데이터 연산과 논리적 판단

오 동 인
경희대학교 의과대학 의공학교실

수식과 연산자

- 수식(expression)
  - 피연산자들과 연산자의 조합으로 구성
  - 피연산자(operand) : 연산의 대상이 되는 것
  - 연산자(operator)
    - 어떤 연산을 나타내는 기호
    - 산술 연산자, 관계 연산자, 논리 연산자, 비트 연산자 등
  - 수식의 연산에 의해 결과 값이 생성됨

- 사칙연산
  - 덧셈, 뺄셈, 곱셈, 나눗셈 연산자

연산	연산자	수식	결과
덧셈	+	6 + 4	10
뺄셈	-	6 - 4	2
곱셈	*	6 * 4	24
나눗셈	/	6 / 4	1.5

- Ex) 사용자로부터 2개의 정수를 입력 받아 각각 변수 x와 y에 대입하고, print()함수를 이용하여 사칙연산의 결과를 출력

```
x = int(input('정수1 : '))
y = int(input('정수2 : '))

print(x+y)
print(x-y)
print(x*y)
print(x/y)
```

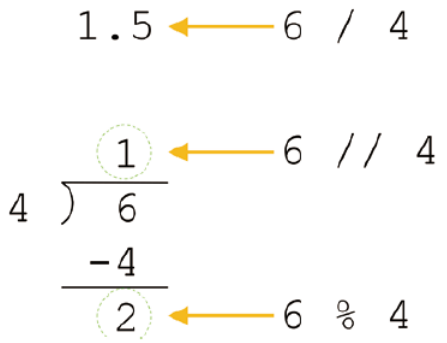
- Ex) 나이 계산

```
name = input('이름 : ')
birth_year = int(input('출생년도 : '))
age = 2021 - birth_year + 1

print('이름 : ', name)
print('출생년도 : ', birth_year)
print('나이 : ', age)
```

정수 나눗셈과 나머지 계산

- 실수 나눗셈
  - / 연산자에 의한 나눗셈 연산은 항상 실수 연산 가정

연산	연산자	수식	결과
나눗셈(실수)	/	6 / 4	1.5
나눗셈(정수)	//	6 / 4	1
나머지	%	6 / 4	2



- 정수 나눗셈과 나머지 계산
  - 나눗셈의 의한 정수 결과(몫)를 구할 경우 // 연산자 사용
  - 나눗셈의 나머지 값을 구할 경우 % 연산자를 사용

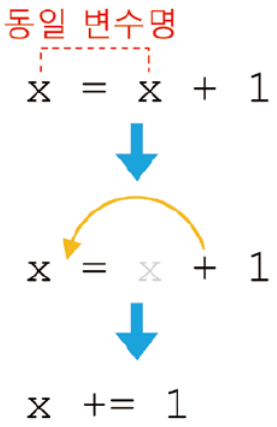
- Ex) 10원 이상의 거스름돈에 해당하는 정수를 입력 받아 줘야하는 500원, 100원, 50원, 10원 동전의 개수를 구함.

```
x = int(input('거스름돈 금액 : '))
x500 = x // 500
temp = x % 500
x100 = temp // 100
temp = temp % 100
x50 = temp // 50
temp = temp % 50
x10 = temp // 10
```

```
print("거스름돈 :", x, '원', "500원 :", x500, "100원 :", x100, "50원 :", x50, "10원 :", x10)
```

복합 대입 연산자

- 복합 대입 연산자 (compound assignment operator)
  - 다른 연산자와 대입 연산자를 결합시켜 놓은 연산자
  - x = x + 1과 같은 문장을 복합 대입 연산자를 사용하여 x += 1로 간략히 작성 가능
  - 일반적으로 '복합 대입 연산자'로 사용
  - '확장(augmented) 대입 연산자' 라는 용어로도 사용




```
x = y = z = 1    % multiple assignment
x += 1           % compound assignment
```

복합 대입 연산자	문장	의미
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
//=	x //= y	x = x // y
%=	x %= y	x = x % y

