return 문은:

1. 함수 호출 부분(function call)을 **return** 문 뒤에 오는 값으로 대체합니다.
2. 현재 함수의 실행을 멈추고 함수가 호출된 지점으로 돌아가서 진행됩니다.

```
1. def f(x):
2.     return x + 1
3.
4. def g(x):
5.     return x * x - 1
6.
7. print(f(2))          # print(3)과 같음
8. print(g(3))          # print(8)과 같음
9. print(f(2) + g(3))   # print(3 + 8)과 같음
```

```
3
8
11
```

위의 코드를 해석해보겠습니다.

1. 우선 **(1)** ~ **(2)** , **(4)** ~ **(5)** 번 줄은 함수 정의니까 건너뜁니다.
2. **(7)** 번 줄에서 **f(2)** 라는 함수 호출이 있습니다. 파이썬은 **f** 라는 함수가 정의된 **(1)** 번 줄로 이동한 후 함수의 바디를 실행합니다.
3. 함수의 바디인 **(2)** 번 줄은 **return x + 1** 인데요. 이에 따라 **(7)** 번 줄의 **f(2)** 부분은 **2 + 1** 로 대체됩니다. 결과적으로 **print(f(2))** 는 **print(3)** 과 같기 때문에, **3** 이 출력됩니다.
4. **(8)** 번 줄에서 **g(3)** 이라는 함수 호출이 있습니다. 파이썬은 **g** 라는 함수가 정의된 **(4)** 번 줄로 이동한 후 함수의 바디를 실행합니다.
5. 함수의 바디인 **(5)** 번 줄은 **return x * x - 1** 인데요. 이에 따라 **(8)** 번 줄의 **g(3)** 는 **3 * 3 - 1** 로 대체됩니다. 결과적으로 **print(g(3))** 는 **print(8)** 과 같기 때문에, **8** 이 출력됩니다.
6. 마지막으로 **(9)** 번 줄에서 **f(2) + g(3)** 이라는 함수 호출이 있습니다. **f(2)** 는 **3** 으로 대체되고 **g(3)** 은 **8** 로 대체되어서, 두 수의 합 **11** 이 출력되고 프로그램은 종료됩니다.

잘 이해되셨나요? 여태까지는 **return** 문의 첫번째 역할에 집중하여 설명했기 때문에, 이번에는 두번째 역할을 더 명확히 설명하기 위해 코드를 조금 더 추가해보겠습니다.

```
def f(x):
    print("f 시작")
    return x + 1
    print("f 끝")

def g(x):
    print("g 시작")
    return x * x - 1
    print("g 끝")

print(f(2))          # print(3)과 같음
print(g(3))          # print(8)과 같음
print(f(2) + g(3))   # print(3 + 8)과 같음
```

```
f 시작
3
g 시작
8
f 시작
g 시작
11
```

f 와 **g** 함수를 정의할 때 **print** 문 몇 개를 추가로 써넣었습니다. 달라진 부분을 중점적으로 설명해보겠습니다.

먼저 **f(2)** 라는 함수 호출로 인해 파이썬은 **f** 함수가 정의된 **(1)** 번 줄로 이동됩니다. 함수 바디 부분에서 **"f 시작"** 이 출력된 후 **3** 이 리턴됩니다. 이로써 **print(f(2))** 는 **print(3)** 과 동일해져서 **3** 이 출력됩니다.

하지만 **"f 끝"** 이라는 문자열은 출력되지 않았습니다. 왜일까요? **return** 문의 두 번째 역할에 따라, 현재 함수의 실행이 멈추어지고 함수 호출 부분으로 돌아가기 때문입니다. 즉 **return x + 1** 이 실행되고 나서, **f** 함수의 실행은 중단되고 함수가 불려온 **print(f(2))** 의 뒷부분부터 파이썬이 다시 진행되기 때문에, **print("f 끝")** 이 절대로 실행될 수 없었던 것입니다. **print("f 끝")** 과 같이 절대 실행될 수 없는 코드를 **dead code**라고 부릅니다.

이와 같은 논리로 **(9)** 번 줄의 **print("g 끝")** 도 dead code라고 할 수 있습니다.

return vs. print

많은 분들이 **return** 문과 **print** 문을 헷갈려합니다. 한번 비교해보겠습니다.

