

[Python]



Python으로 배우는 소프트웨어 원리

Chapter 04. 연산

목차

1. 추상화
2. 연산자

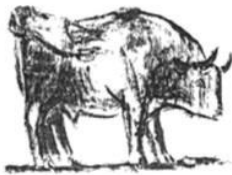
01

추상화

01. 추상화

[피카소의 추상화]

- 황소를 단순화하는 과정이다.
- 단순한 10개의 선으로 표현했다.



Versão 1



Versão 2



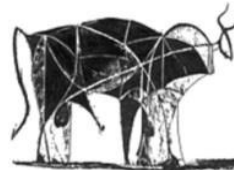
Versão 3



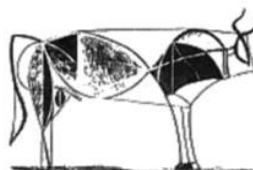
Versão 4



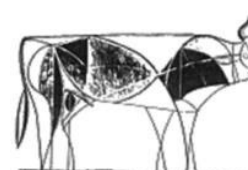
Versão 5



Versão 6



Versão 7



Versão 8



Versão 9



Versão 10



Versão 11

01. 추상화

[추상화의 개념]

- 일상에서의 추상화



01. 추상화

I. 추상화의 개념

- 추상화(abstract)란 복잡한 대상이나 상황을 간결하고 단순화하여 표현하는 과정
- 제어 추상화
 - 순차, 선택, 반복과 같은 프로그램의 제어구조를 나타낼 때 복잡한 과정을 생략하고 단순하게 표현하는 방법

```
while True :  
    age = int(input("<나이 입력> "))  
    print("아~ %d살..." % age)  
    myage = 30  
    print("나하고 %d살 차이가 나네요." % abs(myage - age))  
  
    if myage > age :  
        print("그럼 내가 위네요.")  
    elif myage < age :  
        print("그럼 당신이 위네요.")  
    else :  
        print("그럼 동갑이네요.")  
  
    yn = input(">> 그만 하려면 x를 입력? ")  
    if yn == 'x' or yn == 'X' :  
        break
```

