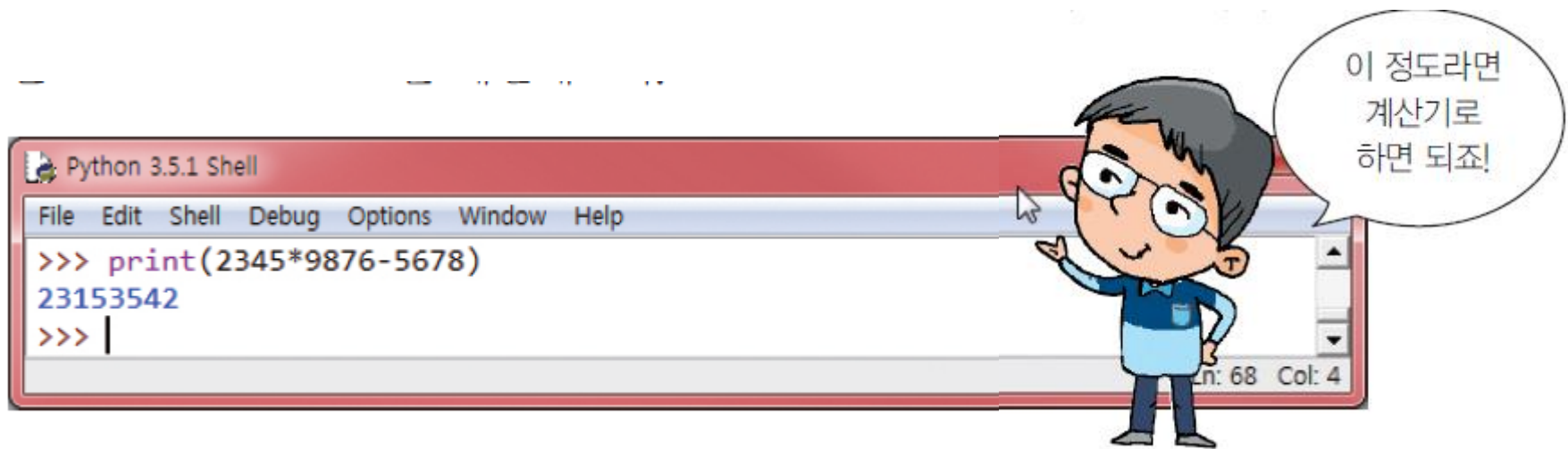


```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
>>> print(2+3)
5
>>> print(2-3)
-1
>>> print(2*3)
6
>>> print(2/3)
0.6666666666666666
>>> |
```

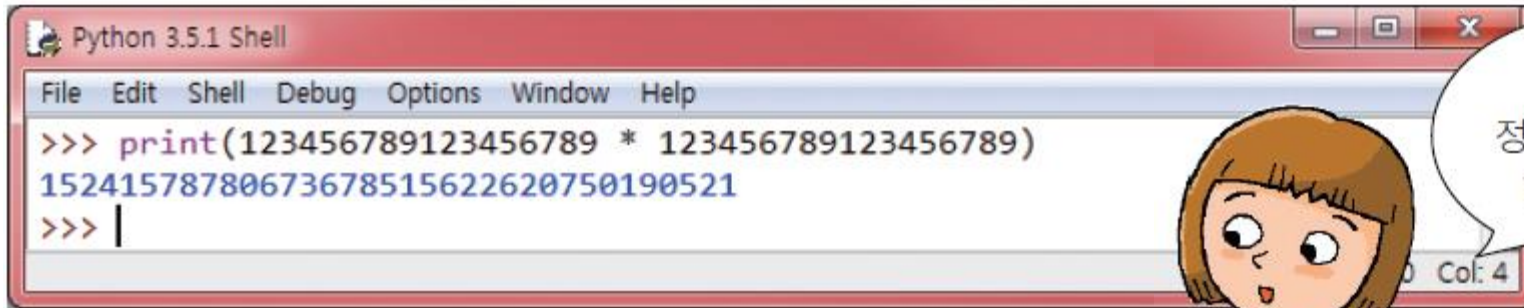


이 정도는
암산으로도 할 수
있어요! 파이썬이
필요 없다고요!

계산하기 #2



계산하기 #3



A screenshot of a Python 3.5.1 Shell window. The window has a red title bar with the text "Python 3.5.1 Shell" and standard window controls. Below the title bar is a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area shows a Python prompt ">>>" followed by the command `print(123456789123456789 * 123456789123456789)`. The output is `15241578780673678515622620750190521`. The prompt is followed by a vertical bar cursor. The status bar at the bottom right shows "Col: 4".

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
>>> print(123456789123456789 * 123456789123456789)
15241578780673678515622620750190521
>>> |
```



이것은
정말 계산기로
안 되네요!

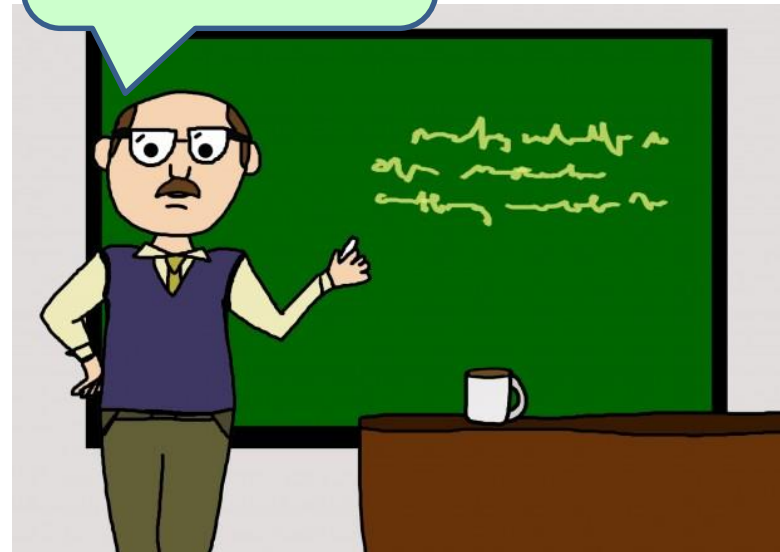
문자열 출력하기

```
>>> print("강아지" + "고양이")  
강아지고양이  
>>>
```

문자열은 어떻게 구별하나요?

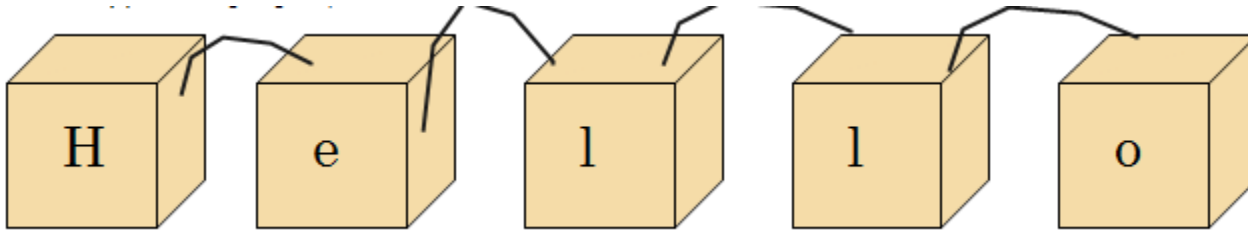


따옴표("...")가 붙으면 문자열입니다.



문자열

- 문자열(string) : 큰따옴표("...")나 작은따옴표('...') 안에 들어 있는 텍스트 데이터



- 반드시 따옴표가 있어야 한다.

