

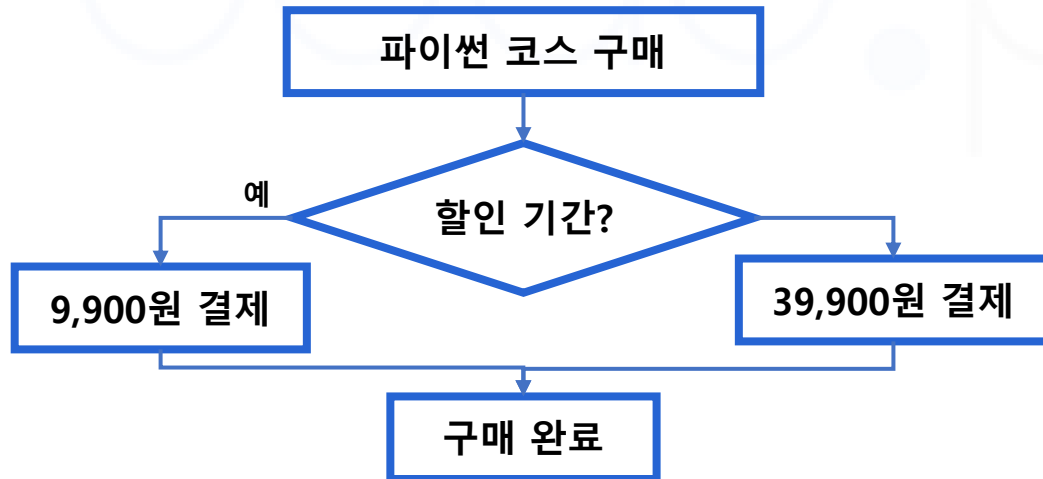
# 조건문 소개

## (Conditional Statement)

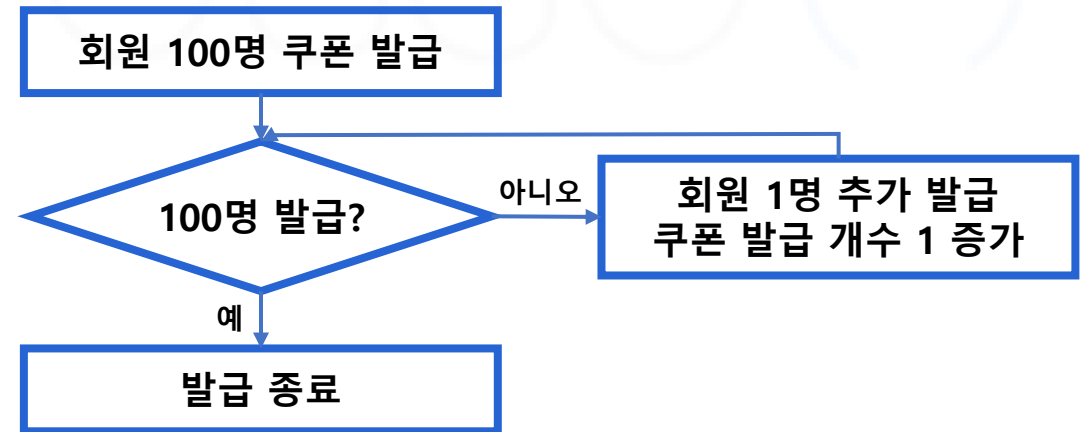
# | 제어문(Control Flow Statement)

- 파이썬 프로그램은 기본적으로 위에서 아래로 순차적으로 실행 됨
- 프로그램의 실행 순서를 제어 가능
  - 조건에 따라 실행 코드를 분기 - 조건문(Conditional Statement)
  - 특정 코드를 반복해서 실행 - 반복문(Loop Statement)

조건문



반복문



# | 조건문(Conditional Statement)

- 조건에 따라 서로 다른 코드가 실행 되도록 분기 처리
- 홀수, 짝수 판별 프로그램
  - 주어진 숫자를 2로 나눈 나머지가 0이면 짝수
  - 주어진 숫자를 2로 나눈 나머지가 0이면 홀수
- 티켓 발급 프로그램(가격은 1,000원이고 미성년자는 20% 할인)
  - 나이가 만 19세 이상이면 → 1,000원 결제
  - 그 외의 경우에는(나이가 만 19세 미만이면) → 800원 결제

# Boolean 자료형과 비교 연산자

# | Boolean 자료형

- Boolean 자료형은 2 종류의 데이터만 가능

- True
- False

- 파이썬 Boolean 데이터의 활용

```
true_value = True  
false_value = False  
  
print(true_value)  
print(false_value)
```



True  
False

```
true_value = true  
false_value = false
```



# | Boolean 데이터 자료형의 확인

- `type()`을 사용하여 자료형 확인

```
true_value = True  
false_value = False  
  
print(type(true_value))  
print(type(false_value))
```



```
<class 'bool'>  
<class 'bool'>
```

# | 비교 연산자(Comparison Operator)

- 두 개의 데이터를 비교

연산자	비교 방법	예시
==	값이 같은가?	x == y
!=	값이 다른가?	x != y
>	값이 큰가? (왼쪽 데이터 기준)	x > y
<	값이 작은가? (왼쪽 데이터 기준)	x < y
>=	값이 크거나 같은가? (왼쪽 데이터 기준)	x >= y
<=	값이 작거나 같은가? (왼쪽 데이터 기준)	x <= y

# |파이썬 비교 연산자(Comparison Operator)의 활용

- 비교 연산자의 결과는 항상 Boolean 값

- 값이 같은가? -> True or False
- 값이 큰가? -> True or False

```
x = 1
y = 3

print(x == y)
print(x != y)
```



```
False
True
```



# |파이썬 비교 연산자(Comparison Operator)의 활용

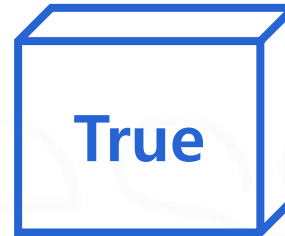
- Boolean 데이터인 True, False도 변수에 저장 가능

```
x = 1  
y = 3  
  
is_equal = (x == y)  
is_not_equal = (x != y)  
  
print(is_equal)  
print(is_not_equal)
```

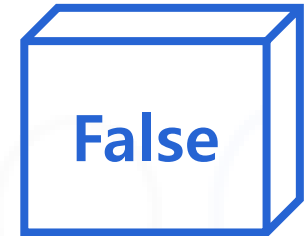


```
False  
True
```

is\_equal



is\_not\_equal



```
print(type(is_equal))  
print(type(is_not_equal))
```



```
<class 'bool'>  
<class 'bool'>
```

# |파이썬 비교 연산자(Comparison Operator)의 활용

```
x = 1
y = 3

print(x == y)
print(x != y)
print(x > y)
print(x < y)
print(x >= y)
print(x <= y)
```

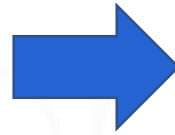


```
False
True
False
True
False
True
```

# |파이썬 비교 연산자(Comparison Operator)의 활용

```
x = 3
y = 3

print(x == y)
print(x != y)
print(x > y)
print(x < y)
print(x >= y)
print(x <= y)
```