01. 추상화

I. 추상화의 개념

- 제어 추상화

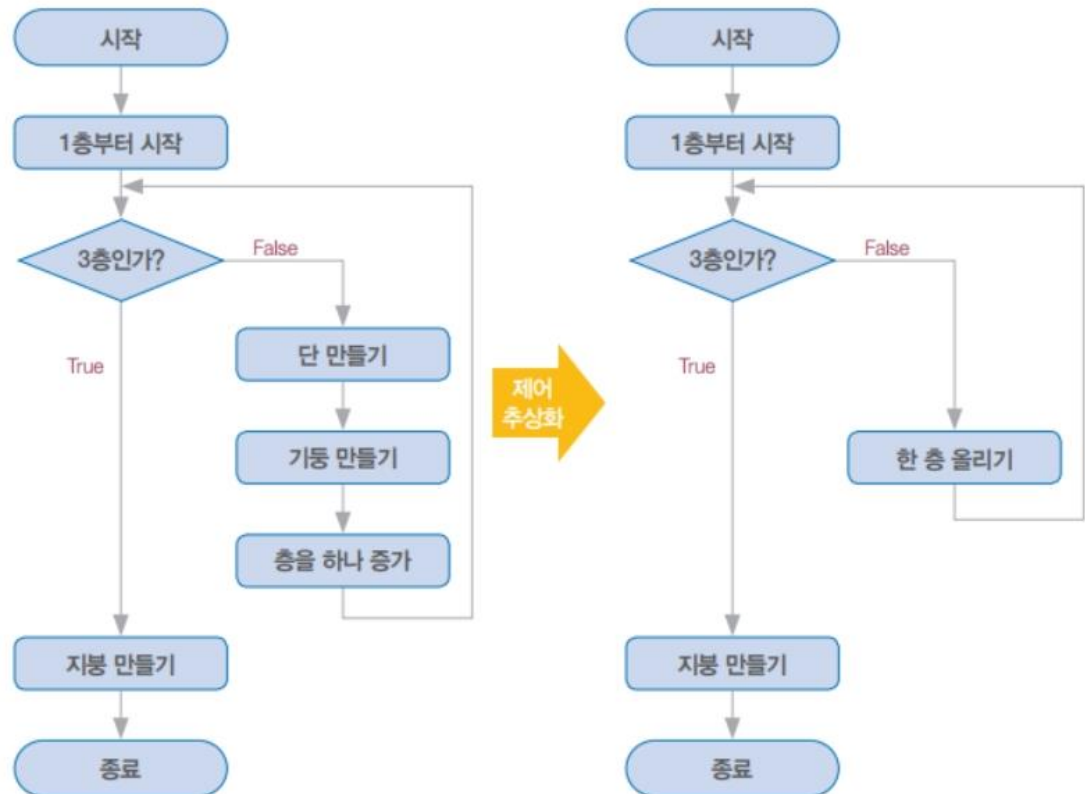


그림 4-2 순서도에서의 제어 추상화

- 오른쪽 알고리즘은 '한 층 올리기'라는 명령으로 간단하게 알고리즘을 표현

01. 추상화

I. 추상화의 개념

- 제어 추상화 : 함수 호출로 추상화

```
def compare_age(myage, age) :  
    if myage > age :  
        print("그럼 내가 위네요.")  
    elif myage < age :  
        print("그럼 당신이 위네요.")  
    else :  
        print("그럼 동갑이네요.")
```

```
while True :  
    age = int(input("<나이 입력> "))  
    print("아~ %d살..." % age)  
    myage = 30  
    print("나하고 %d살 차이가 나네요." % abs(myage - age))  
    ...  
    if myage > age :  
        print("그럼 내가 위네요.")  
    elif myage < age :  
        print("그럼 당신이 위네요.")  
    else :  
        print("그럼 동갑이네요.")  
    ...  
    compare_age(myage, age)  
  
    yn = input(">> 그만 하려면 x를 입력? ")  
    if yn == 'x' or yn == 'X' :  
        break
```


01. 추상화

I. 추상화의 개념

■ 데이터 추상화

- 문제해결을 위해 필요한 데이터를 파악하고 중요한 정보 요소만 구조화시키는 방법

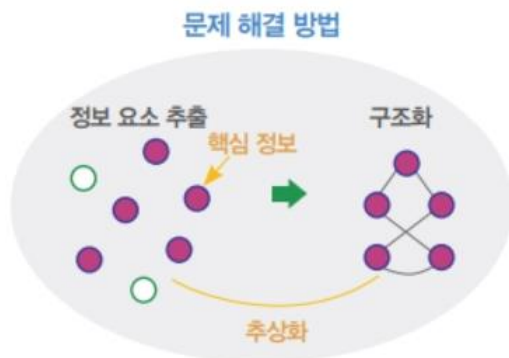


그림 4-3 데이터 추상화 © 김현철, 『정보적 사고에서 인공지능까지』, 한빛아카데미, 2019

```
01 members = (('choi', 93), ('han', 50), ('jung', 92), ('kang', 68), ('kim', 80),
02           ('lee', 90), ('moon', 65), ('na', 100), ('park', 75), ('song', 75))
03 search = input("검색할 아이디 입력 : ")
04 if search in (x[0] for x in members):
05     print("가입한 회원입니다.")
06 else :
07     print("회원이 아닙니다.")
```

아이디만 추출해서 목록을 생성

01. 추상화

II. 추상화 과정

■ 정보의 추출



그림 4-4 수집된 고객 데이터

- 수집된 고객 데이터 중 주민등록번호는 고객의 나이와 생일, 성별을 파악할 수 있는 자료가 되고, 우편번호와 주소를 통해 고객이 거주하고 있는 지역을 분석
- 고객 관리 문제에 고객의 나이와 생일, 성별, 지역, 도시를 활용
- 이를 위해서 주민등록번호, 우편번호, 주소를 중요한 정보 요소로 추출