복합 대입 연산자 예제

- Ex) 정수를 입력 받아 변수 x에 대입 후, 사칙연산을 위한 복합 대입 연산자를 활용하여 변수 x에 각각 2를 계산하고 출력

```
x = int(input('정수 : '))
print(x)
x += 2
print(x)
x -= 2
print(x)
x *= 2
print(x)
x /= 2
print(x)
```

- Ex) 복합 대입 연산자 비교

```
>>> a = 1
>>> b = 2
>>> c = 3
>>> c *= a - b
>>> print(a, b, c)
1 2 -3
```

```
>>> a = 1
>>> b = 2
>>> c = 3
>>> c = c * (a - b)
>>> print(a, b, c)
1 2 -3
```

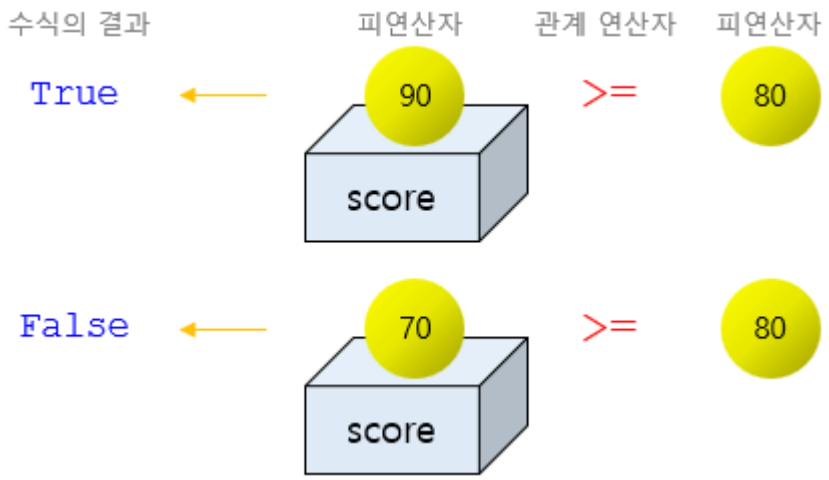
```
>>> a = 1
>>> b = 2
>>> c = 3
>>> c = c * a - b
>>> print(a, b, c)
1 2 1
```

```
a = 1
b = 2
c = 3
a = a + b
b = b + a - c
c = c * (a - b)
print(a, b, c)
```

```
a = 1
b = 2
c = 3
a += b
b += (a - c)
c *= (a - b)
print(a, b, c)
```

관계연산자 (Relational Operator)

- 두 개의 피연산자를 비교
- 관계 연산자 수식의 결과는 참(True)/거짓(False)
  - '점수가 80 이상인' 문장 'score >= 80' 조건 수식으로 나타냄
  - 변수 score의 값이 90인 경우 수식의 결과는 참(True)
  - 변수 score의 값이 70인 경우 수식의 결과는 거짓(False)



관계 연산자	의미	결과 (x:6, y:2 인 경우)
x > y	x가 y보다 큰가?	True
x >= y	x가 y보다 크거나 같은가?	True
x < y	x가 y보다 작은가?	False
x <= y	x가 y보다 작거나 같은가?	False
x == y	x와 y가 같은가?	False
x != y	x와 y가 다른가?	True