```
>>> print(Hello World!)  
SyntaxError: invalid syntax
```

print() 함수

- 여러 개의 값들을 화면에 차례대로 출력할 수 있다.

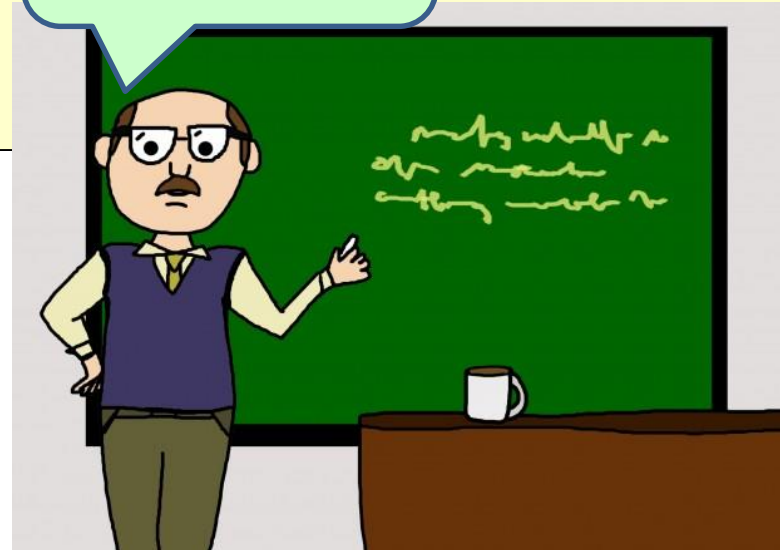
```
>>> print("결과값은", 2*7, "입니다.")  
결과값은 14 입니다.
```

문자열과 숫자

```
>>> print("100" + "200")  
100200  
>>>
```

```
>>> print(100 + 200)  
300  
>>>
```

따옴표("...")가
붙으면 문자열
입니다.

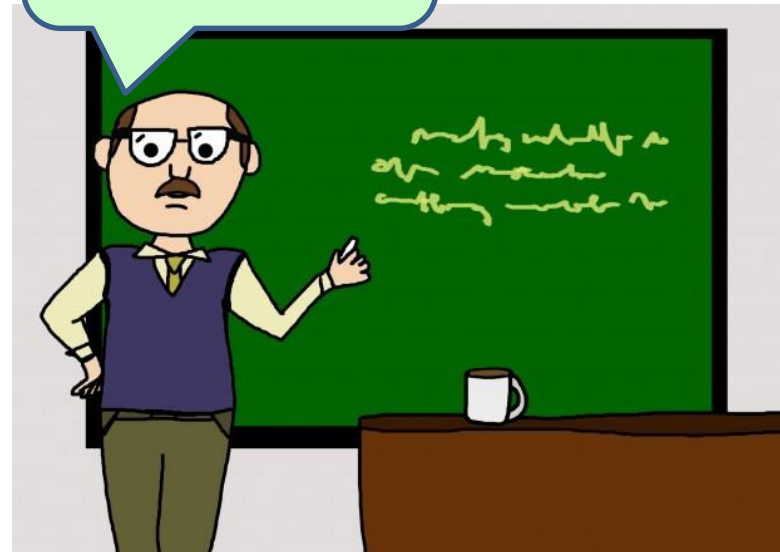


문자열 반복하기

```
>>> print("반가워요 " * 20)
```

```
반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요  
반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요 반가워요  
반가워요 반가워요 반가워요 반가워요  
>>>
```

*은 반복을 의미
하기도 합니다.



Solution

```
>>> print("안녕하세요? 여러분")  
안녕하세요? 여러분
```

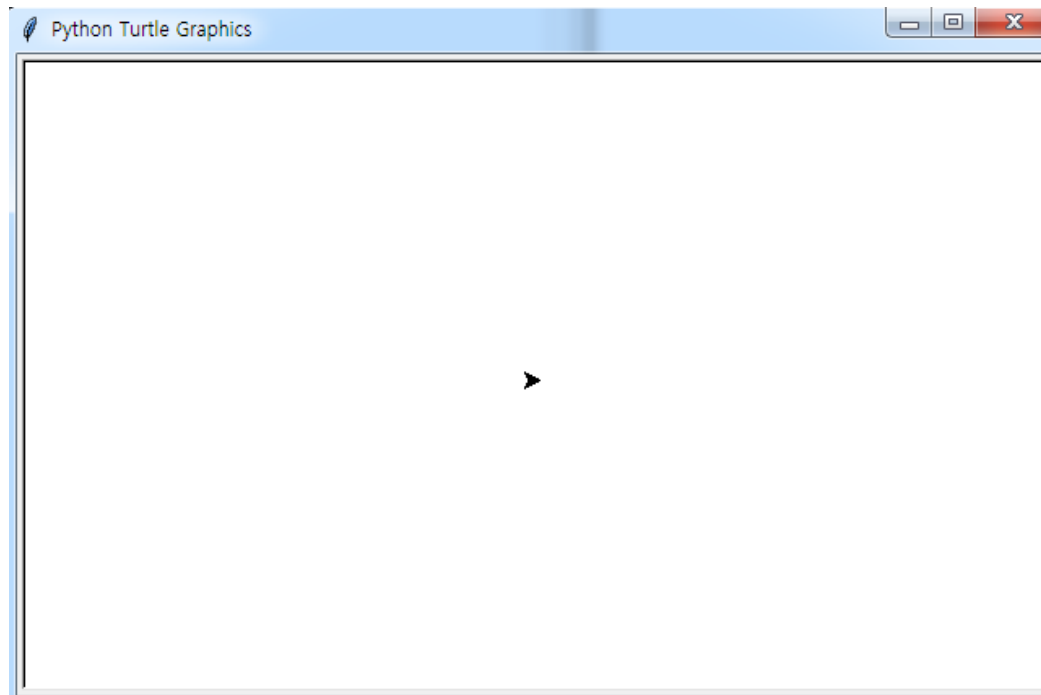
```
>>> print("저는 파이썬을 무척 좋아합니다.")  
저는 파이썬을 무척 좋아합니다.
```

```
>>> print("9*8은", 9*8, "입니다.")  
9*8은 72 입니다.
```

```
>>> print("안녕히 계세요.")  
안녕히 계세요.
```

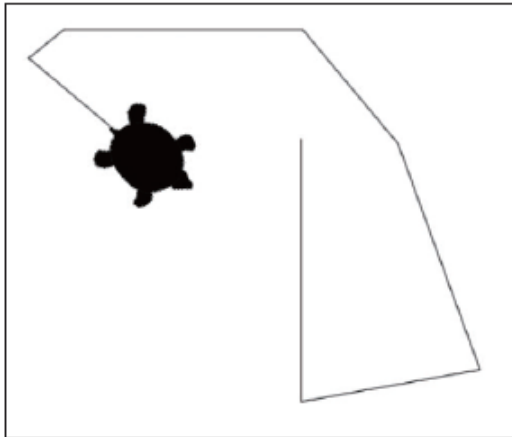
turtle

- 파이썬 내장 그래픽 모듈
- 1960년대 Logo라는 교육용 프로그래밍 언어의 일부로 개발된 컴퓨터 그래픽 방식
- <https://docs.python.org/3/library/turtle.html>



터틀 그래픽

- 터틀 그래픽은 화면에서 거북이를 이용하여서 그림을 그리는 기능이다.



터틀 그래픽은
거북이가 화면 위에서
지나가면서 그림을
그리는 것이라고
생각하세요.



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)
```



책과 강의노트에 나오는 소스에는 인코딩에 관한 내용은 생략되어 있습니다.
한글인코딩 설정을 위해서는 다음과 같은 주석을 달아 사용합니다.