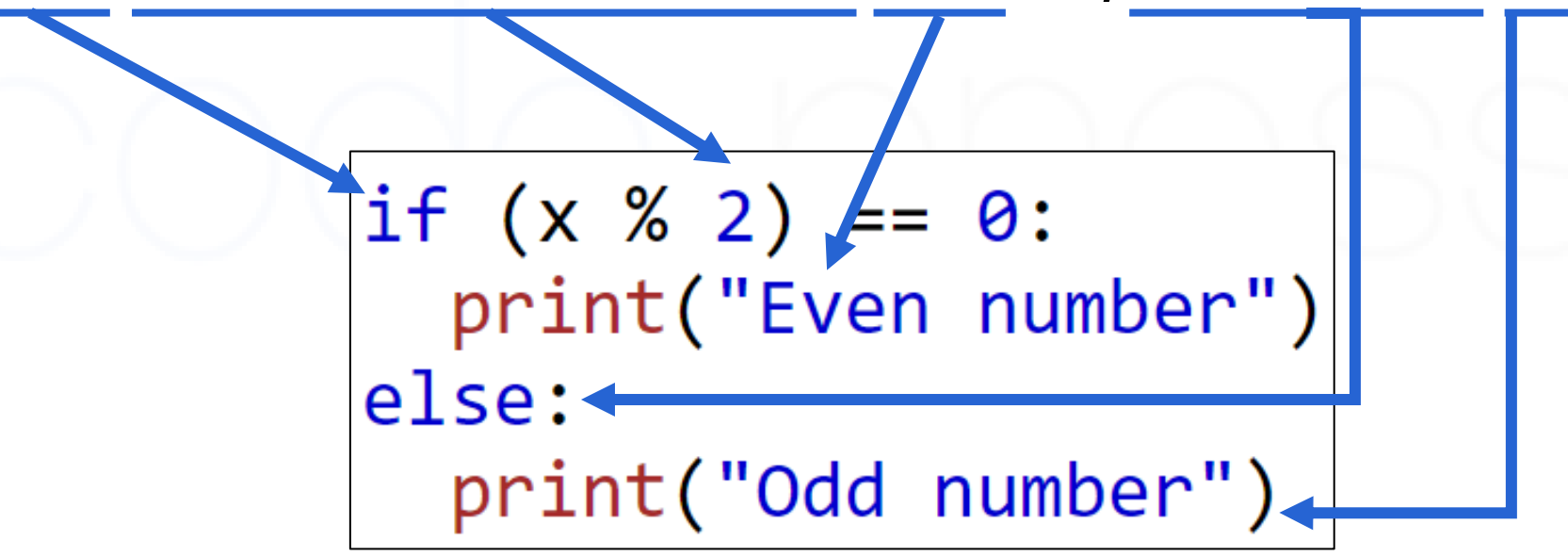
```
True
False
False
True
True
True
```

# 파이썬 조건문 (Conditional Statement)

# | 파이썬 조건문 (Conditional Statement)

- if, elif, else 를 사용(키워드)

- 만약에 x를 2로 나눈 나머지가 0이면 짝수이고, 그것이 아니면 홀수 이다



```
if (x % 2) == 0:
    print("Even number")
else:
    print("Odd number")
```

# |파이썬 조건문 - 단일 if 문

- 조건문의 유형 1 - 단일 if 문(if Statement)
- 문법
  - if 다음에 나오는 조건식을 체크(True, False)
  - 조건식의 결과가 True 이면 if 내부의 코드를 실행
  - 조건식이 결과가 False 이면 if 내부의 코드를 실행하지 않음

```
if condition1:  
    do something
```

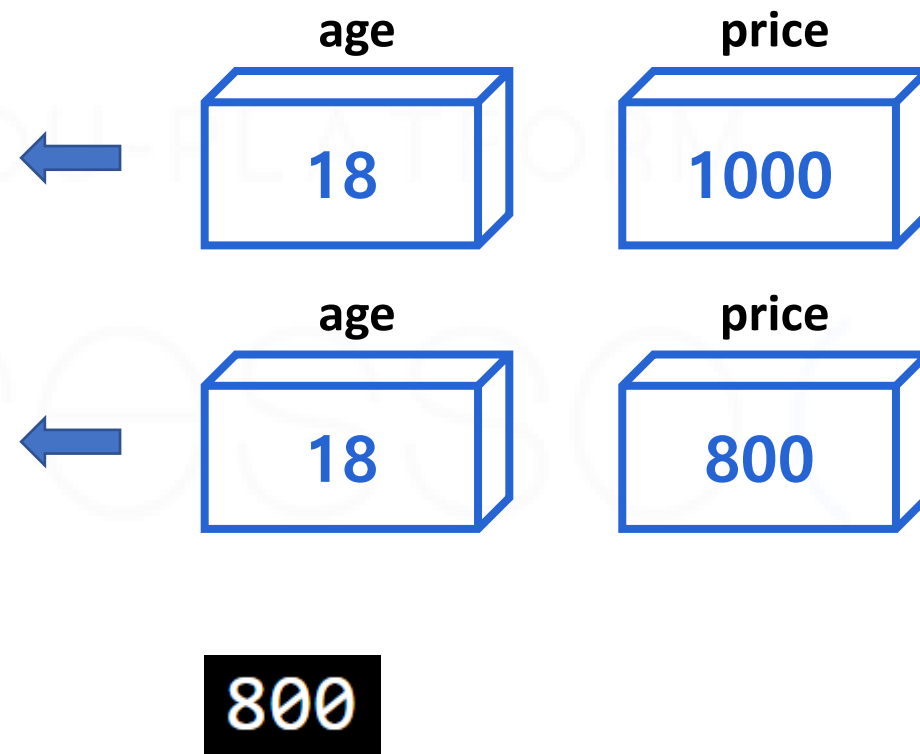
# |파이썬 조건문 - 단일 if 문

- 조건이 True 이면 if 내부의 코드를 실행

```
age = 18
price = 1000

if age < 19: True!
    price = price - 200

print(price)
```