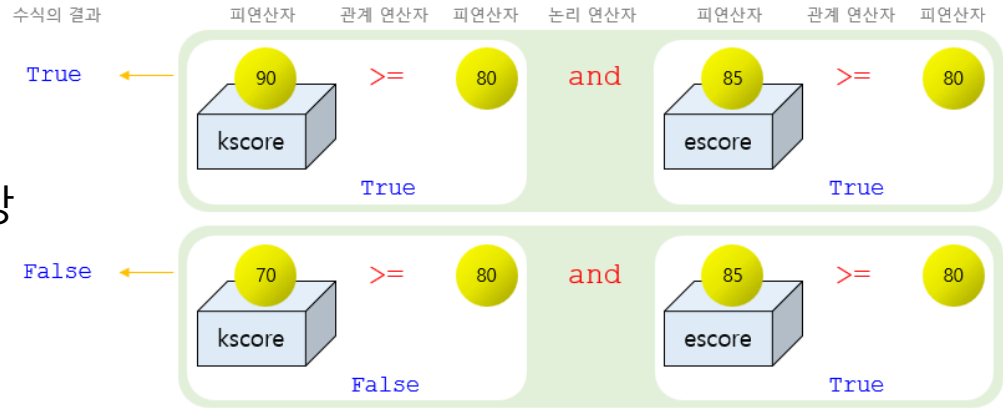
관계연산자 실습

```
x = 6
y = 2
print(x, '>', y, x > y)
print(x, '>=', y, x >= y)
print(x, '<', y, x < y)
print(x, '<=', y, x <= y)
print(x, '==', y, x == y)
print(x, '!=', y, x != y)


x = y
print('x = ', x, 'y = ', y)
```

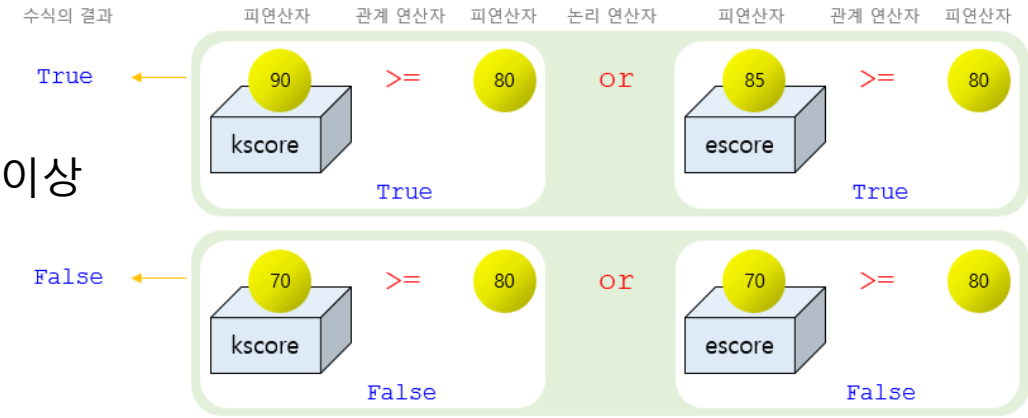
```
# 2는 1+1의 값 보다 크다.
    print(2 > 1+1)
# 7/3의 몫은 2와 같다.
    print(7//3 == 2)
# 1+2+3의 합은 6보다 작거나 같다.
    print(1+2+3 <= 6)
# 6은 짝수이다.
    print(6%2 == 0)
```

논리연산자 (Logical Operator)

- 여러 개의 조건을 조합하여 참인지, 거짓인지 판별
- and 연산자
  - 두 조건 모두 참일 때 전체가 참
  - 두 조건 중 하나라도 거짓일 때 전체가 거짓
  - 국어 점수가 80점 이상이고 영어 점수가 80점 이상
  - 'kscore >= 80' 조건과 'escore >= 80‘
  - 'kscore >= 80 and escore >= 80‘



- or 연산자
  - 두 조건 중 하나라도 참일 때 전체가 참
  - 두 조건 모두가 거짓일 때는 전체가 거짓
  - 국어 점수가 80점 이상이거나 영어 점수가 80점 이상
  - 'kscore >= 80 ' 조건 또는 'escore >= 80‘
  - 'kscore >= 80 or escore >= 80'



논리연산자

논리 연산자	의미
x and y	x와 y가 모두 True이면 True, 그렇지 않으면 False
x or y	x나 y중에서 하나만 True이면 True, 모두 False이면 False
not x	x가 True이면 False, x가 False이면 True

