

### 조건문과 반복문



## 불리언 로직 (Boolean logic)

Data Type	True & False
"Python"	True
1111	False
[1, 2, 3]	True
	False
0	False
{}	False
1	True
0	False
None	False

String, List, Tuple, Dictionary 등의 값이 비어 있지 않으면 참, 비어 있으면 ("", [], (), {}) 거짓.

## 불리언 로직 (Boolean logic)

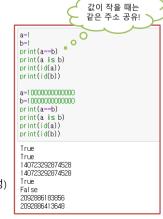
- 불리언 로직
  - 참(true)과 거짓(false)을 판별하는 로직
  - \_ 파이썬에서 참과 거짓
    - 거짓: False, 0, None, [], (), {}
    - 참: 거짓인 경우를 제외한 모든 경우 → -1은 True or False?
  - 불리언 연산자
    - and, or, not, in, all, any

```
A = 1
B = 2
A and B
A or B
not A
1 in [1,2,3]
"b" in "abc"
all([1,1,1])
any([0,1,0])
```



## 불리언 로직 (Boolean logic)

- 비교 연산자
  - -<,>,<=,>=,==,!=, is
  - 예제: 모두 참!
    - [1, 2, 3] < [1, 2, 4]
    - 'C' < 'Pascal' < 'Perl' <'Python'</li>
    - (1, 2, 3, 4) < (1, 2, 4)
    - (1, 2) < (1, 2, -1)
    - (1, 2, 3) == (1.0, 2.0, 3.0)
    - (1, 2, ('aa', 'ab')) < (1, 2, ('abc', 'a'), 4)
  - '=='과 'is'의 차이점
    - ==: 값을 비교
    - is: 주소를 비교 (변수는 주소와 값으로 구성)







### 조건문 (if statement)

- 조건문
  - 조건의 참/거짓에 따라 분기하는 구문
  - \_ 형식

```
if condition:
if condition:
                          if condition:
                                                       do_something
   do something
                             do something
                                                    elif condition2:
                          elif condition2:
                                                       do something else1
if condition: statement
                             do somethina else
                                                       do_something_else2
if condition:
   do something
else:
   do something else
```



### 조건문 (if statement)

- 조건 표현문(conditional expression)
  - 조건에 따라 값을 할당하는 구문
  - 형식: 변수 = 값1 if 조건 else 값2

```
frac = 1/x if x>0 else 0
if 1 < x < 5:
                                        x = int(input("x="))
    print("lower range")
                                        message = "lower range" if 1 < x < 5 else "upper range"
else:
                                        print(message)
    print("out of range")
                                        lower range
if 1 < x < 5:
    print("lower range")
                                       x = int(input("x="))
                                       message = "lower range" if 1 < x < 5 else "upper range" if 5 < x < 10 else "out of range
elif 5 < x < 10:
    print("upper range")
                                       unner range
    print("out of range")
```



### 조건문 (if statement)

```
n1 = float(input("num1="))
x = int(input("x=")
                                         n2 = float(input("num2="))
                                         op = input("operator=")
                                         if(op == '+'):
if 1 < x < 5:
                                            res = n1+n2
                                            print("두 수의 합은",res,"입니다.")
   print("lower range")
                                         elif(op == '-'):
                                            res = n1-n2
                                            print("두 수의 차는",res,"입니다.")
if 1 < x < 5
                                         elif(op == '*'):
   print("lower range")
                                            res = n1*n2
                                            print("두 수의 곱은", res, "입니다.")
                                         elif(op == '/'):
   print("out of range")
                                            res = n1/n2
                                            print("두 수의 나눔은", res, "입니다.")
if 1 < x < 10: print("in range")
                                            print("잘못된 연산자를 입력하셨습니다.")
                                         num1=10
                                         num2=20
                                         operator=/
                                         두 수의 나눔은 0.5 입니다.
```

Edited by Harksoo Kir

#### 반복문 (for statement)

- for문
  - 주어진 리스트에서 값을 하나씩 가져와 참인 동안 특 정 구간을 반복하는 구문
  - \_ 형식

```
for variable in sequence :
statement1
statement2

...

for i in range(10):
    print(i)

for i in range(10,0,-1):
    print(i)

for ch in "mystring":
    print(ch)

for e in ["red","green","blue"]:
    print(e)
```