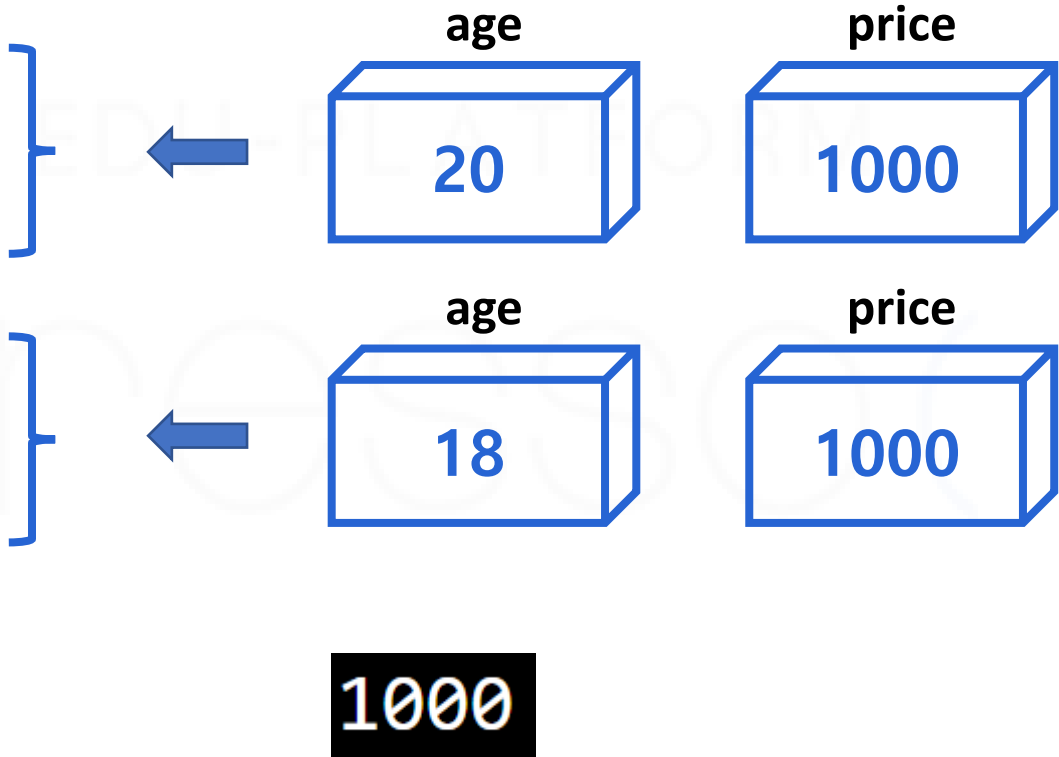
# |파이썬 조건문 - 단일 if 문

- 조건이 False 이면 if 내부의 코드를 실행하지 않음

```
age = 20
price = 1000

if age < 19: False!
price = price + 200

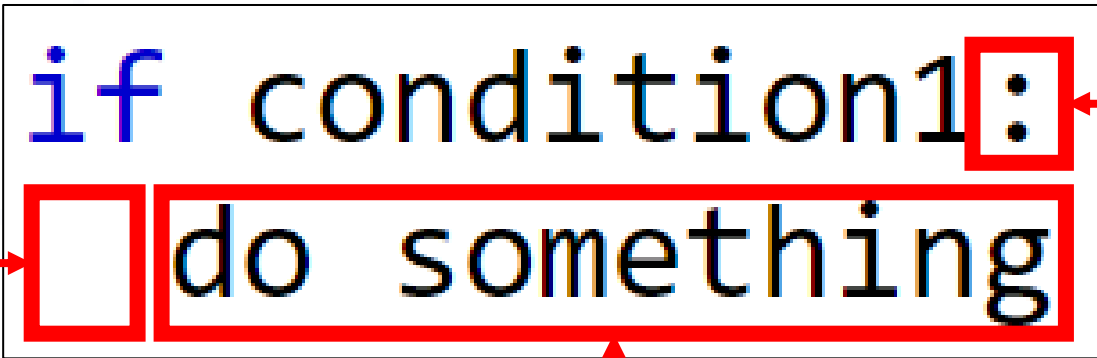
print(price)
```





# | 콜론과 들여쓰기(indentation)

- if 조건문 문법 중 콜론과 들여쓰기는 필수
- 파이썬은 들여쓰기로 코드의 영역을 명시
  - 조건식의 결과가 True 이면 if 내부의 코드를 실행
  - 조건식이 결과가 False 이면 if 내부의 코드를 실행하지 않음



The diagram shows the syntax of an if statement in Python. The text `if condition1:` is on the first line, and `do something` is on the second line, indented. A red box highlights the colon at the end of the first line, with a red arrow pointing to it from the label '콜론' (colon). Another red box highlights the indentation of the second line, with a red arrow pointing to it from the label '들여쓰기' (indentation). A third red box highlights the entire second line, with a red arrow pointing to it from the text '이 영역의 코드는 if 조건식이 True 일 경우에만 실행 됨' (The code in this area is only executed if the condition is True).

```
if condition1:  
    do something
```

콜론

들여쓰기

이 영역의 코드는  
if 조건식이 True 일 경우에만 실행 됨

# | 콜론과 들여쓰기(indentation)

- 들여쓰기는 보통 띄어쓰기 2개, 4개 또는 탭을 사용
- 특정 코드 영역에서는 동일한 들여쓰기 사용해야 함
- 들여쓰기는 필수! 들여쓰기 무시할 경우 IndentationError 발생



```
age = 18
price = 1000

if age < 19:
    price = price - 200
    print("10% discount.")

print(price)
```



```
age = 18
price = 1000

if age < 19:
    price = price - 200
    print("10% discount.")

print(price)
```



```
age = 18
price = 1000

if age < 19:
    price = price - 200
    print("10% discount.")

print(price)
```

**IndentationError: unindent does not match any outer indentation level**

# 파이썬 조건문 (Conditional Statement)

# |파이썬 조건문 - 단일 if 문

- 조건문의 유형 1 - 단일 if 문(if Statement)
- 문법
  - if 다음에 나오는 조건식을 체크(True, False)
  - 조건식의 결과가 True 이면 if 내부의 코드를 실행
  - 조건식이 결과가 False 이면 if 내부의 코드를 실행하지 않음

```
if condition1:  
    do something
```

# |파이썬 조건문 - if-else 문

- 조건문의 유형 2 - if-else 문
- if 조건식이 True이면, if 내부의 코드를 실행, else 내부의 코드는 실행하지 않음
- if 조건식이 False 이면, if 내부의 코드는 실행하지 않고, else 내부의 코드 실행