Mac OS

```
# -*- coding: utf-8 -*-
```

Windows

```
# -*- coding: cp949 -*-  
# -*- coding: euc-kr -*-
```

■ 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)



여기서 잠깐

변수의 선언

C/C++, 자바 등 컴파일러 언어에서 변수를 사용하려면 반드시 선언부터 해야 한다. 반면 자바스크립트, 파이썬 등 인터프리터 언어에서는 변수를 선언하지 않고 바로 사용해도 된다. Code02-01.py의 1행 `a=100`처럼 `a`를 선언하지 않고 바로 `a`에 100을 대입하는 방식이다. 그러면 대입하는 순간에 `a`라는 변수가 자동 선언되고 생성되는 것이다. 정수 100을 대입하기 때문에 `a`는 자동으로 정수형 변수로 선언되는 것이다. 이를 C나 자바로 만든다면 다음과 같이 선언 후에 값을 대입해야 한다.

```
int a, b;      ## 두 변수를 선언한다. ##
a = 100;      ## a에 100을 대입한다. ##
b = 50 ;      ## b에 50을 대입한다. ##
```

파이썬은 변수 선언이 꼭 필요하지는 않지만, 가급적 변수에 초깃값을 미리 대입시켜서 변수 선언 부분에 준비해 놓는 것이 바람직하다는 정도로 이해하면 좋겠다.

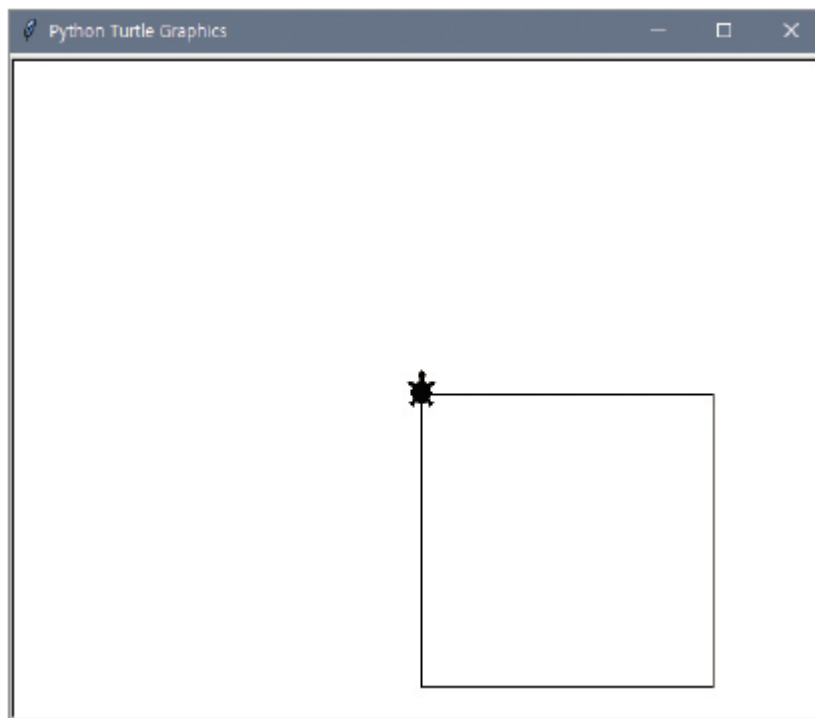
Section05 터틀 그래픽 프로그램 작성

■ 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()
```



- 파이썬의 turtle 모듈은 펜을 움직이고, 펜의 크기를 설정하고, 펜을 들어올리고 내려놓기 위한 메소드를 포함한다



Turtle의 펜 그리기 상태 메소드 파이썬

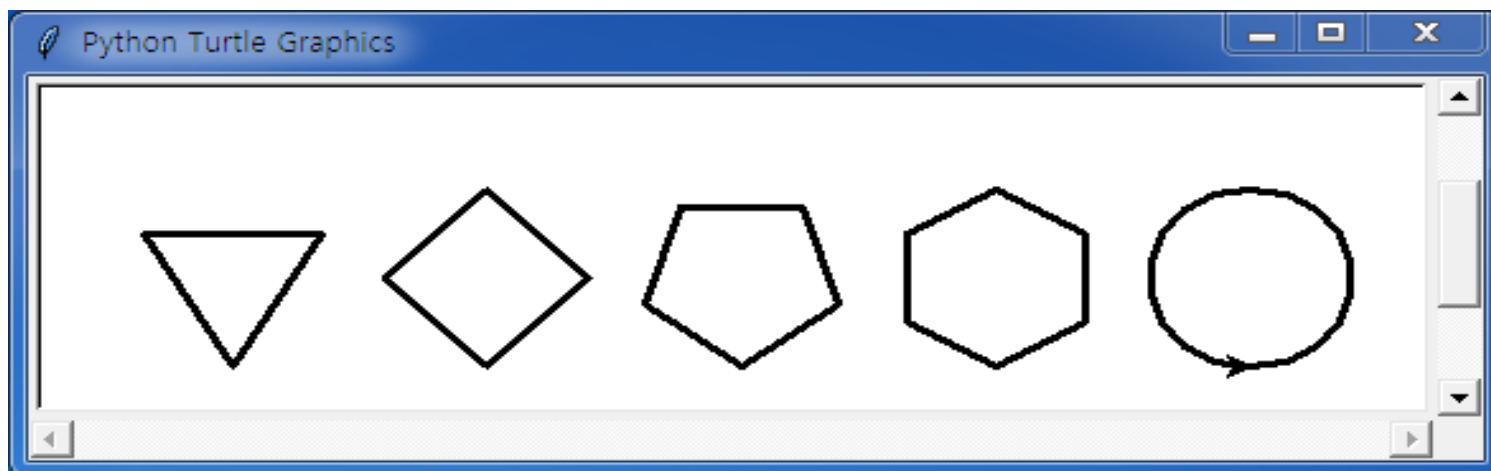
| 메소드 명 | 설명 |
|-------------------------------|----------------------------|
| <code>turtle.pendown()</code> | 펜을 내려놓는다 - 움직일 때 그려진다. |
| <code>turtle.penup()</code> | 펜을 들어올린다 - 움직일 때 그려지지 않는다. |
| <code>turtle.pensize()</code> | 선의 두께를 특정 너비로 설정한다. |



| 메소드 명 | 설명 |
|--|---|
| <code>turtle.forward(d)</code> | Turtle이 향하고 있는 방향으로 특정 거리 d만큼 앞으로 이동시킨다. |
| <code>turtle.backward(d)</code> | Turtle이 향하고 있는 반대 방향으로 특정 거리 d만큼 뒤로 이동시킨다. Turtle의 방향은 바뀌지 않는다. |
| <code>turtle.right(angle)</code> | Turtle을 특정 각 만큼 오른쪽으로 회전시킨다. |
| <code>turtle.left(angle)</code> | Turtle을 특정 각 만큼 왼쪽으로 회전시킨다. |
| <code>turtle.goto(x, y)</code> | Turtle을 절대 위치 (x, y)로 옮긴다. |
| <code>turtle.setx(x)</code> | Turtle의 x 좌표를 특정 위치로 옮긴다. |
| <code>turtle.sety(y)</code> | Turtle의 y 좌표를 특정 위치로 옮긴다. |
| <code>turtle.setheading(angle)</code> | 특정 각도로 Turtle의 방향을 설정한다. 0-동쪽, 90-북쪽, 180-서쪽, 270-남쪽. |
| <code>turtle.home()</code> | Turtle을 원점 (0, 0)으로 옮기고 동쪽 방향으로 설정한다. |
| <code>turtle.circle(r, ext, step)</code> | 특정 반지름 r, 경계 ext와 단계 step인 원을 그린다. |
| <code>turtle.dot(diameter, color)</code> | 특정 지름 diameter와 색상 color인 원을 그린다. |
| <code>turtle.undo()</code> | Turtle의 마지막 명령을 (반복적으로)되돌린다. |
| <code>turtle.speed(s)</code> | 1부터 10사이의 정수(10이 최대) s로 Turtle의 속도를 설정한다. |



사례 연구: 간단한 도형 그리기 파이썬





색 및 폰트 지정하기

파이썬

| 메소드명 | 설명 |
|---|--|
| <code>turtle.color(c)</code> | 펜 색상을 c로 설정한다. |
| <code>turtle.fillcolor(c)</code> | 펜 채움 색상을 c로 설정한다. |
| <code>turtle.begin_fill()</code> | 도형을 채우기 전에 이 메소드를 호출한다. |
| <code>turtle.end_fill()</code> | <code>begin_fill</code> 에 대한 마지막 호출 전까지 그려진 도형을 채운다. |
| <code>turtle.filling()</code> | 채우기 상태를 반환한다. 채우기 상태이면 True, 그렇지 않으면 False |
| <code>turtle.clear()</code> | 창을 깨끗하게 지운다. turtle의 상태와 위치를 영향을 받지 않는다. |
| <code>turtle.reset()</code> | 창을 깨끗하게 지우고 turtle의 상태와 위치를 원래 기본값으로 재설정한다. |
| <code>turtle.screensize(w, h)</code> | 캔버스의 높이 w와 높이 h를 설정한다. |
| <code>turtle.hideturtle()</code> | turtle을 보이지 않게 만든다. |
| <code>turtle.showturtle()</code> | turtle을 보이게 만든다. |
| <code>turtle.isvisible()</code> | turtle이 보이면 True를 반환한다. |
| <code>turtle.write(s, font=("Arial", 8, "normal"))</code> | 현재 turtle의 위치에 문자열 s를 쓴다. 폰트는 폰트명, 폰트크기, 폰트유형의 세 값으로 구성된다. |



색상 및 채우기를 사용하여 그리기 파이썬

- turtle 객체는 색상과 폰트를 설정하는 메소드를 포함한다.



■ 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()
```



