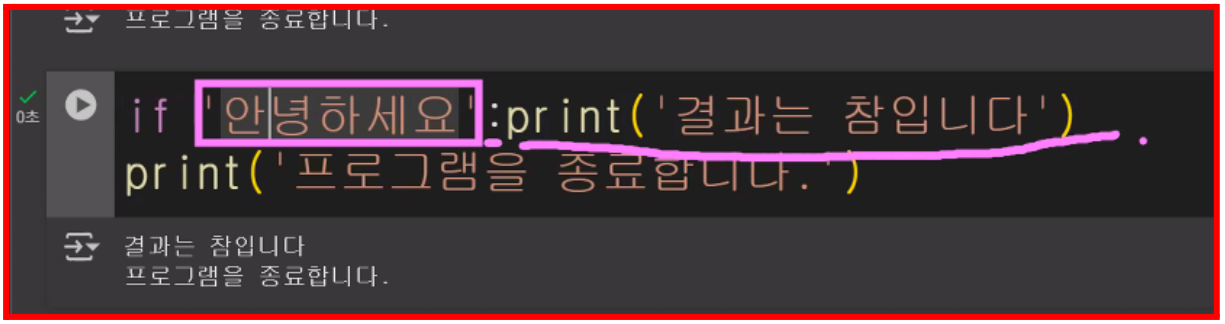
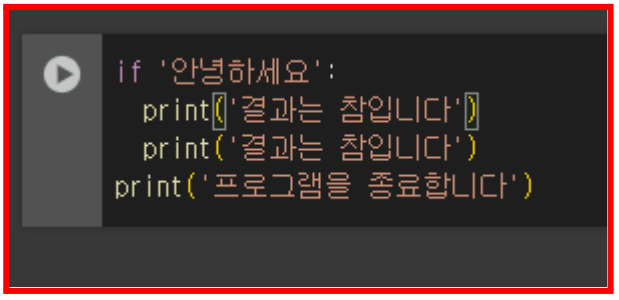
Day 6 - Control Statement (Conditionals)

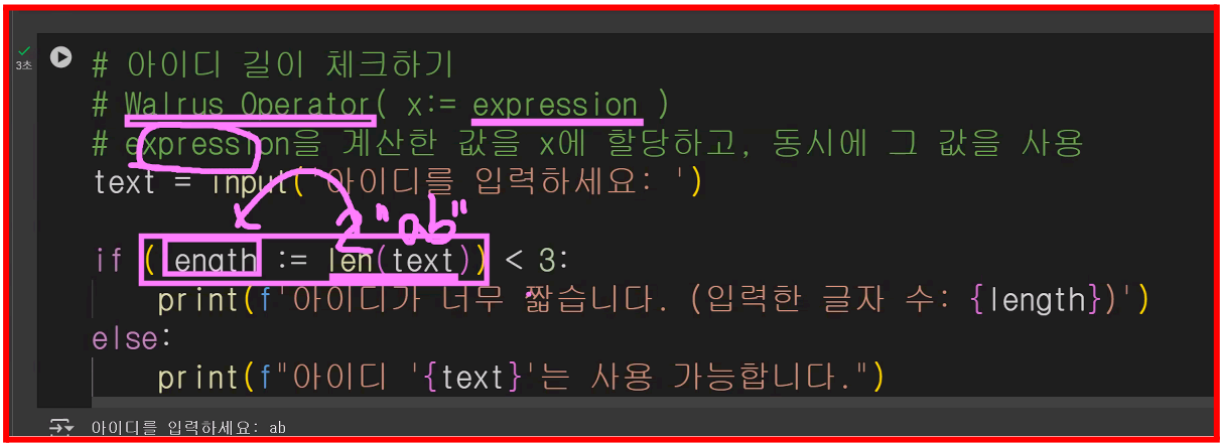


- if 조건이 True 면 ":" 뒤의 한줄을 출력한다.
- 만약 블록을 넣어놓으면 묶어서 한꺼번에 실행된다



  - 이 경우 블록 안에 있는 내용이 묶어서 실행된다.

