파이썬 프로그래밍

파이썬의 각종 연산자

한국기술교육대학교
온라인평생교육원

▣ 논리 연산자

• 피연산자의 값으로 진리값인 True 또는 False을 취하여 논리 적인 계산을 수행하는 연산자
 - and
 - or
 - not

•[주의] 논리 연산자 자체가 값을 반환하지는 않는다.
 - 논리 연산을 따라 최종적으로 평가(Evaluation)되어진 값이 반환된다.

```
a = 20
b = 30
print a > 10 and b < 50
```

```
True
```

- and, or, not → 값을 절대 반환하지 않음
- a>10 → true
- 앞 and 뒤: 앞도 true 뒤도 true여야 true
- 앞 and 뒤: 앞이 true고 뒤는 false면 false
- and가 리턴한 것이 아니라 b=50을 확인한 결과가 리턴
- 50 >b가 true이기 때문에 true값이 리턴된 것
- or 연산자는 evaluate 하지 않음
- 앞 or 뒤: 앞이 true 이면 뒤쪽은 볼 것도 없이 true

▣ 논리 연산자

• 진리값에 해당하는 True와 False는 다른 사칙연산자를 만나면 다음과 같이 평가됨
- True: 1
- False: 0

```
print True + 1
print False + 1
print False * 75
print True * 75
```

```
2
1
0
75
```

- true가 1이니까 1을 출력
- false가 0 → 0+1 =1이 출력
- false가 0 → 0*75 = 0이 출력
- true가 1 → 1* 75 = 75 출력

▣ 논리 연산자

• bool() 내장 함수를 이용해서 수치 값을 진리 값으로 교환 가능

```
print bool(0) # 정수 0은 거짓
print bool(1)
print bool(100)
print bool(-100)
print
print bool(0.0) # 실수 0.0은 거짓
print bool(0.1)
```

```
False
True
True
True

False
True
```

- bool(불): 하나 받는 객체를 evaluate하여 true 또는 false로 반환
- 객체는 대부분 true지만 false로 변환되는 객체들이 존재
- 정수값 0은 bool(불) 내장함수에서 false로 evaluate 됨
- 0.0이 아닌 나머지 모든 실수 = true

▣ 논리 연산자

• 값이 없는 빈 객체나 None 객체는 False로 평가됨

```
print bool('abc')
print bool('')
print
print bool([]) # 공 리스트는 거짓
print bool([1,2,3])
print
print bool(()) # 공 튜플은 거짓
print bool((1,2,3))
print
print bool({}) # 공 사전은 거짓
print bool({1:2})
print
print bool(None) # None 객체는 거짓
```

```
True
False

False
True

False
True

False
True

False
```

- 일반적인 문자열(ex. abc) → true
- 공백조차 없는 비어있는 문자열 → false
- 비어있는 tuple, 비어있는 list, 비어있는 사전 → false
- 내용이 하나라도 있으면 → true
- none 객체 → false

▣ 논리 연산자

```
print 1 and 1
print 1 and 0
print 0 or 0
print 1 or 0
print
print [] or 1 # [] 거짓
print [] or () # [], () 거짓
print [] and 1 # [] 거짓이므로 1은 참조할 필요 없음
```

```
1
0
0
1

1
()
[]
```

- 앞 and 뒤 = 앞, 뒤 모두 evaluate한 결과
- 앞 or 뒤: 앞이 false면 뒤쪽도 확인하여 false면 false
- 1 or 0 = 앞이 true 이므로 뒤 확인 생략 = 1 출력
- 비어있는 tuple = false
- false or false = false
- 앞 and 뒤: 앞이 false면 뒤도 안보고 무조건 false

▣ 논리 연산자

```
print 1 and 2
print 1 or 2
print
print [[]] or 1 # [[]] 참으로 간주
print [{}] or 1 # [{}] 참으로 간주
print '' or 1 # 빈 문자열('')은 거짓
```

```
2
1

[[]]
[{}]
1
```

- [[ ]]: 공 list를 가지고 있는 list = 바깥 list는 공list가 아님 = true
- [{ }]: 공 사전을 가지고 있는 list = true
- or 연산자가 true를 반환하지 않고 뒤를 evaluate 한 결과 출력

```
print not(True)
print not(1 and 2)
print not('' or 1)
```

```
False
False
False
```

- not 뒤: 뒤가 false면 true, true면 false로 바꿔 출력
- 연산자 우선순위에 의존보다 적절한 괄호 활용이 필요