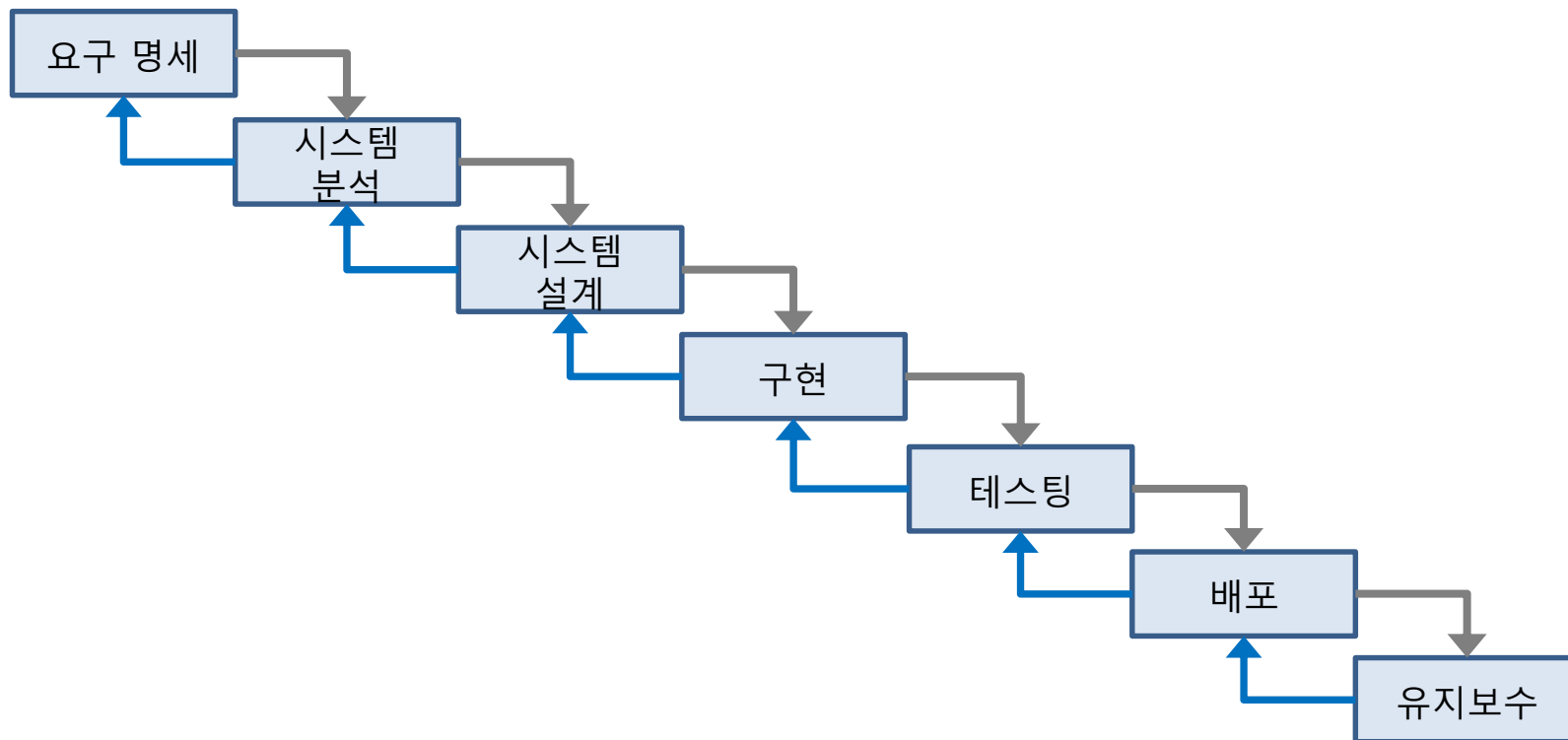
Section05 터틀 그래픽 프로그램 작성

표 2-1 Code02-07.py에 사용된 기타 함수

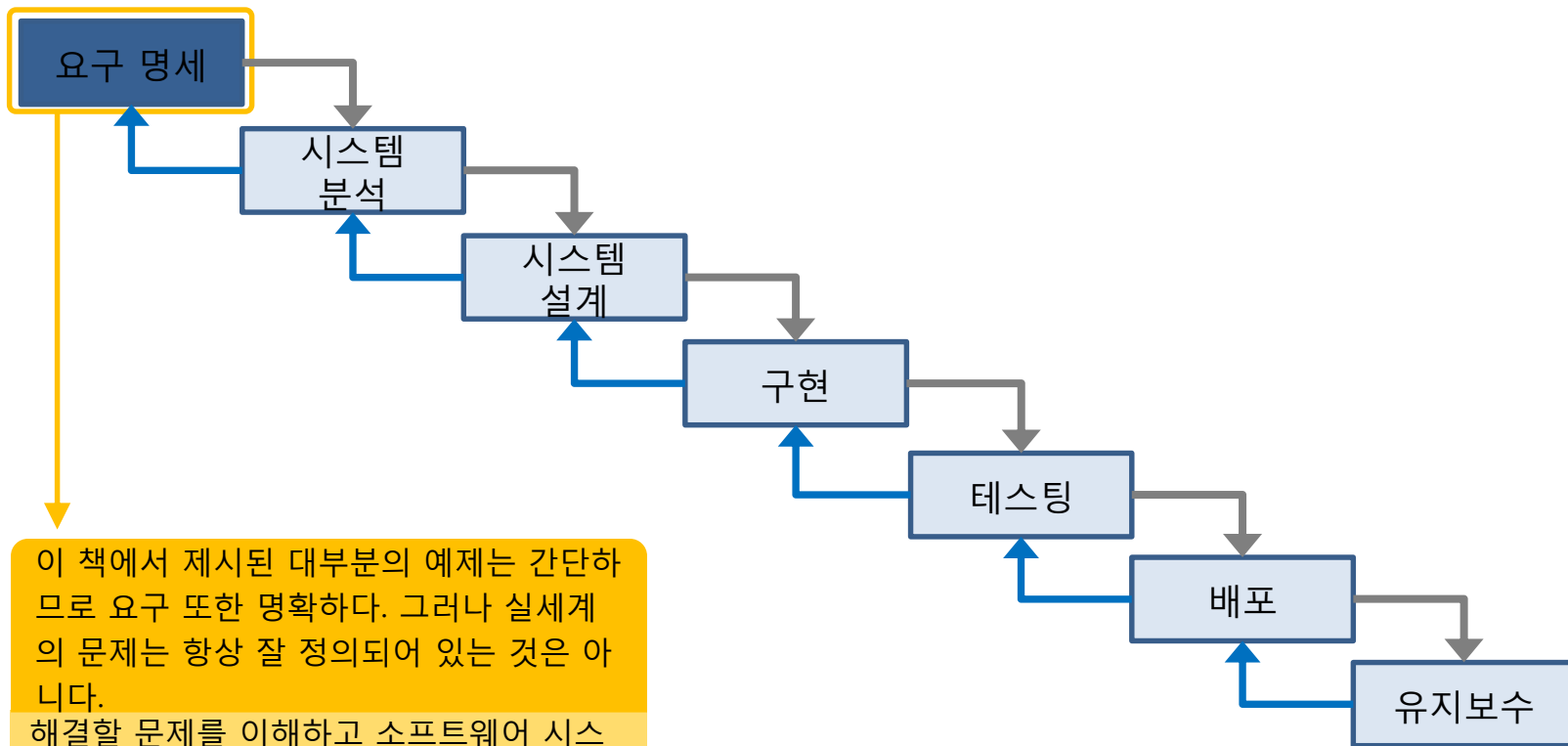
| 함수 | 설명 |
|--|--|
| <code>turtle.title(제목)</code> | 윈도창의 제목을 설정한다. |
| <code>turtle.pensize(펜 두께)</code> | 그릴 선의 두께를 설정한다. |
| <code>turtle.onscreenclick(함수명, 번호)</code> | 윈도창을 마우스로 클릭하면 '함수명' 함수가 작동한다. 1은 마우스 왼쪽 버튼, 2는 마우스 가운데 버튼, 3은 마우스 오른쪽 버튼을 지정한다. |

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.



