

4주

조건문



염 희 균

## 목차

- 불 자료형과 if 조건문
- if~elif
- elif
- 확인문제

# 불자료형

- Boolean
  - 불린 / 불리언 / 불
  - True와 False 값만 가질 수 있음

```
>>> print(True)
True
>>> print(False)
False
```

연산자	설명	연산자	설명
==	같다	>	크다
!=	다르다	<=	작거나 같다
<	작다	>=	크거나 같다

## 불 만들기 : 비교 연산자

- 숫자 또는 문자열에 적용

```
>>> print(10 == 100)
False
>>> print(10 != 100)
True
>>> print(10 < 100)
True
>>> print(10 > 100)
False
>>> print(10 <= 100)
True
>>> print(10 >= 100)
False
```

조건식	의미	결과
10 == 100	10과 100은 같다	거짓
10 != 100	10과 100은 다르다	참
10 < 100	10은 100보다 작다	참
10 > 100	10은 100보다 크다	거짓
10 <= 100	10은 100보다 작거나 같다	참
10 >= 100	10은 100보다 크거나 같다	거짓

## 불 만들기 : 비교 연산자

- 문자열에도 비교 연산자 적용 가능

```
>>> print("가방" == "가방")
True
>>> print("가방" != "하마")
True
>>> print("가방" < "하마")
True
>>> print("가방" > "하마")
False
```

# 불 연산하기 : 논리 연산자

- 불끼리 논리 연산자 사용 가능

연산자	의미	설명
not	아니다	불을 반대로 전환합니다.
and	그리고	피연산자 두 개가 모두 참일 때 True를 출력하며, 그 외는 모두 False를 출력합니다.
or	또는	피연산자 두 개 중에 하나만 참이라도 True를 출력하며, 두 개가 모두 거짓일 때만 False를 출력합니다.

- not 연산자
  - 단항 연산자
  - 참과 거짓 반대로 바꿈

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

## 불 연산하기 : 논리 연산자

- 예시 - not 연산자 조합하기

```
01 x = 10
02 under_20 = x < 20
03 print("under_20:", under_20)
04 print("not under_20:", not under_20)
```

실행결과

under\_20: True  
not under\_20: False

# 불 연산하기 : 논리 연산자

- and 연산자와 or 연산자

- and 연산자는 양쪽 변의 값이 모두 참일 때만 True를 결과로 냄

- and 연산자

좌변	우변	결과
True	True	True
True	False	False
False	True	False
False	False	False

- or 연산자

좌변	우변	결과
True	True	True
True	False	True
False	True	True
False	False	False



## 불 연산하기 : 논리 연산자

- 예시 - and 연산자와 or 연산자

"사과 그리고 배 가져와!"

"사과 또는 배 가져와!"

"치킨(True) 그리고 쓰레기(False) 가져와!"

"치킨(True) 또는 쓰레기(False) 가져와!"

```
>>> print(True and True)
True
>>> print(True and False)
False
>>> print(False and True)
False
>>> print(False and False)
False
>>> print(True or True)
True
>>> print(True or False)
True
>>> print(False or True)
True
>>> print(False or False)
False
```

## in / not in 연산자

- in 연산자
  - 특정 값이 컨테이너 자료형(예:리스트,튜플, 딕셔너리) 내부에 있는지 확인

값 in 리스트

```
>>> list_a = [273, 32, 103, 57, 52]
>>> 273 in list_a
True
>>> 99 in list_a
False
>>> 100 in list_a
False
>>> 52 in list_a
True
```

## in/not in 연산자

- not in 연산자
  - 컨테이너자료형 내부에 해당 값이 없는지 확인

```
>>> list_a = [273, 32, 103, 57, 52]
>>> 273 not in list_a
False
>>> 99 not in list_a
True
>>> 100 not in list_a
True
>>> 52 not in list_a
False
>>> not 273 in list_a
False
```

## 논리 연산자의 활용

- and 연산자



- or 연산자



# if 조건문이란

- if (조건식) :
  - 조건에 따라 코드 실행하거나 실행하지 않게 할 때 사용하는 구문
  - 조건 분기

if 불 값이 나오는 표현식: → if의 조건문 뒤에는 반드시 콜론(:)을 붙여줘야 합니다.