### 반복문 (for statement)

- 중첩 for문
  - for문 내부에 for문이 존재하는 반복 구문
  - \_ 형식

```
for variable in sequence:
  for variable in sequence:
     statement1
     statement2
```

```
for i in range(2,10):
    print("[{0:d}EH]".format(i))
    for j in range(1,10):
                                                                2*2=4
2*3=6
                                                                2+4=8
          print("{0:d}*{1:d}={2:d}".format(i,j,k))
                                                                 2*5=10
                                                                 2*6=12
2*7=14
                                                                 2*8=16
                                                                 2+9=18
v=[[1,2,3],[4,5,6]]
                               [1, 2, 3]
                                                                 [35]
for i in v:
                                                                 3+1=3
    print(i)
                                                                 3+2=6
    for j in i:
                                                                 3+3=9
                              [4, 5, 6]
         print(j)
                                                                 3+4=12
                                                                 3+5=15
                                                                 3+6=18
                                                                 3+7=21
```

[2단] 2±1=2

3+8=24

3+9=27



## 반복문 (for statement)

- continue와 break
  - continue: 반복문으로 이동하는 구문
  - break: 반복문을 빠져나가는 구문

```
for i in range(10):
  if 2<i<5:
      continue
   print(i)
for ch in "this is a string":
   if ch == ' ':
      break
   print(ch)
```



### 반복문 (for statement)

- One-line for문
  - for문을 1개 라인으로 축약해서 표현하는 구문
  - \_ 형식

```
[variable for variable in sequence]
```

```
v = list(range(10))
print(v)
for i in v:
   print(i)
print([i for i in v])
v=[[1,2,3],[4,5,6]]
for i in v:
   for j in i:
       print(j)
print([i for i in v for i in i])
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[1, 2, 3, 4, 5, 6]
```

## 반복문 (for statement)

• 예제

```
i = 0
                                      for i,ch in enumerate("mystring"):
for ch in "mystring":
                                          print(i,ch)
    print(i,ch)
    i = i + 1
nums = [1,2,3,4,5]
                                                                           3 t
for i in nums:
                                 Don't modify list while
   if i==3: del nums[3]
                                                                          5 i
                                 iterating over it!
                                                                           6 n
 print(i)
```



### 반복문 (while statement)

- while문
  - 주어진 조건이 참인 동안 특정 구간을 반복하는 구문
  - \_ 형식

```
while condition:
                          i = 0
                                                  i = 0
   statement1
                          while i < 10:
                                                  while 1:
                             print(i)
                                                     i += 1
   statement2
                                                     if i < 5:
                             i += 1
                                                         continue
                          i = 0
                                                     print(i)
                          while 1:
                             print(i)
                             i += 1
                             if i > = 10:
                                break
```



### 실습

- 국어, 영어, 수학 점수를 입력 받아서 평균을 계산하고 평균에 따라 등급을 부여하는 프로그램을 작성 하시오.
  - \_ 실행 예

```
국어= 90
영어= 85
수학 = 72
평균은 82.3이며, 등급은 B입니다.
```

?

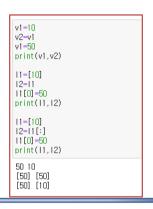
## 참조 (references)

- 참조 할당
  - 리스트와 딕셔너리 할당은 값이 아니라 주소를 할당

```
# content copy
v1 = 10
v2 = v1
v1 = 50

# address copy in list
I1 = [10]
I2 = I1
I1[0] = 50

# content copy in list
I1 = [10]
I2 = I1[:]
I1[0] = 50
```





Edited by Harksoo Kir

### 실습

- 라인 수를 입력 받아서 아래와 같은 모양을 출력하는 프로그램을 작성 하시오.
  - 실행 예







## 실습

- 반복문과 조건문을 사용하여 입력된 정수들 중에 가장 큰 값을 출력하는 프로그램을 작성하시오.
  - \_ 실행 예

Numbers? 10 456 87 78 Max = 456

?



# 질의응답



Homepage: http://nlp.konkuk.ac.kr E-mail: nlpdrkim@konkuk.ac.kr