True 이면

실행

실행 안함

```
if condition1:  
    do something  
else:  
    do something2
```

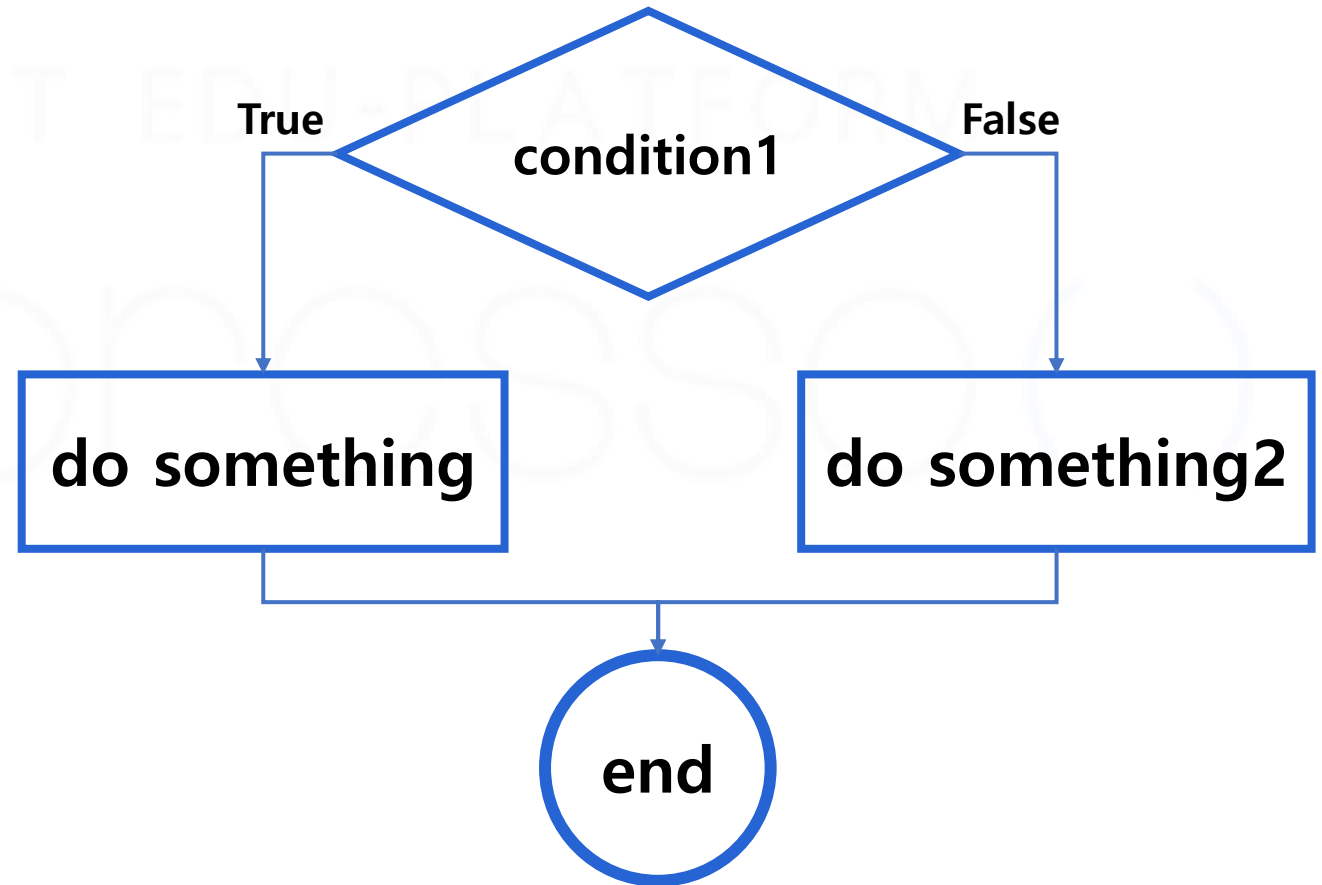
False 이면

실행 안함

실행

# |파이썬 조건문 - if-else 문

```
if condition1:  
    do something  
else:  
    do something2
```



# |파이썬 조건문 - if-else 문의 활용

- in 연산자(멤버 연산자)

- 데이터 in 리스트 변수
- 데이터가 리스트에 포함되어 있으면 True, 포함되어 있지 않으면 False

```
course_list = ["Python", "AI", "Git"]
search_keyword = "AI"

if search_keyword in course_list:
    print(search_keyword + " is in the list.")
else:
    print(search_keyword + " isn't in the list.")
```

# |파이썬 조건문 - if-else 문의 활용

- “AI”(search\_keyword)가 course\_list에 포함되어 있는지 확인
  - “AI”는 course\_list에 포함되어 있
  - “AI is in the list.” 문장이 출력 됨

```
course_list = ["Python", "AI", "Git"]
search_keyword = "AI"

if search_keyword in course_list:
    print(search_keyword + " is in the list.")
else:
    print(search_keyword + " isn't in the list.")
```



# |파이썬 조건문 - if-else 문의 활용

- “Deep Learning”(search\_keyword)가 course\_list에 포함되어 있는지 확인
  - “Deep Learning”은 course\_list에 포함되어 있지 않음
  - “Deep Learning isn't in the list.” 문장이 출력 됨

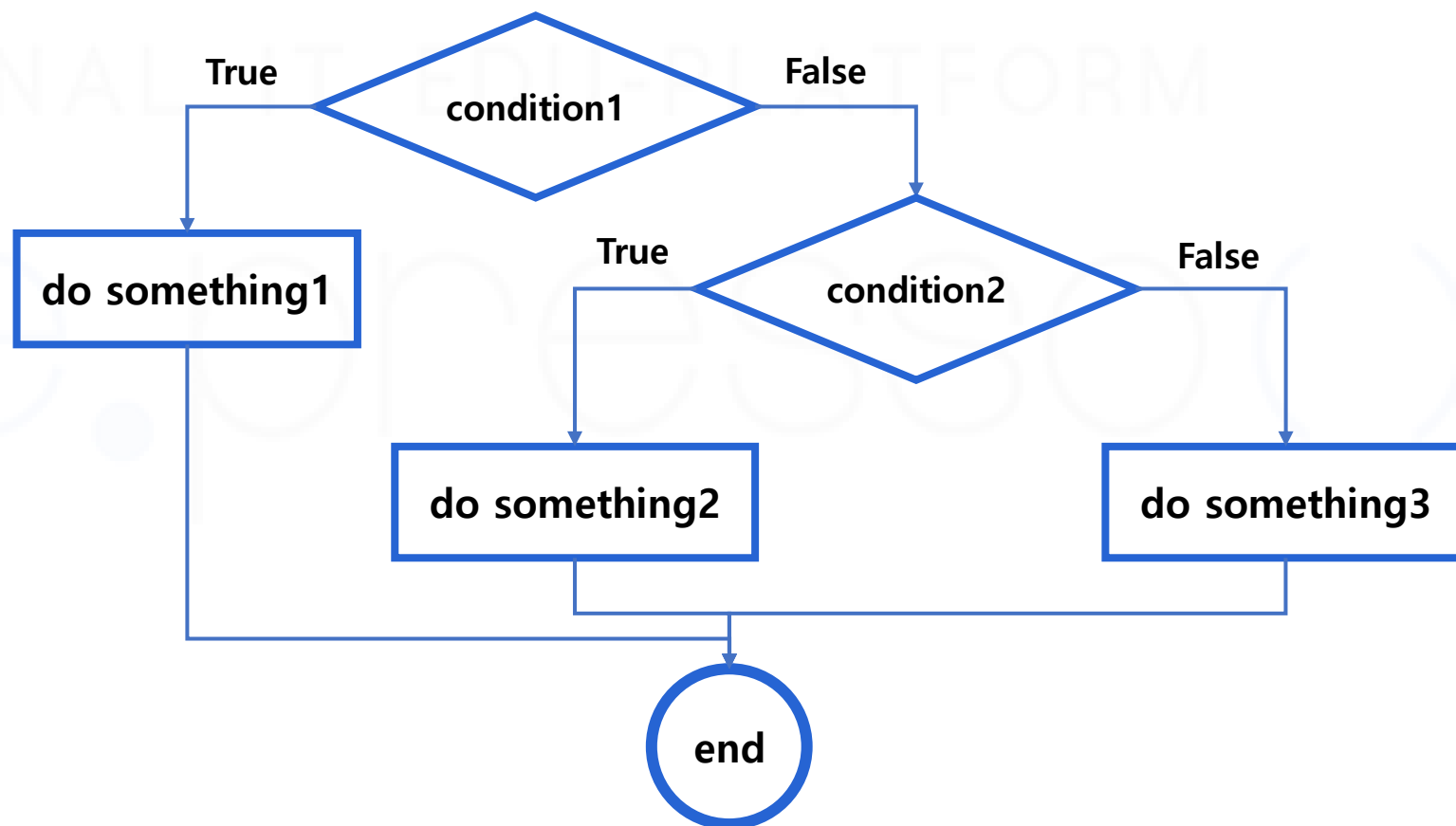
```
course_list = ["Python", "AI", "Git"]
search_keyword = "Deep Learning"

if search_keyword in course_list:
    print(search_keyword + " is in the list.")
else:
    print(search_keyword + " isn't in the list.")
```