Walrus Operator



- len(text) 값을 계산해서 length에 할당
- 3과 비교
- True or False 계산해서 도출

Conditional Statement:

```
조건이 참인 경우 값 if condition else 조건이 거짓인 경우 값
```

```
[23] num = int(input('숫자를 입력하세요: '))
     if num % 2 == 0:
         print('짝수')
     else:
         print('홀수')
```

```
숫자를 입력하세요: 13
홀수
```

```
# 조건부 표현식
num = int(input('숫자를 입력하세요: '))
print('짝수') if num % 2 == 0 else print('홀수')
```

```
숫자를 입력하세요: 13
홀수
```

- (조건이 참인 경우 값) if condition else (조건이 거짓인 경우 값)

Structural Pattern Matching (구조적 패턴 매칭)

```
match 값:
    case 패턴1:
        실행할 코드1
    case 패턴2:
        실행할 코드2
    case _:
        기본 실행 코드 (default)
```

- elif, if, else 구문이 범위에 최적화 되어있다고 하면
- match-case는 1:1 범위에 최적화 되어있다.

ex) Structural Pattern Matching:

```
month = int(input('월을 입력하세요 (1~12): '))

match month:
    case 1 | 3 | 5 | 7 | 8 | 10 | 12:
        print(f"{month}월은 31일까지 있습니다.")
    case 4 | 6 | 9 | 11:
        print(f"{month}월은 30일까지 있습니다.")
    case 2:
        print('2월은 28일 또는 윤년이면 29일까지 있습니다.')
    case _: # 이 케이스에 모두 해당이 안될경우
        print("잘못된 월 입니다. 1~12 사이의 숫자를 입력해주세요")
```

```
월을 입력하세요 (1~12): 2
2월은 28일 또는 윤년이면 29일까지 있습니다.
```

- case _:
  - 모든 케이스에 해당이 안 될경우 작동한다.
  - if 문의 else 같은 존재이다.

Structural Pattern Matching + Conditional Statement1:

```
# user = ('김사과', 20)
user = ('반하나', 13)

match user:
    case (name, age) if age > 19:
        print(f'{name}님은 성인입니다.')
    case (name, age) if age > 15:
        print(f'{name}님은 청소년입니다.')
    case (name, age) if age > 6:
        print(f'{name}님은 어린이입니다.')
    case _:
        print(f'{name}님은 유아입니다.')
```

```
반하나님은 어린이입니다.
```

- 이런식으로 섞어서 사용가능하다.
- (name, age) 튜플로 저장해서 age > 19와 비교한뒤
- True / False 받은 이후 출력 또는 다른 케이스로 간다.

Structural Pattern Matching + Conditional Statement2:

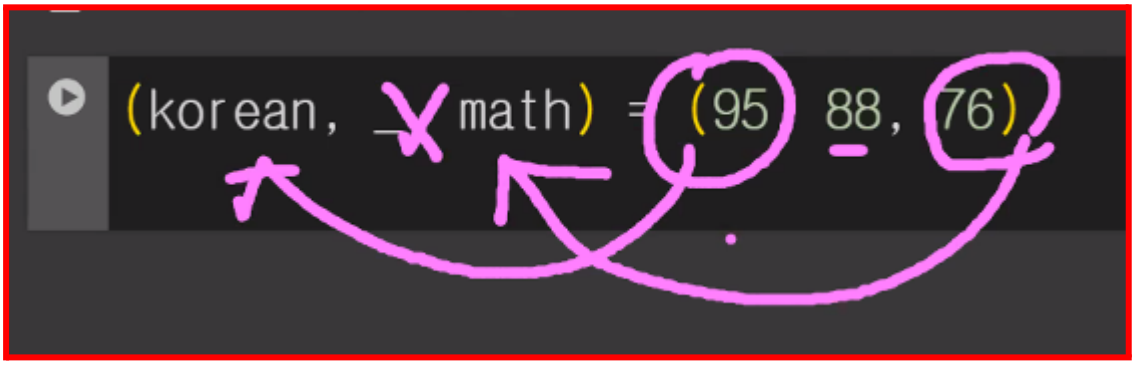
```
scores = [95, 88, 76]
# scores = (95, 88, 76)
# scores = {"국어":95, "영어":88, "수학":76}

match scores:
    case [korean, english, math]:
        print(f"1. 국어: {korean}, 영어: {english}, 수학: {math}")
    case (korean, _, math):
        print(f"2. 국어: {korean}, 수학: {math}")
    case {"국어": korean, "영어": english, "수학": math}:
        print(f"3. 국어: {korean}, 영어: {english}, 수학: {math}")
```

```
1. 국어: 95, 영어: 88, 수학: 76
```

- case를 이용해 같은 데이터타입인경우만 작동하게 만들 수 있다.
- 위의 경우 1번 케이스 (리스트타입) 만 작동한다.
- 그러나 리스트와 튜플은 같은 타입으로 인식한다.

'_' 사용



- 이런식으로 언더바 _ 로 채우면 변수가 만들어지지 않는다.
- 메모리에 안올라간다.