01. 추상화

II. 추상화 과정

■ 정보의 연결

- 그래프나 표를 사용하면 정보의 의미와 관계를 더 효과적으로 나타낼 수 있음

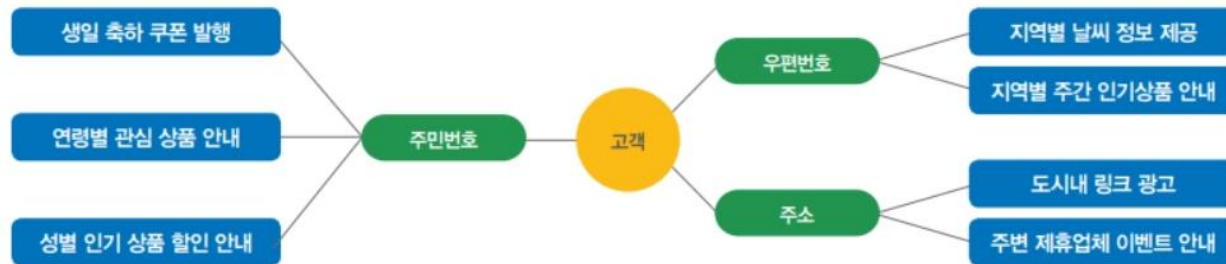


그림 4-5 그래프로 표현한 정보의 구조화

표 11-1 tkinter 모듈의 위젯 종류

위젯명	설명
Button	클릭 이벤트를 만들 수 있는 버튼
Canvas	선과 점으로 그림을 그리는 데 사용
Checkbutton	여러 개의 항목을 동시에 선택할 수 있는 옵션에 사용
Entry	한 줄로 텍스트를 입력하는 입력 상자
Frame	기본 윈도우 하위에서 위젯을 묶을 수 있는 컨테이너
Label	텍스트나 이미지를 표시할 때 사용

01. 추상화

II. 추상화 과정

■ 정보의 연결

실습 4-1

컴퓨터를 구매하기 위한 정보 추상화하기

- 컴퓨터의 많은 장치를 기능별로 분류하고 단순화하여 표현
- 컴퓨터 구매를 위한 정보 추상화하는 실습

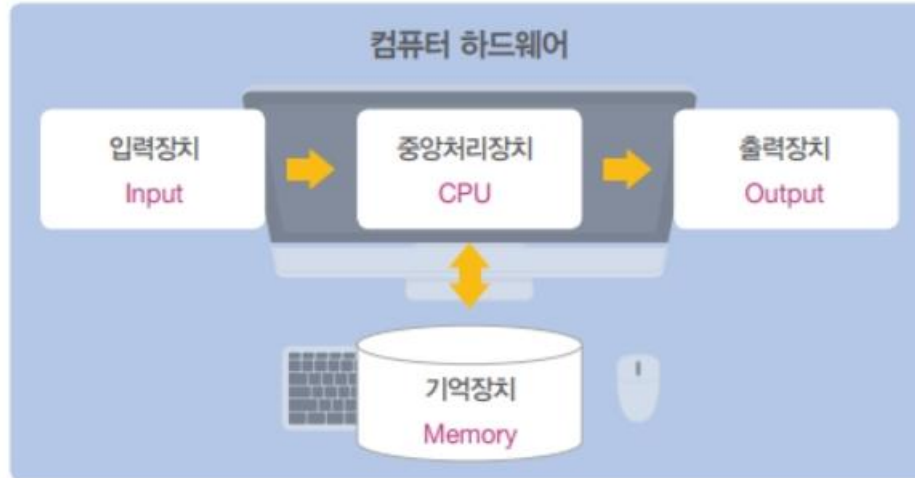


그림 4-6 컴퓨터 하드웨어의 구성

01. 추상화

II. 추상화 과정

■ 정보의 연결

실습 4-1

컴퓨터를 구매하기 위한 정보 추상화하기

- ① 컴퓨터를 구매할 때 고려해야 할 정보를 먼저 수집
- ② 수집한 장치 정보 중에서 주요 정보만 추출 → 컴퓨터의 주요 장치인 CPU, 메모리, 메인보드, 케이스, 파워를 추출해 다른 색상으로 표시