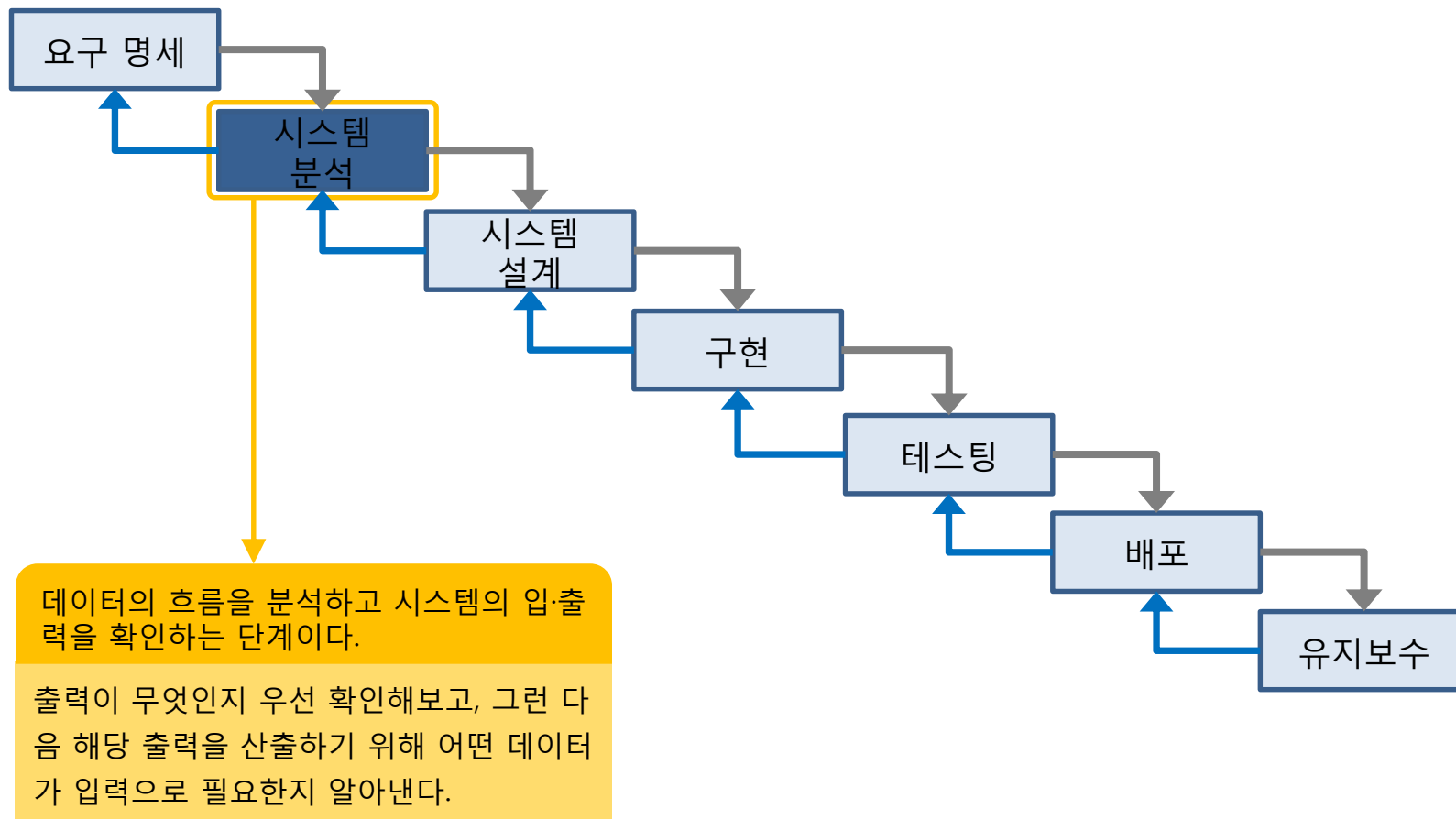
- 소프트웨어 개발 과정



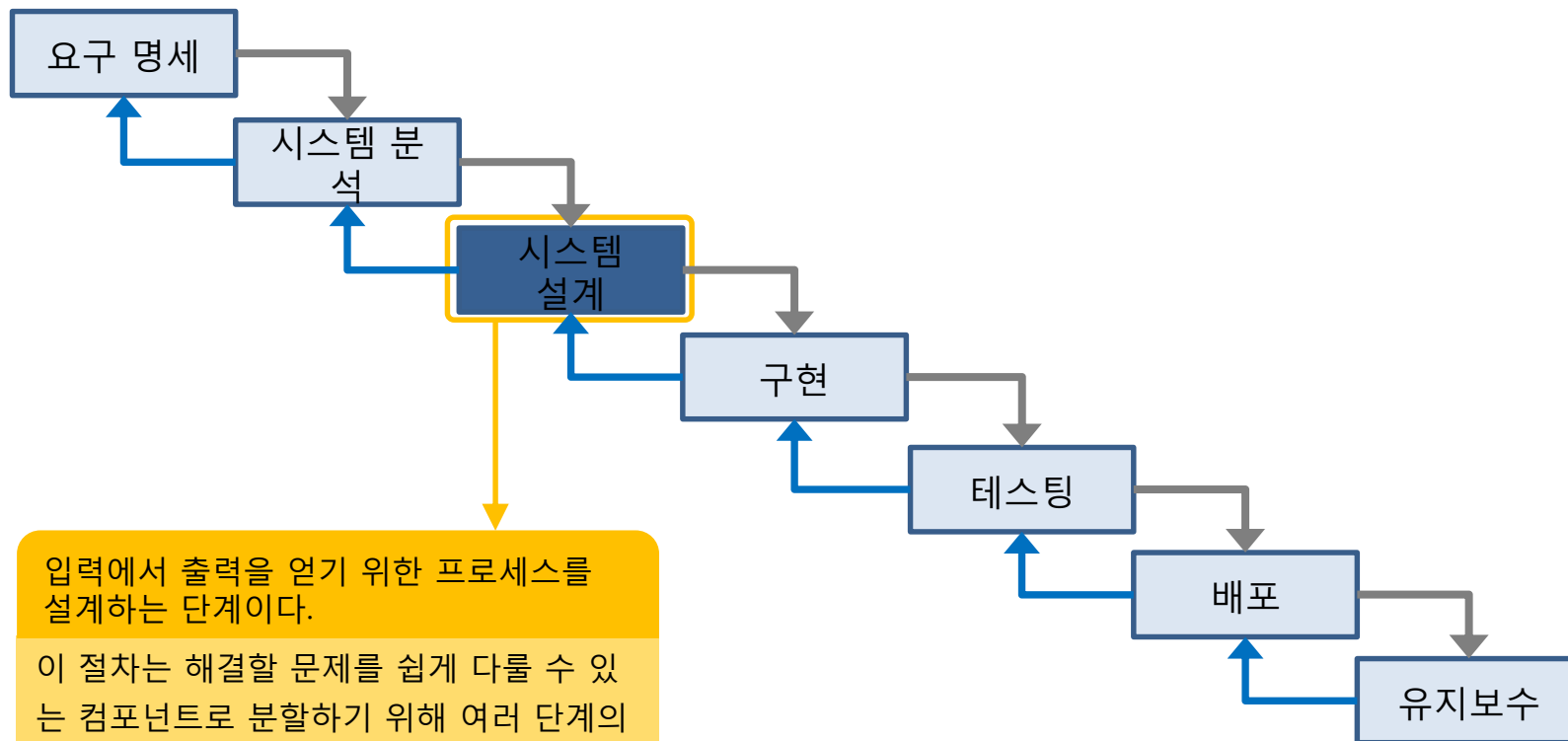
이 책에서 제시된 대부분의 예제는 간단하므로 요구 또한 명확하다. 그러나 실세계의 문제는 항상 잘 정의되어 있는 것은 아니다.

해결할 문제를 이해하고 소프트웨어 시스템이 무엇을 해야 하는지를 상세히 문서화하기 위한 정형적 절차이다. 이 절차는 사용자와 개발자와의 밀접한 상호작용을 포함한다.