# | 파이썬 조건문 - if-elif-else 문

- 조건문의 유형 3 - if-elif-else 문

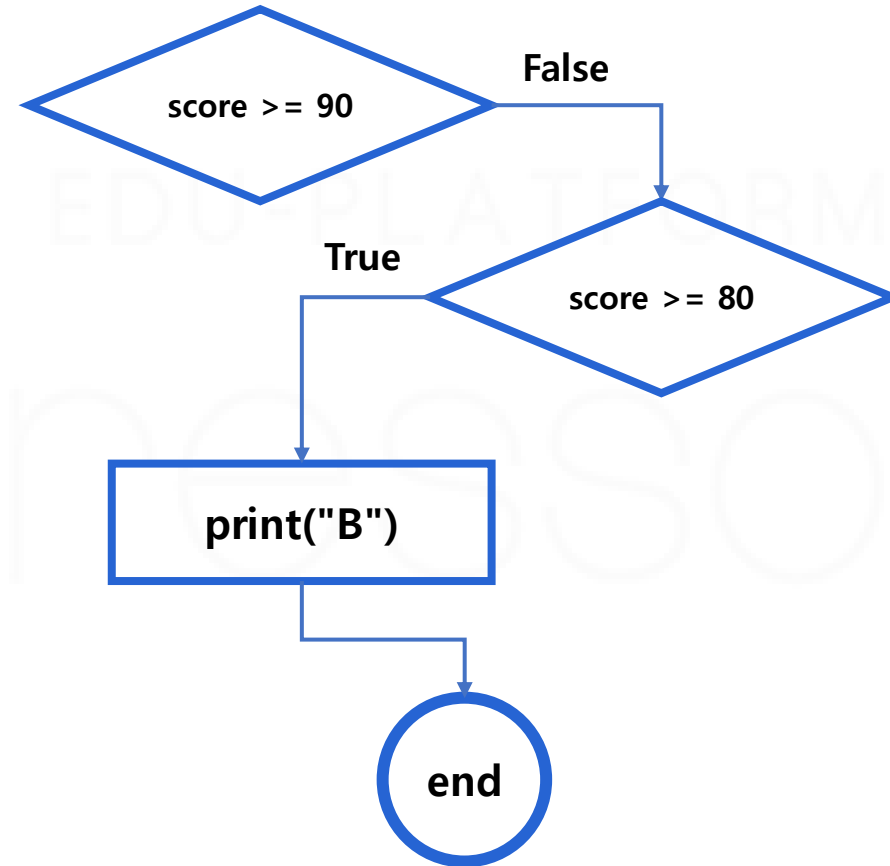
```
if condition1:  
    do something1  
elif condition2:  
    do something2  
else:  
    do something3
```



# |파이썬 조건문 - if-elif-else 문

```
score = 85

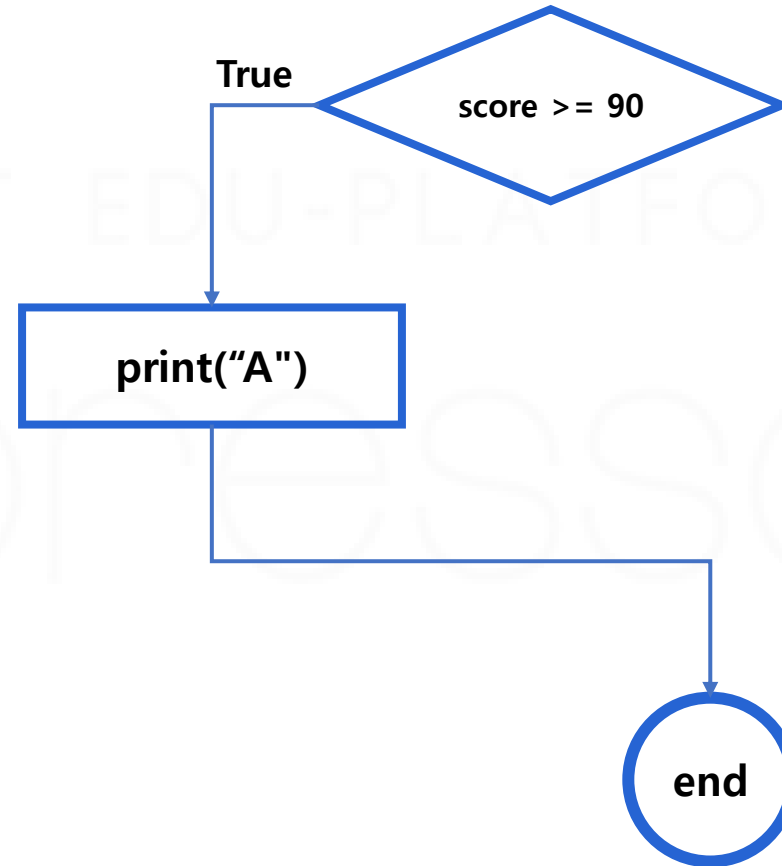
if score >= 90:
    print("A")
elif score >= 80:
    print("B")
else:
    print("F")
```



# | 파이썬 조건문 - if-elif-else 문

```
score = 97

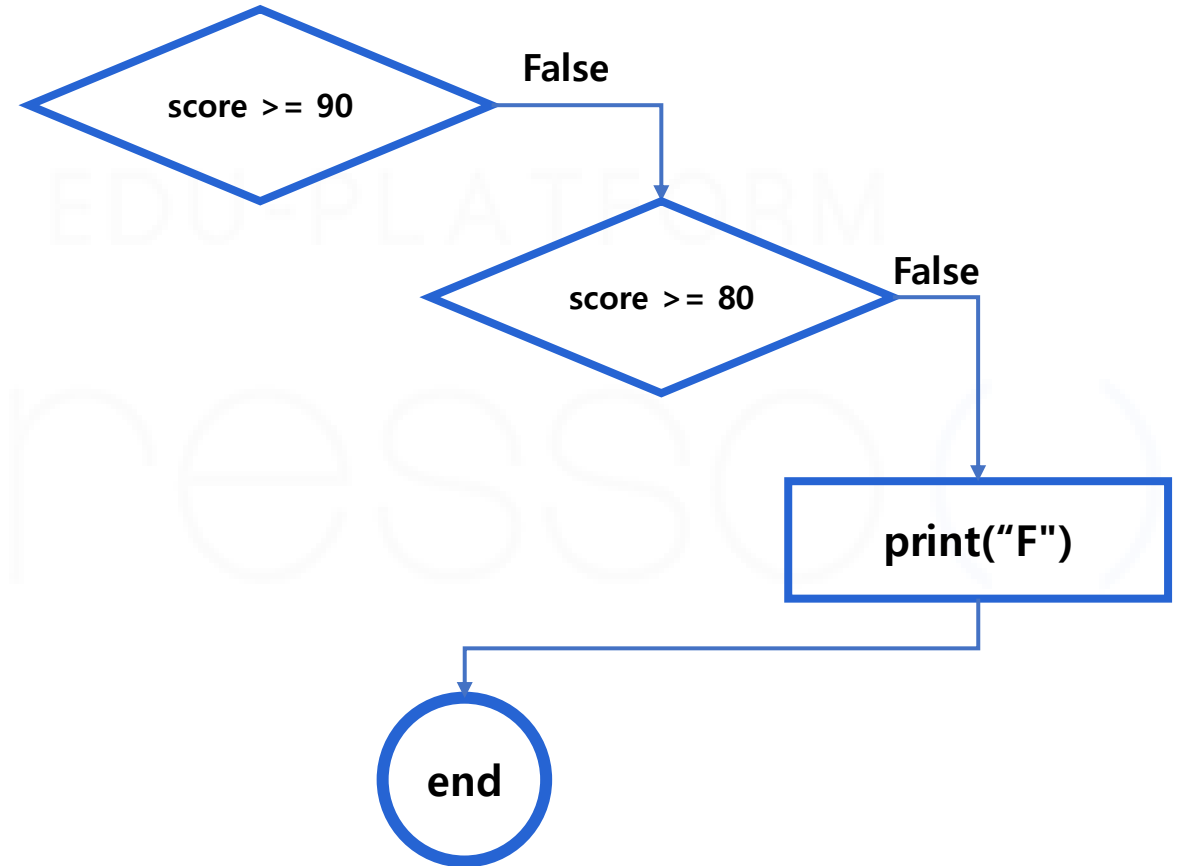
if score >= 90:
    print("A")
elif score >= 80:
    print("B")
else:
    print("F")
```