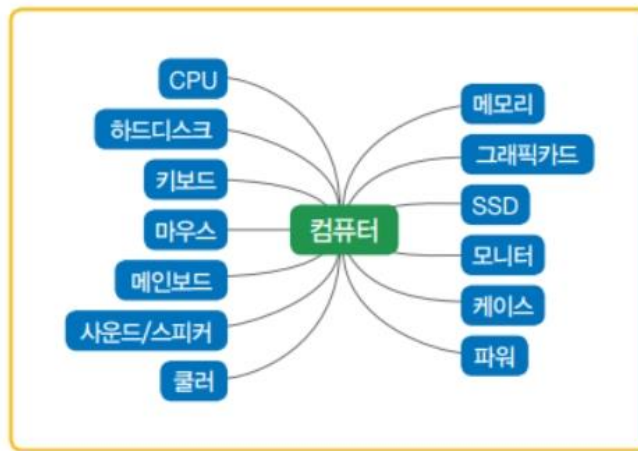


그림 4-7 컴퓨터 구매에 고려할 장치 정보

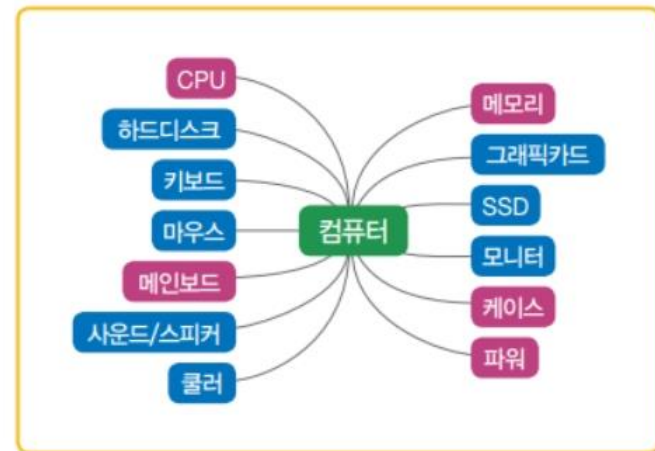


그림 4-8 주요 장치 정보의 추출

01. 추상화

II. 추상화 과정

■ 정보의 연결

실습 4-2

정다각형 그리기 추상화하기

- ① 정 n 각형의 한 내각의 크기와 외각의 크기는 다음의 수식 이용

$$\text{내각의 크기} = 180^\circ * (n-2) / n$$

$$\text{외각의 크기} = 360^\circ / n$$

- ② 이동 거리와 회전 각도를 알려주면 로봇이 움직이는 경로를 따라 선을 그림

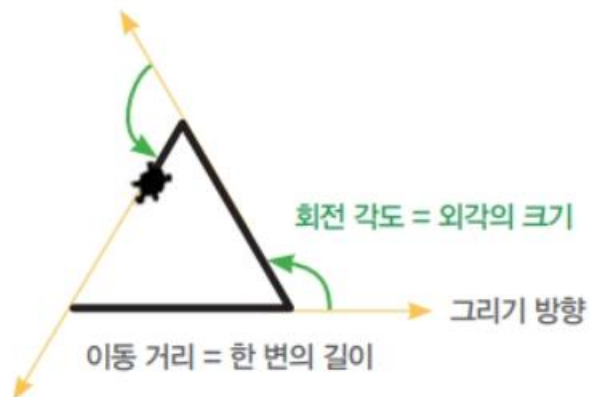


그림 4-10 정삼각형을 그릴 때 로봇의 이동과 회전

01. 추상화

II. 추상화 과정

















■ 정보의 연결

실습 4-2

정다각형 그리기 추상화하기

- ③ 한 변의 길이만큼 이동한 후 외각의 크기만큼 회전하는 동작을 꼭짓점의 개수(n)만큼 반복

표 4-2 정 n 각형 그리기의 이동과 회전

정 n 각형(n =꼭짓점 개수)		단계 1	단계 2	...	단계 n
정삼각형					
정사각형					
정오각형					
정육각형					

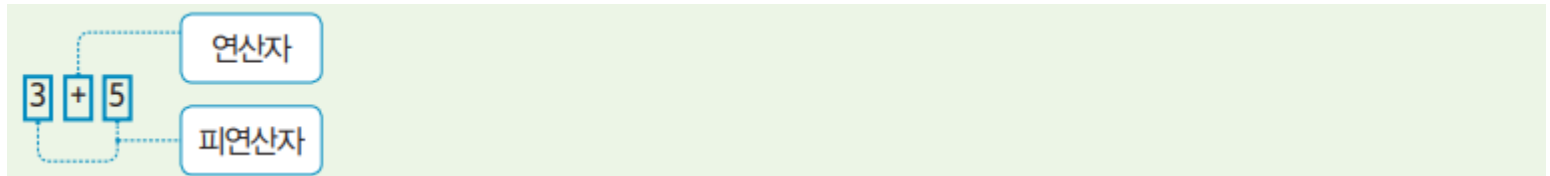
02

연산자

02. 연산자

I. 연산자의 종류

- 연산자와 피연산자로 이루어지는 계산식을 수식이라고 부름



- 파이썬 프로그래밍에는 사칙연산을 비롯한 다양한 산술 연산자와 비교 연산자, 논리 연산자, 대입 연산자 등을 사용

표 4-3 다양한 연산자 종류

산술 연산자	+ - * / // % **
비교 연산자	< > <= >= == !=
논리 연산자	and or not
대입 연산자	= += -= *= /= //= %=