- 진리표(Truth table)
  - 진리식/논리식/논리회로에 대한 입출력 결과를 기록한 표

x	y	x and y	x or y	not x
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

논리연산자 실습

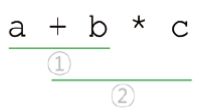
```
k = 90
e = 85
print(k >= 80 and e >= 80)
print(k >= 80 or e >= 80)

k, e = map(int,input("두개의 숫자를 입력하세요: ").split())
print(k >= 80 and e >= 80)

k, e = map(int,input("두개의 숫자를 입력하세요: ").split())
print(k >= 80 or e >= 80)
```

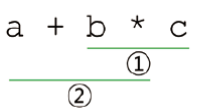
```
# 3은 1보다 크고 5보다 작다.
    print(3>1 and 3<5)
# 3은 1보다 크거나 5보다 작다.
    print(3>1 or 3<5)
# 85점은 80점 이상이고 90점 미만이다.
    print(85>=80 and 85 <90)
# 85는 80부터 89까지 사이의 숫자이다.
    print(85>=80 and 85<=89)
```

연산자 우선순위

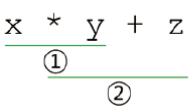
- 연산의 우선 순위
  - 수식에 2개 이상의 연산자가 사용될 때 어느 연산자를 먼저 평가하여 계산할지 결정을 해야 함

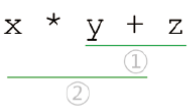
```
a + b * c
```





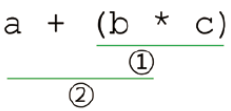

```
x * y + z
```



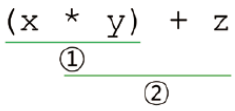


  - 괄호 사용시 계산의 순서를 보다 더 명확히 할 수 있음

```
a + (b * c)
```

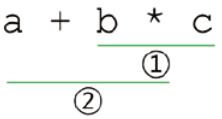


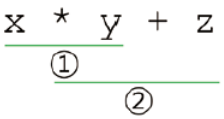

```
(x * y) + z
```

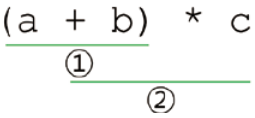


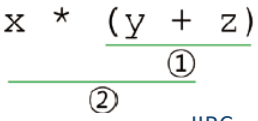
- 연산자 우선순위의 변경
  - 괄호를 사용하여 연산자 우선순위 변경 가능

```
a + b * c     # a + (b * c)
x * y + z     # (x * y) + z

(a + b) * c
x * (y + z)
```









연산자 우선순위

- 산술연산의 경우 기본적으로 수학적 관례를 따름. 첫 글자를 따서 PEMDAS로 기억
  - **괄호(Parentheses)**는 가장 높은 우선순위를 가지며, 괄호 내의 식이 먼저 실행됨
  - **지수승(Exponentiation)**은 다음으로 높은 우선순위를 가짐
  - **곱셈(Multiplication)**과 **나눗셈(Division)**은 동일한 우선순위를 가짐
  - **덧셈(Addition)**과 **뺄셈(Subtraction)**은 동일한 우선순위를 가짐
  - 같은 우선순위를 갖는 연산자는 왼쪽에서 오른쪽 순서로 실행됨

순위	연산자	설명	순위	연산자	설명
1	**	지수 연산	8	< > <= >=	비교 연산
2	~ + -	비트 반전, +부호, -부호	9	== !=	동등 연산
3	* / // %	곱셈, 실수나눗셈, 정수나눗셈, 나머지	10	is, is not	아이덴티티 연산
4	+ -	덧셈, 뺄셈	11	in, not in	소속 연산
5	<< >>	왼쪽 비트 이동, 오른쪽 비트 이동	12	not	논리 부정
6	&	비트 AND	13	and, or	논리 AND, 논리 OR
7	^ |	비트 XOR, 비트 OR	14	= += -= *= /= //= %= **=	대입 연산

연산자 우선순위 고려 계산

- Ex) 두 정수를 입력 받아 평균값을 구해서 출력

```
x = int(input('값1 : '))
y = int(input('값2 : '))
z = (x + y) / 2
print("평균 :", z)
z1 = x + y / 2
print("평균 :", z1)
```

EOD