# |파이썬 조건문 - if-elif-else 문

```
score = 50

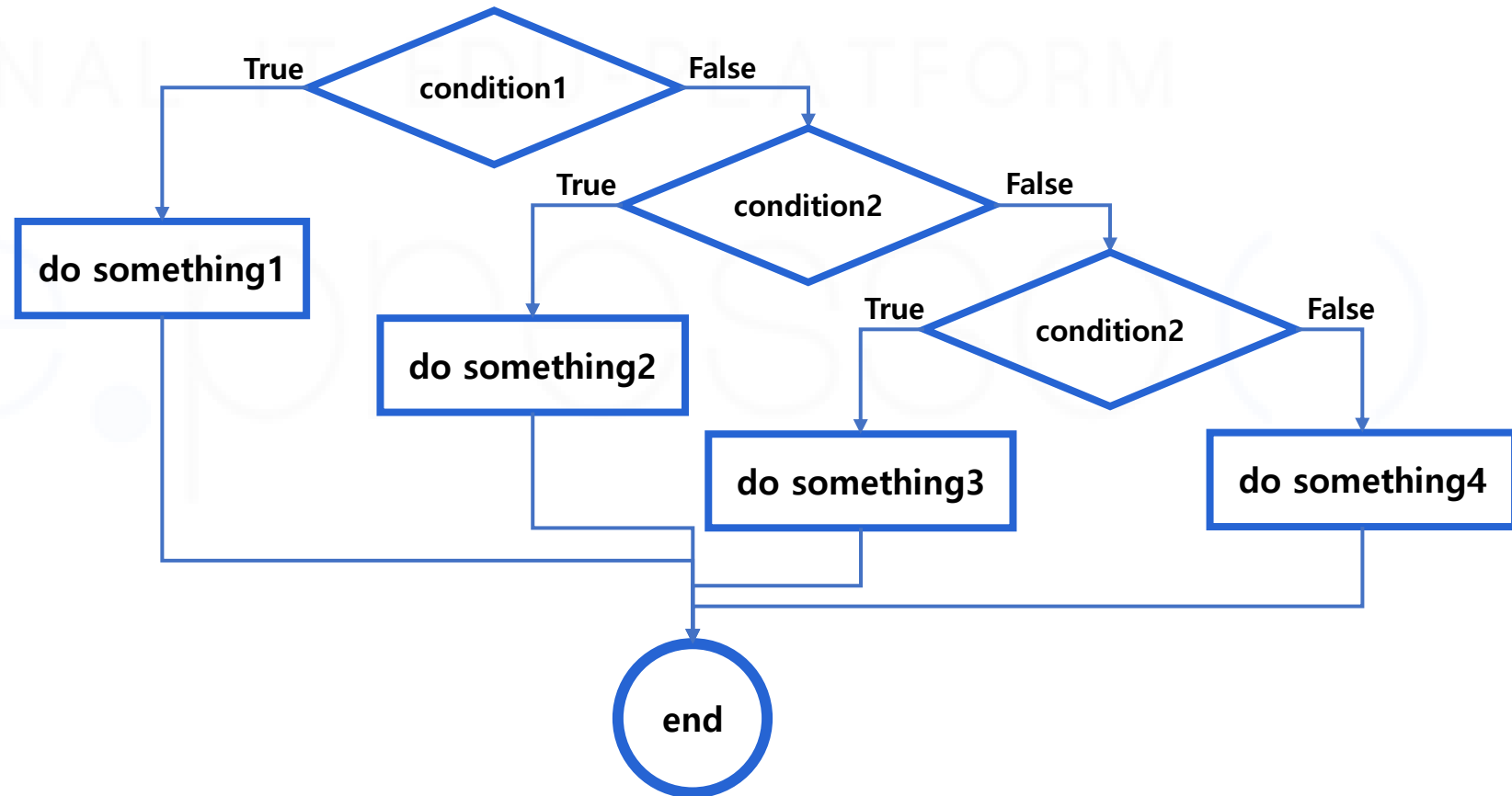
if score >= 90:
    print("A")
elif score >= 80:
    print("B")
else:
    print("F")
```



# | 파이썬 조건문 - if-다중elif-else 문

- 조건문의 유형 4 - if-다중elif-else 문

```
if condition1:  
    do something1  
elif condition2:  
    do something2  
elif condition3:  
    do something3  
else:  
    do something4
```



# |파이썬 조건문 - if-다중elif-else 문의 활용

```
number1 = 20
number2 = 4
operator = '+'
result = 0

if operator == '+':
    result = number1 + number2
elif operator == '-':
    result = number1 - number2
elif operator == '*':
    result = number1 * number2
elif operator == '/':
    result = number1 / number2
else:
    print("Error!!")

print(result)
```

```
number1 = 20
number2 = 4
operator = '*'
result = 0

if operator == '+':
    result = number1 + number2
elif operator == '-':
    result = number1 - number2
elif operator == '*':
    result = number1 * number2
elif operator == '/':
    result = number1 / number2
else:
    print("Error!!")

print(result)
```

```
number1 = 20
number2 = 4
operator = '?'
result = 0

if operator == '+':
    result = number1 + number2
elif operator == '-':
    result = number1 - number2
elif operator == '*':
    result = number1 * number2
elif operator == '/':
    result = number1 / number2
else:
    print("Error!!")

print(result)
```

# |파이썬 조건문 규칙

- if 문은 반드시 1번 사용
- else 문은 없거나 1번 사용
- elif 문은 없거나 1번 이상 다중 사용 가능
- 조건문의 유형 5 - if-elif 문

```
if condition1:  
    do something1  
elif condition2:  
    do something2
```

```
if condition1:  
    do something1  
elif condition2:  
    do something2  
elif condition3:  
    do something3
```



# 논리 연산자의 활용 (Logical Operator)

# | 논리 연산자 (Logical Operator)

- 다수의 조건들을 결합하여 최종 True/False를 연산
- 단일 조건
  - 당신은 한국 사람인가요?
- 다중 조건
  - 당신은 한국 사람이**고**, 파이썬 프로그래머 인가요?
  - 당신은 한국 사람이**거나**, 미국 사람인가요?
  - 당신은 파이썬 프로그래머 이**거나**, 자바 프로그래머 인가요?