02. 연산자

I. 연산자의 종류

■ 산술 연산자

- 산술 연산자에는 덧셈(+), 뺄셈(-), 곱셈(*), 나눗셈(/)의 사칙연산 외에도 몫 (//)과 나머지(%)를 구하거나 제곱을 구하는 지수(**) 연산도 포함

표 4-4 산술 연산자의 종류

연산자	예시	의미	실행 결과
+	3 + 5	3과 5의 더하기	8
-	3 - 5	3에서 5를 빼기	-2
*	3 * 5	3과 5의 곱하기	15
/	3 / 5	3을 5로 나누기	0.6
//	3 // 5	3을 5로 나눈 몫	0
%	3 % 5	3을 5로 나눈 나머지	3
**	3 ** 5	3의 5제곱	243

✓ **TIP** 산술 연산자 사이에서는 지수(**), 곱셈·나눗셈·몫·나머지(*, /, //, %), 덧셈·뺄셈(+, -) 연산자의 순서로 우선순위가 정해짐

02. 연산자

I. 연산자의 종류

- 산술 연산자

실습 4-3

몫과 나머지 계산하기

- ① 나뉘지는 수와 나누는 수를 정수형으로 입력받아 변수 x와 y에 각각 저장

```
>>> x = int(input("나뉘지는 수 : "))  
나뉘지는 수 : 3000  
>>> y = int(input("나누는 수 : "))  
나누는 수 : 500
```

- ② 몫과 나머지 계산 수식을 실행한 결과를 화면에 출력

```
>>> print("%d를 %d로 나눈 몫 = %d" %(x, y, x//y))  
3000를 500로 나눈 몫 = 6  
>>> print("%d를 %d로 나눈 나머지 = %d" %(x, y, x%y))  
3000를 500로 나눈 나머지 = 0
```

02. 연산자

I. 연산자의 종류

- 산술 연산자

실습 4-4

원의 면적 구하기

- 원의 면적은 $s = \pi * r * r$

- ① 원의 반지름을 실수형으로 입력받아 변수 radius에 저장

```
>>> radius = float(input("원의 반지름 : "))  
원의 반지름 : 3.5
```

- ② 공식에 맞게 수식을 작성하고 계산한 원의 면적을 변수 area에 저장

```
>>> area = 3.14 * radius ** 2
```

- ③ 원의 면적 area를 실수형 포맷 코드를 사용하여 화면에 출력

```
>>> print("원의 면적 = %f" % area)  
원의 면적 = 38.465000
```

02. 연산자

I. 연산자의 종류

■ 비교 연산자

- 2개의 피연산자를 비교하여 참이나 거짓으로 판별하는 수식(조건식)에 사용
 - 연산 결과는 문자열로 'True' 또는 'False' 중에 하나로 표시
 - 연산 결과는 숫자로 1(True) 또는 0(False) 중에 하나로 표시

표 4-5 비교 연산자의 종류

연산자	예시	의미	실행 결과
<	3 < 5	3은 5보다 작다	True
>	3 > 5	3은 5보다 크다	False
<=	3 <= 5	3은 5보다 작거나 같다	True
>=	3 >= 5	3은 5보다 크거나 같다	False
==	3 == 5	3은 5와 같다	False
!=	3 != 5	3은 5와 다르다	True

✓ **TIP** 비교 연산자는 **관계 연산자**라고도 부름

02. 연산자

I. 연산자의 종류

■ 비교 연산자

실습 4-5

나이를 기준으로 영화 매표 여부 확인하기

- 영화관에서 17세 이상 관람 가능한 영화를 판매한다고 가정

- ① 나이를 입력받고 변수 age에 저장

```
>>> age = int(input("나이를 입력하세요 : "))  
나이를 입력하세요 : 21
```

- ② 비교 연산자를 이용하여 관람 가능한 나이인지 판단해서 출력

```
>>> print("매표 가능 여부 : ", age >= 17)  
매표 가능 여부 : True
```

실습 4-6

입력한 정수가 3의 배수인지 확인하기

- 정수를 하나 입력받고 그 수가 3의 배수이면 'True'를 출력하는 프로그램을 작성

02. 연산자

I. 연산자의 종류

■ 비교 연산자

- ① 정수를 하나 입력받아 변수 x에 저장

```
>>> x = int(input("정수를 입력하세요 : "))  
정수를 입력하세요 : 12345678
```