□□□ 불 값이 참일 때 실행할 문장

□□□ 불 값이 참일 때 실행할 문장

□□□□는 들여쓰기 4칸

↓  
if문 다음 문장은 4칸 들여쓰기 후 입력합니다.

- 조건식 다음엔 반드시 콜론(:)사용
- 조건식 쓸 때 괄호 안써도 됨
- 실행문 앞부분엔 반드시 들여쓰기

## if 조건문이란

```
if num>0:
    print("양수입니다.")
elif num<0:
    print("음수입니다.")
else:
    print("0 입니다.")
```



```
if num>0:print("양수입니다.")
elif num<0:print("음수입니다.")
else:print("0 입니다.")
```

들여쓰지 않고 한줄로도 표현 가능하나, 가독성이 떨어짐

```
if num>0:
    ←||→ print("입력한 수는")
    ←||→ print("양수")
    ←||→ print("입니다.")
```

조건문이 참일 때 실행하고자 하는 명령이 2개 이상이면 **들여쓰기 수준을 맞춰서** 아래쪽으로 명령 계속 나열

# if 조건문이란

- 예시1

```
>>> if True: Enter
    print("True입니다...!") Enter
    print("정말 True입니다...!") Enter
    Enter
True입니다...!
정말 True입니다...!
```

```
>>> if False: Enter
    print("False입니다...!") Enter
    Enter
>>>
```

## if 조건문이란


- 예시2

```
num1 = 20
num2 = 30

if(num1 > num2):
    print('{0} > {1}'.format(num1, num2))

if(num1 < num2):
    print('{0} < {1}'.format(num1, num2))

if(num1 == num2):
    print('{0} == {1}'.format(num1, num2))
```



```
20 < 30
>>>
```



## else 조건문의 활용

- else 구문

- if 조건문 뒤에 사용하며, if 조건문의 조건이 거짓일 때 실행되는 부분

```
if 조건:  
    조건이 참일 때 실행할 문장  
else:  
    조건이 거짓일 때 실행할 문장
```

□□□□는 들여쓰기 4칸

- 조건문이 오로지 두 가지로만 구분될 때 if else 구문을 사용하면 조건 비교를 단 한번만 하므로 이전의 코드보다 두 배 효율적

## else 조건문의 활용

- 예시

```
num1 = 20
num2 = 30

# if ~ else문
if(num1 > num2):
    print('{0} > {1}'.format(num1, num2))
else:
    print('{0} <= {1}'.format(num1, num2))
```

20 <= 30  
>>>

## elif 구문

- elif 구문(else if, 다른 언어의 switch-case 문 대체)
  - 세 개 이상의 조건을 연결해서 사용
  - if 조건문과 else 구문 사이에 입력

```
if 조건A:  
    조건A가 참일 때 실행할 문장  
elif 조건B:  
    조건B가 참일 때 실행할 문장  
elif 조건C:  
    조건C가 참일 때 실행할 문장  
...  
else:  
    모든 조건이 거짓일 때 문장
```

□□□□는 들여쓰기 4칸

# elif 구문

- 예시

```
score=65
if score>=90:
    print("A+")
else:
    if score>=80:
        print("A")
    else:
        if score>=70:
            print("B")
        else:
            if score>=60:
                print("C")
            else:
                print("F")
```

else if 사용  
(계속 들여써야 함 - 가독성 ↓)



```
score=65
if score>=90:
    print("A+")
elif score>=80:
    print("A")
elif score>=70:
    print("B")
elif score>=60:
    print("C")
else:
    print("F")
```

elif 사용  
(가독성 Good)

## 조건식에서 문자열 비교 가능 예

```
lang="python"
if lang=="Python":
    print("파이썬1 입니다.")
elif lang=="python":
    print("파이썬2 입니다.")
elif lang=="Java":
    print("자바 입니다.")
```

```
===== RESTART: C:\Users\WJihoon\Desktop\w2.py =====
파이썬2 입니다.
```

- == 연산자를 이용해 문자열 비교 가능
- 대소문자를 구분하여 비교
- 문자열끼리 크고 작음을 비교할 때는 사전순으로 비교  
뒤에 나올 수록 더 큰 문자로 판단 e.g. a<c

```

1 package test;
2
3 public class Example{
4
5     public static void main(String[] args){
6         // TODO Auto-generated method stub
7
8         String str1 = new String("Morph");
9         String str2 = new String("House");
10
11         if(str1.equals(str2))
12         {
13             System.out.println("같다.");
14         }
15         else
16         {
17             System.out.println("다르다.");
18         }
19     }
20 }
21

```

Console Problems Javadoc Declaration

<terminated> Example [Java Application] C:\Program Files\

다르다.