# 'and' 연산자

- 모든 조건이 True일 경우에만 True
- 조건 중 하나라도 False이면 False
- 당신은 한국 사람이~~고~~, 파이썬 프로그래머 인가요?
  - 한국 사람 **and** 파이썬 프로그래머
- 당신은 한국 사람이~~고~~, 서울 거주하~~고~~, 나이가 20 이상~~고~~, 키가 190 이하 인가요?
  - 한국 사람 **and** 서울 거주 **and** 나이  $\geq 20$  **and** 키  $\leq 190$

# 'and' 연산자

- 모든 조건이 True일 경우에만 True
- 조건 중 하나라도 False이면 False

and 연산자의 논리식

Input		Output
True	True	True
True	False	False
False	True	False
False	False	False

# 'and' 연산자의 활용

```
print(True and True)
print(True and False)
print(False and True)
print(False and False)
```

```
True
False
False
False
```

```
print(1 > 0 and 1 < 2)
print(1 > 1 and 1 < 2)
print(2 < 1 and 1 >= 1)
print(1 > 1 and 2 < 1)
```

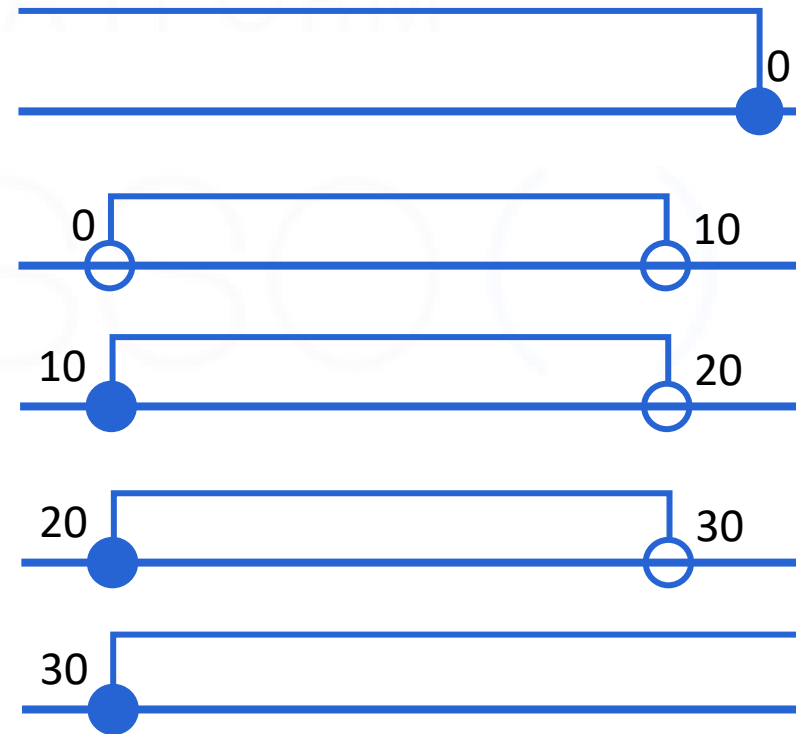
```
True
False
False
False
```

# |if 조건문과 'and' 연산자의 활용

- and 연산자는 조건문에서 빈번하게 사용 됨

```
x = 11

if x <= 0:
    print("x values between 0 and negative")
elif x > 0 and x < 10:
    print("x values between 1 and 9")
elif x >= 10 and x < 20:
    print("x values between 10 and 19")
elif x >= 20 and x < 30:
    print("x values between 20 and 29")
else:
    print("x is greater than or equal to 30")
```



# 'or' 연산자

- 결합 된 조건 중 하나라도 True이면 True
- 모든 조건이 False일 경우에만 False
- 당신은 파이썬 프로그래머 이거나, 자바 프로그래머 인가요?
  - 전문 분야 == 파이썬 or 전문 분야 == 자바
- 당신은 한국 사람이거나, 미국 사람이거나, 중국 사람인가요?
  - 국적 == 한국 or 국적 == 미국 or 국적 == 중국

# 'or' 연산자

- 결합 된 조건 중 하나라도 True이면 True
- 모든 조건이 False일 경우에만 False

or 연산자의 논리식

Input		Output
True	True	True
True	False	True
False	True	True
False	False	False



# 'or' 연산자의 활용

```
print(True or True)
print(True or False)
print(False or True)
print(False or False)
```

```
True
True
True
False
```

```
print(1 > 0 or 1 < 2)
print(1 > 1 or 1 < 2)
print(2 < 1 or 1 >= 1)
print(1 > 1 or 2 < 1)
```

```
True
True
True
False
```

# |if 조건문과 'or' 연산자의 활용

- or 연산자는 조건문에서 빈번하게 사용 됨

```
speciality = "Python"

if speciality == "Python" or speciality == "Java":
    print("Pass!")
else:
    print("We need Python or Java Programmers")
```

```
speciality = "Java"

if speciality == "Python" or speciality == "Java":
    print("Pass!")
else:
    print("We need Python or Java Programmers")
```

# |if 조건문과 'or' 연산자의 활용

- or 연산자는 조건문에서 빈번하게 사용 됨

```
speciality = "HTML"

if speciality == "Python" or speciality == "Java":
    print("Pass!")
else:
    print("We need Python or Java Programmers")
```