- 소프트웨어 개발 과정



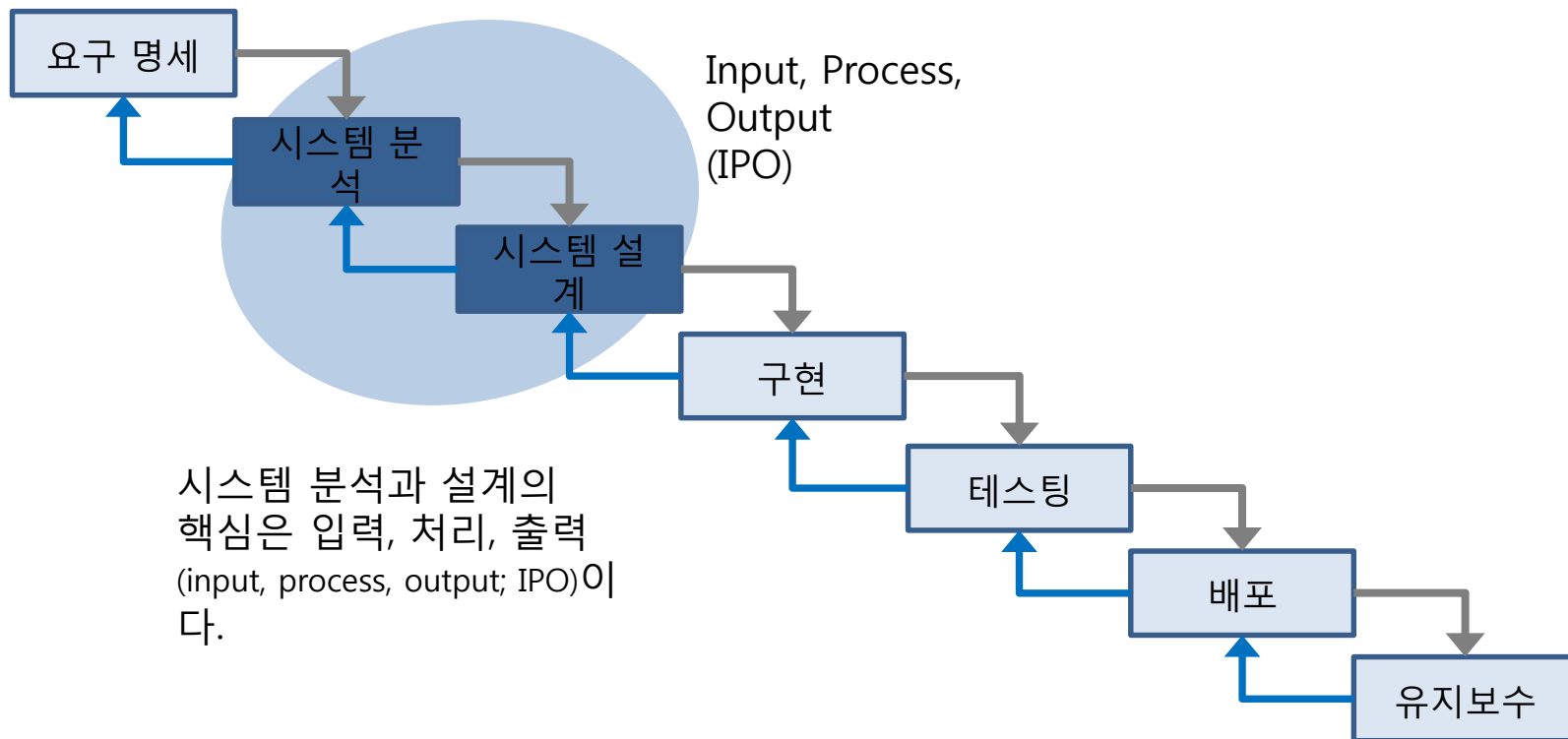
- 소프트웨어 개발 과정



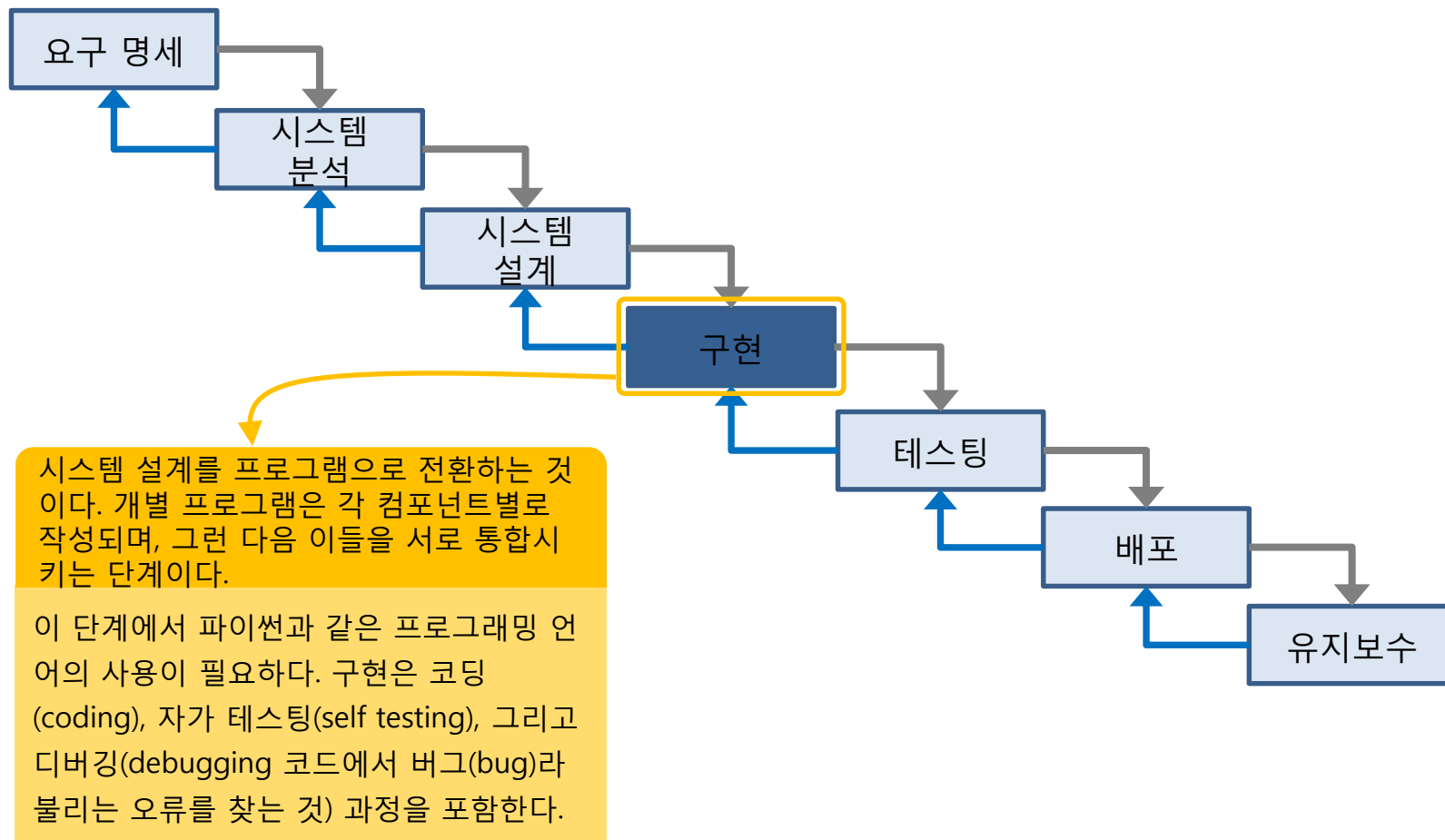
입력에서 출력을 얻기 위한 프로세스를 설계하는 단계이다.

이 절차는 해결할 문제를 쉽게 다룰 수 있는 컴포넌트로 분할하기 위해 여러 단계의 추상화를 사용하고 각 컴포넌트를 구현하기 위한 설계 전략을 포함하고 있다. 컴포넌트는 시스템의 특정 기능을 수행하는 서브시스템으로 생각할 수 있다.

