불 자료형, 관계 연산자, 논리 연산자

1 불(Bool) 자료형

참(True)과 거짓(False)을 나타내는 자료형

- 오직 참과 거짓 두 가지 값만 가질 수 있음
- 파이썬은 대·소문자를 구별하기 때문에 정확히 True, False로 사용해야 함

```
a = True
print(type(a))
```

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

```
b = False
print(type(b))
```

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

```
a = true
```

```
---------------------------------------------
NameError                                 Tra
<ipython-input-4-c100e6034b0a> in <module>
----> 1 a = true

NameError: name 'true' is not defined
```

불 자료형, 관계 연산자, 논리 연산자

1 불(Bool) 자료형

내장 함수 bool()을 활용해 참과 거짓을 확인할 수 있음

자료형의 참과 거짓
① 문자열 ② 리스트 ③ 튜플 ④ 사전 ⑤ 숫자

```
print(bool("abcd"))
print(bool(""))
```

```
True
False
```

```
print(bool({'a':1,'b':2}))
print(bool({}))
```

```
True
False
```

```
print(bool([1,2,3]))
print(bool([]))
```

```
True
False
```

```
print(bool((1,2,3)))
print(bool(()))
```

```
True
False
```

```
print(bool(1))
print(bool(0))
```

```
True
False
```

불 자료형, 관계 연산자, 논리 연산자

2 관계 연산자(비교 연산자)

크다, 작다, 같다, 다르다, 크거나 같다,
작거나 같다

관계 연산자는 Bool 값을 반환

```
print(1 == 1)
print(1 == 2)
```

```
True
False
```

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

```
True
True
```

```
print(1 >= 1)
print(1 != 0)
```

```
True
True
```

불 자료형, 관계 연산자, 논리 연산자

3 논리 연산자

논리 값을 판단해주는 연산자로 and, or, not
세 가지가 있음

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

불 자료형, 관계 연산자, 논리 연산자

4 연산자 활용

프로그래밍을 할 때, 관계 연산자와 논리 연산자를 함께 사용하는 경우가 많음

```
a = 1
b = 2

print(a > 0 and b >1)
```

```
True
```

```
a = 1
b = 2

print(a == 0 or b != 1)
```

```
True
```

불 자료형, 관계 연산자, 논리 연산자

4 연산자 활용

and, or 연산자의 출력

and 연산자	or 연산자
둘 다 True일 때만 True이므로, 오른쪽 출력	둘 중 하나만 True여도 True이므로, 왼쪽 우선 출력

```
print("a" and "b")
print("a" or "b")
```

```
b
a
```

파이썬의 조건문

1 조건문의 정의

프로그램의 실행을 제어하기 위한 제어문 중 하나로 조건에 따라 실행 결과를 다르게 할 수 있음

- if (조건) :
  코드
- **콜론, 들여쓰기 필수!**(들여쓰기는 Tab 키 활용)

```
if (True):
    print(1)
print(2)
```

```
1
2
```

```
if (False):
    print(1)
print(2)
```

```
2
```

```
if (True):
print(1)
print(2)
```

```
  File "<ipython-input-52-6bfbf1da58c9>",
    print(1)
        ^
IndentationError: expected an indented block
```

파이썬의 조건문

2 조건문의 종류와 특징

1 if else 조건문

들여쓰기 주의! 이때, if와 else는 같은 레벨

```
a = 2
if (a == 1):
    print(1)
else:
    print(2)
```

```
2
```

```
a = 2
if (a == 1):
    print(1)
    else:
        print(2)
```

```
  File "<ipython-input-55-5b
    else:
    ^
SyntaxError: invalid syntax
```

파이썬의 조건문

2 조건문의 종류와 특징

2 if elif else 조건문

- elif를 활용해 여러 개의 조건 추가 가능

조건이 같지 않도록 주의!
조건은 위에서부터 순서대로 확인하며 내려옴

조건이 같기 때문에 두 번째 조건은 무시

```
a = 2
if (a == 2):
    print(1)
elif(a == 2):
    print(2)
else:
    print(4)
```

```
1
```

```
a = 2
if (a == 1):
    print(1)
elif(a == 2):
    print(2)
else:
    print(3)
```

```
2
```

```
a = 2
if (a == 1):
    print(1)
elif(a == 2):
    print(2)
elif(a == 3):
    print(3)
else:
    print(4)
```

```
2
```

파이썬의 조건문

2 조건문의 종류와 특징

3 중첩 조건문

- 들여쓰기를 조정해서 조건문 안에 또 다른 조건문 추가 가능

```
a = 1
b = 2
if (a == 1):
    if(b ==3):
        print(1)
    else:
        print(2)
else:
    print(3)
```

```
2
```

```
a = 1
b = 3
if (a == 1):
    if(b == 3):
        if(a < b):
            print(0)
        print(1)
    else:
        print(2)
else:
    print(3)
```

```
0
1
```

조건문의 활용

1 연산자를 활용한 조건문

관계 연산자, 논리 연산자 등 각종 연산자를
활용해 복잡한 조건의 프로그램 제어 가능

- 복잡한 조건문일수록, 들여쓰기가 중요!

```
a = 2
b = 1

if a == 1:
    print('1')
    if b == 1:
        print('11')
elif a == 2:
    print('2')
else:
    print('3')
```

```
2
```

≠

```
a = 2
b = 1

if a == 1:
    print('1')
if b == 1:
    print('11')
elif a == 2:
    print('2')
else:
    print('3')
```

```
11
```

조건문의 활용

1 연산자를 활용한 조건문

여러 줄의 복잡한 조건문을 한 줄로 표현 가능

- True일 경우 if 조건식
- False일 경우 else 조건식

```
a = 1
if (a == 1):
    msg = 'a is 1'
else:
    msg = 'a is not 1'

print(msg)
```

```
a is 1
```

```
msg2 = "a is 1" if (a == 1) else "a is not 1"
print(msg2)
```

```
a is 1
```

Run! 프로그래밍

Mission 1 | 조건문 활용

```
a = input("키를 입력 해주세요 :")
b = input("몸무게를 입력 해주세요 :“)
```

정답

```
a = int(a)
b = int(b)

if (a>180):
    print("남자입니다.")

elif (a>=160 and a <= 180):
     if(b>(a-110)):
        print("남자입니다.")
     else:
        print("여자입니다.")
else:
    print("여자입니다.")
```

Run! 프로그래밍

Mission 2 | 조건문 표현식 활용

```
a = 1

if (a>0):
    msg = "a는 양수입니다.“
else:
    msg = 'a는 음수입니다.’

if(a==0):
    msg2 = 'a는 0입니다.’
else:
    msg2 = 'a는 0이 아닙니다.'

print(msg,msg2)
```

정답

```
msg = "a는 양수" if (a>0) else "a는 음수",
"a는 0입니다" if (a ==0) else "a는 0이 아닙니다."
print(msg)
```