# | 'not' 연산자

- True는 False로, False는 True로 조건식의 결과를 뒤집음

not 연산자의 논리식

Input	Output
True	False
False	True

```
print(not True)
print(not False)
```

False  
True

# 조건문의 중첩

# | 조건문의 중첩 (Nested Conditional Statement)

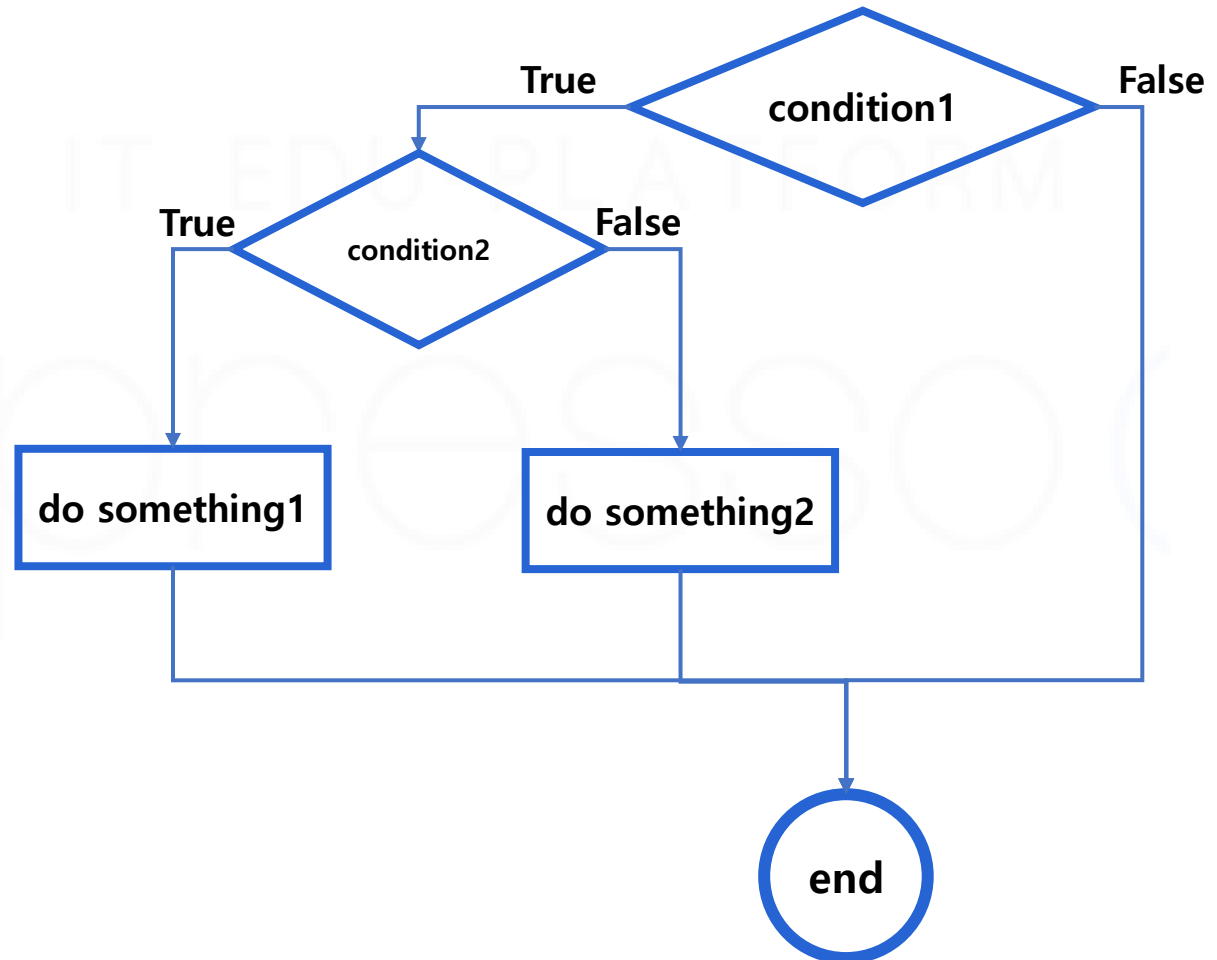
- if, elif, else 문 하위에 또다른 if-(elif-else)가 중첩된 형태

```
if condition1:  
    if condition2:  
        do something1  
    else:  
        do something2
```

```
if condition1:  
    if condition2:  
        do something1  
    else:  
        do something2  
else:  
    if condition3:  
        do something3  
    else:  
        do something4
```

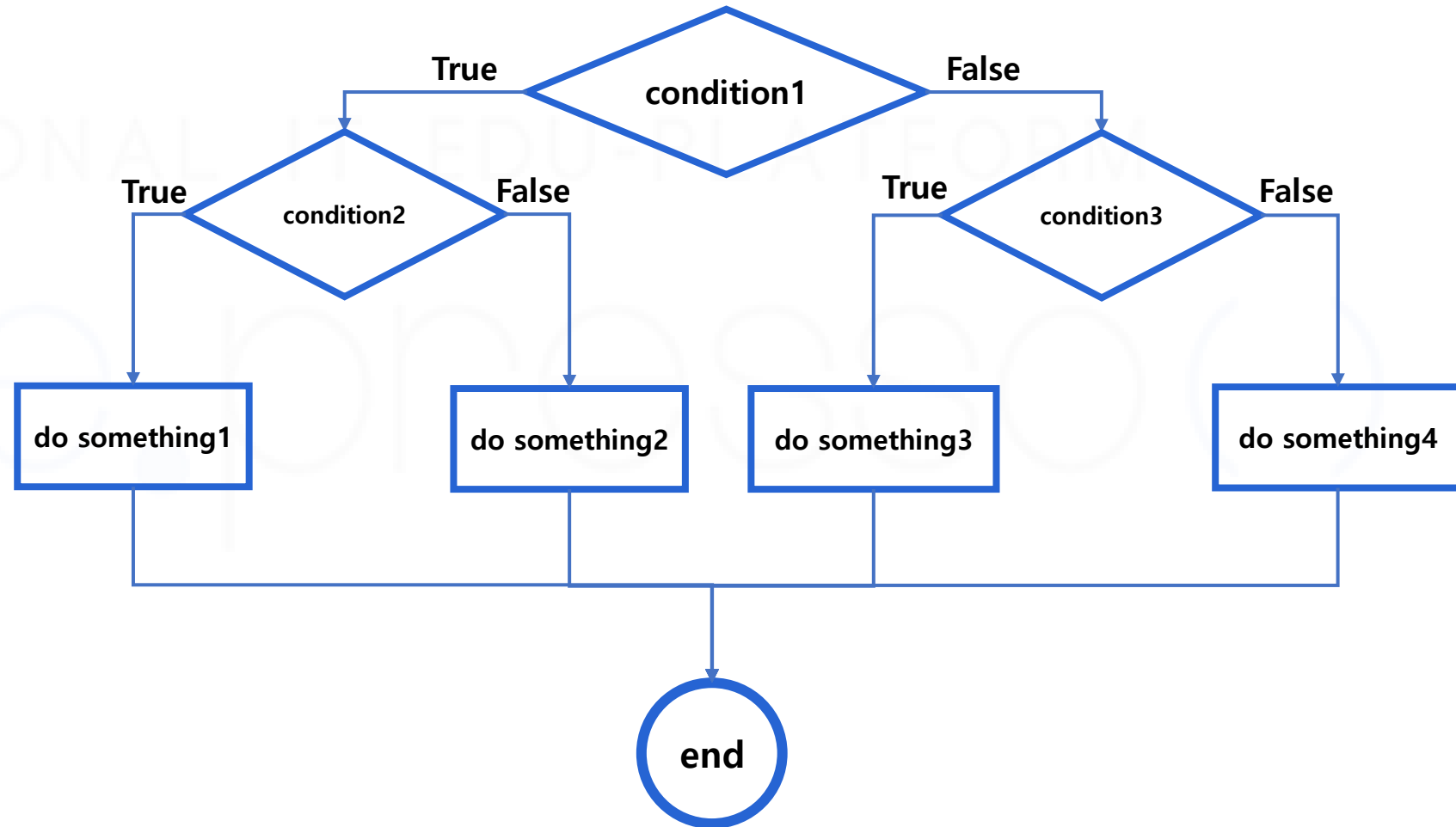
# | 조건문의 중첩 (Nested Conditional Statement)

```
if condition1:  
    if condition2:  
        do something1  
    else:  
        do something2
```



# | 조건문의 중첩 (Nested Conditional Statement)

```
if condition1:  
    if condition2:  
        do something1  
    else:  
        do something2  
else:  
    if condition3:  
        do something3  
    else:  
        do something4
```





# | 조건문의 중첩의 활용

```
age = 27
ticket_price = 0

if age >= 19:
    if age >= 65:
        ticket_price = 5000
    else:
        ticket_price = 10000
else:
    if age >= 8:
        ticket_price = 5000
    else:
        ticket_price = 0

print(ticket_price)
```

```
age = 5
ticket_price = 0

if age >= 19:
    if age >= 65:
        ticket_price = 5000
    else:
        ticket_price = 10000
else:
    if age >= 8:
        ticket_price = 5000
    else:
        ticket_price = 0

print(ticket_price)
```