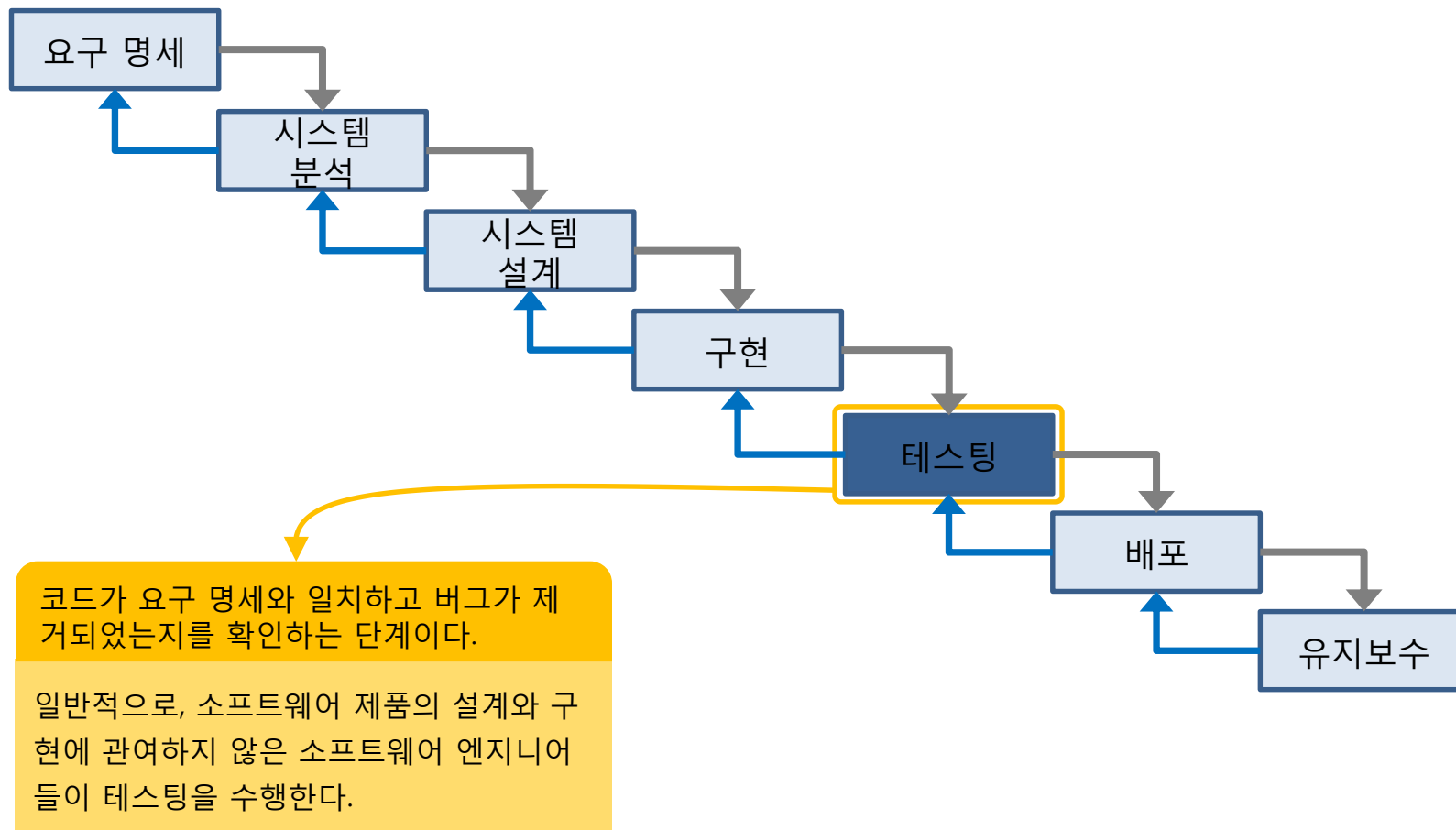
- 소프트웨어 개발 과정



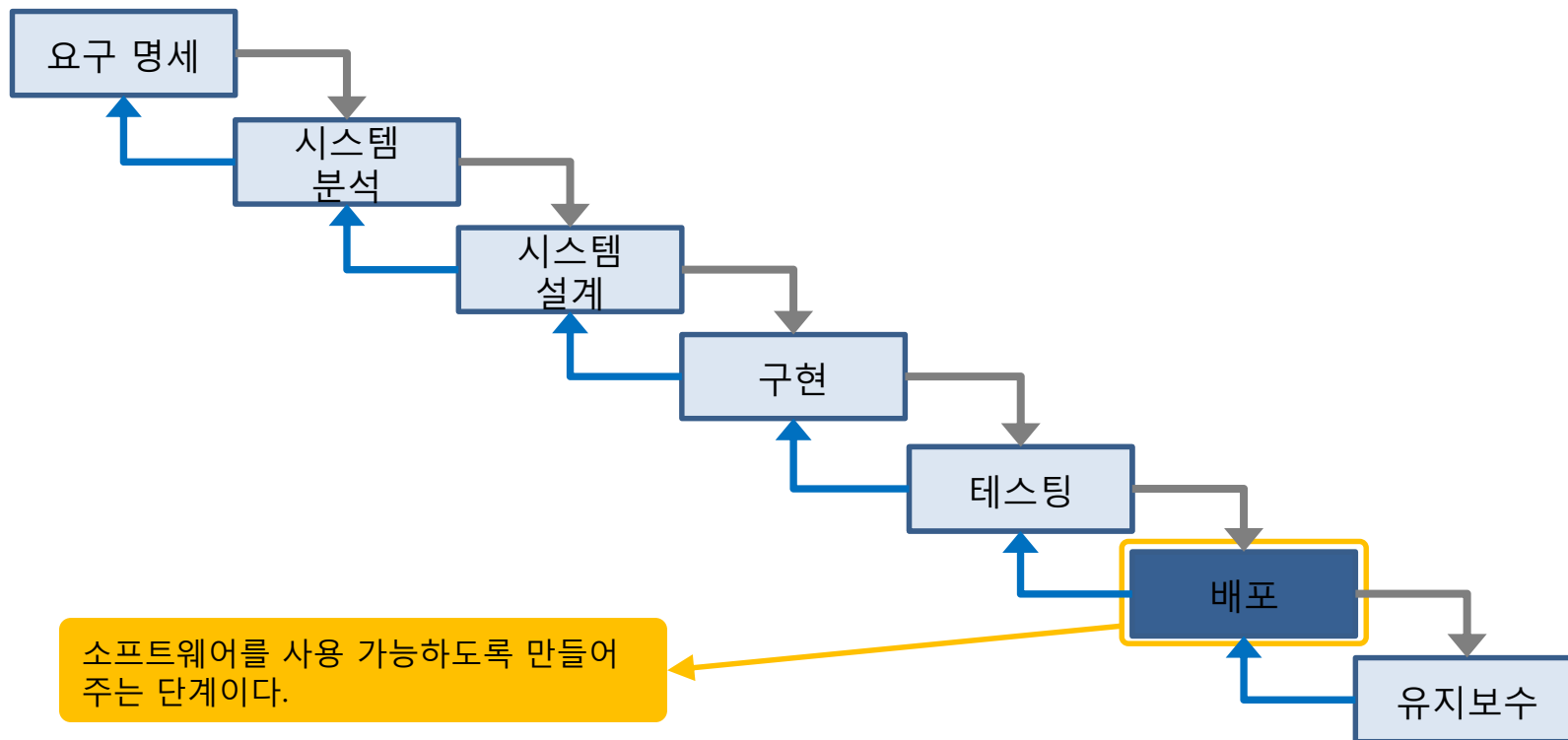
- 소프트웨어 개발 과정