사례 1

```
def print_square(x):
    print(x * x)

def get_square(x):
    return x * x

print_square(3)
print("--")
get_square(3)
print("--")
print(get_square(3))
print("--")
print(print_square(3))
```

```
9
--
--
9
--
9
None
```

print_square 함수는 파라미터의 제곱을 출력해줍니다. 리턴값이 지정되어있지 않으므로, 이 함수의 리턴값은 **None** 입니다.

반면 **get_square** 함수는 파라미터의 제곱을 리턴시켜줍니다. 출력값은 없습니다.

1. **print_square(3)** 이라는 코드를 실행하면 **print(x * x)** 에 의해 **9** 라는 값이 출력됩니다.
2. **get_square(3)** 은 **return** 문에 의해 **9** 라는 값으로 대체됩니다. 그러나 대체된 값을 출력하라는 명령이 없기 때문에, 아무것도 출력되지 않습니다.
3. **print(get_square(3))** 는 **get_square(3)** 함수의 **return** 문에 의해 대체된 값을 출력하는 명령입니다. **9** 가 리턴되므로, **9** 가 출력됩니다.
4. **print(print_square(3))** 의 경우 우선 **print_square** 함수에 있는 **print(x * x)** 에 의해 **9** 가 출력됩니다. 그리고 함수 호출 부분은 함수의 리턴값으로 대체되는데, **print_square** 함수의 리턴값은 **None** 입니다. 따라서 **None** 도 함께 출력됩니다.

사례 2

```
1. def secret_number():
2.     print("우리의 비밀 번호는: ")
3.     return 486
4.
5. print(secret_number())
```

```
우리의 비밀 번호는:
486
```

1. 파이썬은 **(1)** ~ **(3)** 번 줄의 함수 정의를 건너뛰고, **(5)** 번 줄에 도착합니다.

2. **(5)** 번 줄에서 **secret_number()** 라는 함수 호출 때문에, 파이썬은 **secret_number** 함수가 정의되어 있는 **(1)** 번 줄로 이동합니다.
3. **(2)** 번 줄의 **print** 명령에 따라 파이썬은 **"우리의 비밀 번호는: "** 을 출력합니다.
4. **486** 을 리턴하라는 명령이 있기 때문에
    1. 함수의 실행이 중단되고,
    2. 함수를 호출한 **secret_number()** 라는 부분이 **486** 으로 대체됩니다.
5. **(5)** 번 줄의 **print(secret_number())** 는 **print(486)** 과 동일해져서 **486** 이 출력됩니다.

사례 3

만일 위 코드에서 **(5)** 번 줄이 **print(secret_number())** 가 아닌, **secret_number()** 였으면 어땠을까요?

```
1. def secret_number():
2.     print("우리의 비밀 번호는: ")
3.     return 486
4.
5. secret_number()
```

```
우리의 비밀 번호는:
```

왜 **"우리의 비밀 번호는: "** 만 출력되고, **486** 은 출력되지 않은 것일까요?

486 을 리턴하라는 **(3)** 번 줄의 명령으로 인해, 함수를 호출한 **secret_number()** 라는 부분이 **486** 으로 대체됩니다. 그러나 **(5)** 번 줄에서 대체된 값을 출력하라는 명령은 없기 때문에, **486** 이 출력되지 않는 것입니다.

사례 4

그렇다면 함수 내부에서 **print** 문과 **return** 문의 순서가 바뀌어 있으면 어떻게 될까요?

```
1. def secret_number():
2.     return 486
3.     print("우리의 비밀 번호는: ")
4.
5. print(secret_number())
```

```
486
```

왜 **486** 만 출력되고, **"우리의 비밀 번호는: "** 은 출력되지 않은 것일까요?

파이썬이 **(2)** 번 줄에서 **return** 문에 도달하면, **486** 이 리턴되고 나서 함수의 실행은 곧바로 종료됩니다. **(3)** 번 줄이 dead code여서 **"우리의 비밀 번호는: "** 이 출력되지 않은 것입니다.

✔ 수업을 완료하셨으면 체크해주세요.

수강생 Q&A 보기 (/questions?assignment_id=48&sort_by=popular)

질문하기 (/questions/new?assignment_id=48&op1=%ED%94%84%EB%A1%9C%EA%B7%B

이전 강의
return과 print 차이 (/assignments/47)

다음 강의
global변수와 local 변수 (/assignments/49)