# elif 구문

- 예시 - 계절 구하기

```
01  # 날짜/시간과 관련된 기능을 가져옵니다.
02  import datetime
03
04  # 현재 날짜/시간을 구하고
05  # 쉽게 사용할 수 있게 월을 변수에 저장합니다.
06  now = datetime.datetime.now()
07  month = now.month
08
09  # 조건문으로 계절을 확인합니다.
10  if 3 <= month <= 5:
11      print("현재는 봄입니다.")
12  elif 6 <= month <= 8:
13      print("현재는 여름입니다.")
14  elif 9 <= month <= 11:
15      print("현재는 가을입니다.")
16  else:
17      print("현재는 겨울입니다.")
```

실행결과

현재는 봄입니다

# if 조건문을 효율적으로 사용하기

- 조건문의 활용

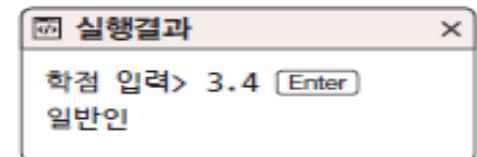
- 예시

조건	설명(학생 평가)	조건	설명(학생 평가)
4.5	신	1.75~2.3	오락문화의 선구자
4.2~4.5	교수님의 사랑	1.0~1.75	불가촉천민
3.5~4.2	현 체제의 수호자	0.5~1.0	자벌레
2.8~3.5	일반인	0~0.5	플랑크톤
2.3~2.8	일탈을 꿈꾸는 소시민	0	시대를 앞서가는 혁명의 씨앗

```
01  # 변수를 선언합니다.
02  score = float(input("학점 입력> "))
03
04  # 조건문을 적용합니다.
05  if score == 4.5:
06      print("신")
07  elif 4.2 <= score < 4.5:
08      print("교수님의 사랑")
```

## if 조건문을 효율적으로 사용하기

```
09 elif 3.5 <= score < 4.2:
10     print("현 체제의 수호자")
11 elif 2.8 <= score < 3.5:
12     print("일반인")
13 elif 2.3 <= score < 2.8:
14     print("일탈을 꿈꾸는 소시민")
15 elif 1.75 <= score < 2.3:
16     print("오락문화의 선구자")
17 elif 1.0 <= score < 1.75:
18     print("불가촉천민")
19 elif 0.5 <= score < 1.0:
20     print("자벌레")
21 elif 0 < score < 0.5:
22     print("플랑크톤")
23 elif score == 0:
24     print("시대를 앞서가는 혁명의 씨앗")
```



- 위에서 제외된 조건을 한 번 더 검사하여 비효율적



# if 조건문을 효율적으로 사용하기

```
01 # 변수를 선언합니다.
02 score = float(input("학점 입력> "))
03
04 # 조건문을 적용합니다.
05 if score == 4.5:
06     print("신")
07 elif 4.2 <= score:
08     print("교수님의 사랑")
09 elif 3.5 <= score:
10     print("현 체제의 수호자")
11 elif 2.8 <= score:
12     print("일반인")
13 elif 2.3 <= score:
14     print("일탈을 꿈꾸는 소시민")
15 elif 1.75 <= score:
16     print("오락문화의 선구자")
17 elif 1.0 <= score:
18     print("불가촉천민")
19 elif 0.5 <= score:
20     print("자벌레")
21 elif 0 < score:
22     print("플랑크톤")
23 else:
24     print("시대를 앞서가는 혁명의 씨앗")
```

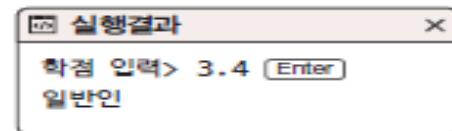
하위 값만 검사하고 상위 값은 검사를 생략

elif 4.2 <= score < 4.5:



elif 4.2 <= score:

조건 비교를 반으로 줄이고 코드 가독성 향상됨

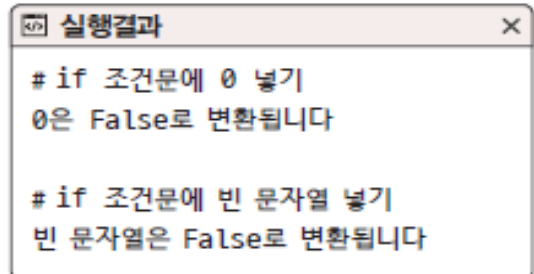


# False로 변환되는 값

- 빈 컨테이너

- if 조건문의 매개변수에 불 아닌 다른 값이 올 때 자동으로 불로 변환
- 이 때 False로 변환되는 값: None, 0.0, 빈 문자열, 빈 바이트열, 빈 리스트

```
01 print("# if 조건문에 0 넣기")
02 if 0:
03     print("0은 True로 변환됩니다")
04 else:
05     print("0은 False로 변환됩니다")
06 print()
07
08 print("# if 조건문에 빈 문자열 넣기")
09 if "":
10     print("빈 문자열은 True로 변환됩니다")
11 else:
12     print("빈 문자열은 False로 변환됩니다")
```



실행결과

```
# if 조건문에 0 넣기
0은 False로 변환됩니다

# if 조건문에 빈 문자열 넣기
빈 문자열은 False로 변환됩니다
```

# pass 키워드

- 나중에 구현하고자 구문을 비워 두는 경우

```
if zero == 0
    빈 줄 삽입
else:
    빈 줄 삽입
```

---

```
01  # 입력을 받습니다.
02  number = input("정수 입력> ")
03  number = int(number)
04
05  # 조건문 사용
06  if number > 0:
07      # 양수일 때: 아직 미구현 상태입니다.
08  else:
09      # 음수일 때: 아직 미구현 상태입니다.
```

---

# pass 키워드

- IndentationError

- if 조건문 사이에는 무조건 들여쓰기 4칸 넣고 코드 작성해야 함

- pass 키워드

- 아무것도 작성하지 않고 임시적으로 비워 둠

---

```
01  # 입력을 받습니다.
02  number = input("정수 입력> ")
03  number = int(number)
04
05  # 조건문 사용
06  if number > 0:
07      # 양수일 때: 아직 미구현 상태입니다.
08      pass
09  else:
10      # 음수일 때: 아직 미구현 상태입니다.
11      pass
```

---

## 키워드로 정리하는 핵심 포인트

- **else 구문** : if 조건문 뒤에 사용하며, if 조건문의 조건이 거짓일 때 실행
- **elif 구문** : if 조건문과 else 구문 사이에 입력하며, 세 개 이상의 조건을 연결해서 사용할 때 적절
- **False로 변환되는 값** : if 조건문의 조건식에서 False로 변환되는 값은 None, 0, 0.0, 빈 문자열, 빈 바이트 열, 빈 리스트, 빈 튜플, 빈 딕셔너리 등이 있음
- **pass 키워드** : 프로그래밍의 전체 골격을 잡아두고 내부에 처리할 내용은 나중에 만들고자 할 때 pass 키워드 입력

## 확인문제

- 다음 코드의 실행결과를 예측해 빈칸에 결괏값을 입력하세요. 아래의 코드는 모두 같고 입력 결과가 다른 경우입니다.

```
x = 2  
y = 10
```

```
if x > 4:  
    if y > 2:  
        print(x * y)  
else:  
    print(x + y)
```

```
x = 1  
y = 4
```

```
if x > 4:  
    if y > 2:  
        print(x * y)  
else:  
    print(x + y)
```

```
x = 10  
y = 2
```

```
if x > 4:  
    if y > 2:  
        print(x * y)  
else:  
    print(x + y)
```

## 확인문제

- 다음 중첩 조건문에 논리 연산자 적용해 하나의 if 조건문으로 만들어 주세요.

```
if x > 10:  
    if x < 20:  
        print("조건에 맞습니다.")
```



```
print("조건에 맞습니다.")
```

## 확인 문제

- 사용자에게 태어난 연도를 입력 받아 띠를 출력하는 프로그램을 작성해 보시오.
- 작성 시 입력 받은 연도를 12로 나눈 나머지를 사용합니다. 나머지가 0,1,2,3,4,5,6,7,8,9,10,11 일 때 각각 원숭이 , 닭, 개, 돼지, 쥐, 소, 범, 토끼, 용, 뱀, 말, 양띠입니다.