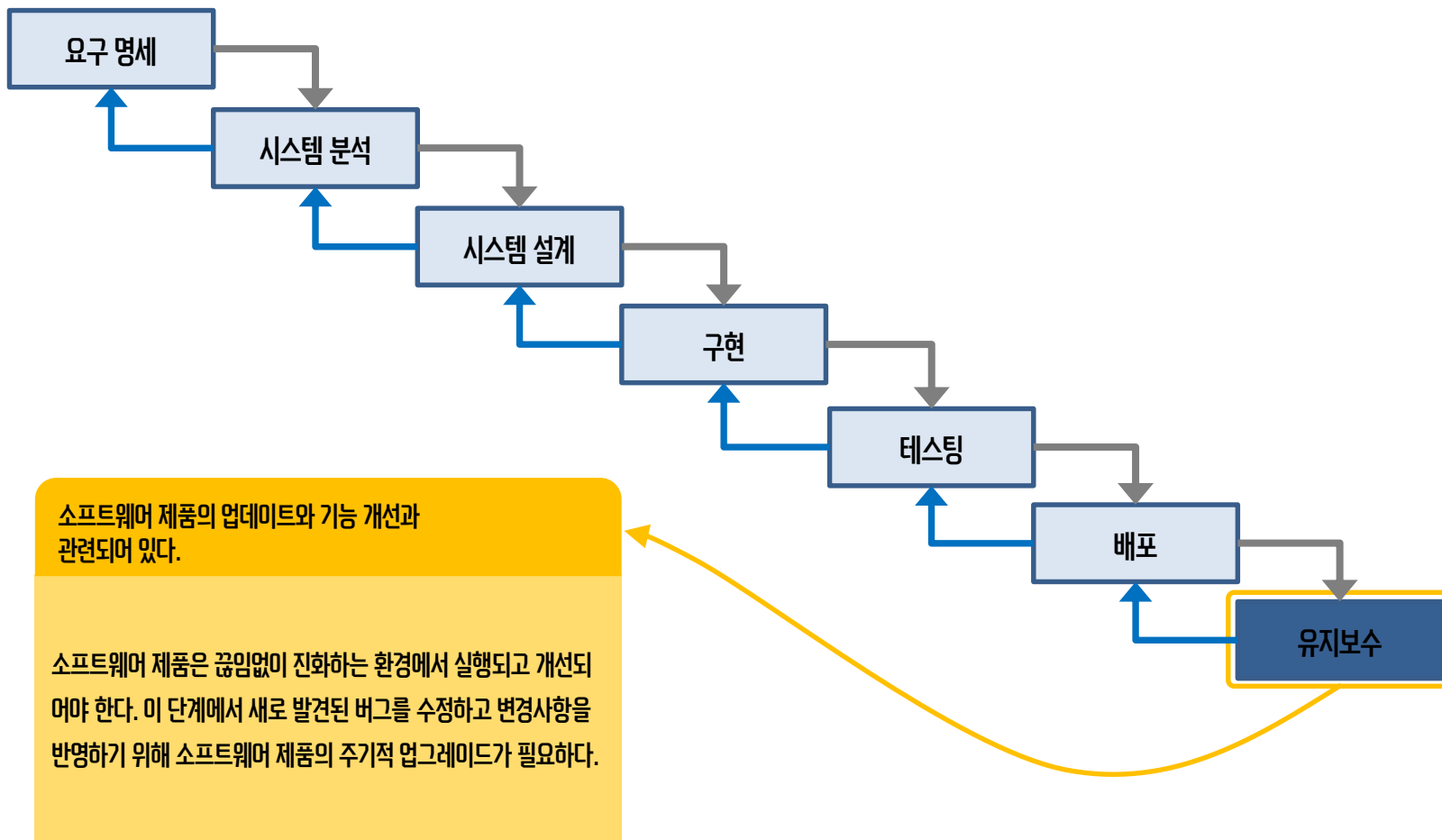
- 소프트웨어 개발 과정



- 소프트웨어 개발 과정



- 소프트웨어 개발 과정





Thank You