- ② 입력한 정수 x를 3으로 나누어 나머지를 변수 remain에 저장

```
>>> remain = x % 3
```

- ③ 변수 remain에 저장된 값이 0인지 비교 연산자를 이용해서 확인 후 출력

```
>>> print("x는 3의 배수입니까? : ", remain == 0)  
x는 3의 배수입니까? : True
```

✓ **TIP** '=='은 '같다'를 의미하는 비교 연산자이고, '='는 변수에 값을 저장하는 대입 연산

02. 연산자

I. 연산자의 종류

■ 논리 연산자

- 2개 이상의 조건이 결합되어 전체 식의 True와 False를 판단하는 형태

표 4-6 논리 연산자의 종류

연산자	예시	예시의 조건	실행 결과
and	x and y	x와 y가 모두 True인 경우	True
		x와 y가 하나 이상 False인 경우	False
or	x or y	x와 y가 하나 이상 True인 경우	True
		x와 y가 모두 False인 경우	False
not	not x	x가 True인 경우	False
		x가 False인 경우	True

x	y	result
0	0	
0	1	
1	0	
1	1	

02. 연산자

I. 연산자의 종류

- 논리 연산자

실습 4-7

학점 계산하기

- 점수가 60점 미만이거나 결석이 4회 이상이면 F 학점으로 판정하는 프로그램

① 점수와 결석 횟수를 입력받아 변수에 저장

```
>>> score = float(input("점수 입력 : "))
점수 입력 : 85.7
>>> absence = int(input("결석 횟수 입력 : "))
결석 횟수 입력 : 1
```

② 논리 연산자를 활용한 수식으로 F 학점 여부를 판단해서 출력

```
>>> print("F 학점 여부 = ", (score < 60) or (absence >= 4))
F 학점 여부 = False
```

02. 연산자

I. 연산자의 종류

■ 논리 연산자

실습 4-8

아이디와 비밀번호로 로그인하기

- 아이디와 비밀번호를 입력하고 로그인할 수 있는지 알려주는 프로그램

- ① 가입할 때 사용한 아이디와 비밀번호를 임의로 정해서 변수에 저장

```
>>> id = "hong"
>>> password = "1234"
```

- ② 로그인하려는 아이디와 비밀번호를 입력하고 변수에 각각 저장

```
>>> input_id = input("아이디를 입력하세요 : ")
아이디를 입력하세요 : hong
>>> input_pass = input("비밀번호를 입력하세요 : ")
비밀번호를 입력하세요 : 12345
```

- ③ 아이디와 비밀번호가 모두 일치하는지 판단해서 로그인 여부를 출력

```
>>> print("로그인 여부 : ", id == input_id and password == input_pass)
로그인 여부 : False
```

02. 변수의 개념과 활용

[문제]

1. 아래 코드의 실행 결과를 우측에 적으시오.

```
➤ a = 10  
➤ b = -12
```

- ① `print(a == b)` _____
- ② `print("%s" %(a != b))` _____
- ③ `print("%d" %(a <= b))` _____

2. 아래 코드 이후에 실행될 각 항의 코드 실행 결과를 우측에 적으시오.

```
➤ x = 3  
➤ y = -3.0
```

- ① `print((x == y) or (abs(x) == abs(y)))` _____
- ② `print((x == y) and (abs(x) == abs(y)))` _____
- ③ `print((x > 0) and (y < 0))` _____

02. 연산자

I. 연산자의 종류

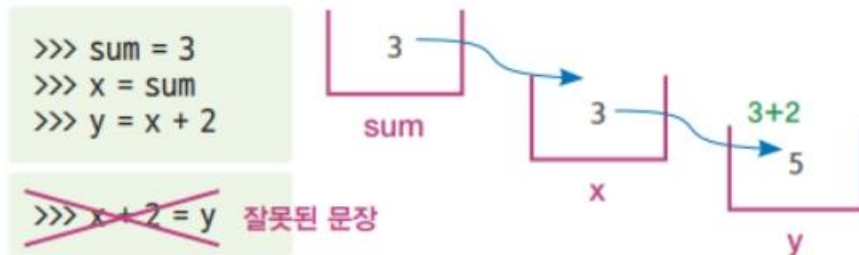
■ 대입 연산자

- 대입 연산자의 오른쪽에는 저장할 값이나 수식 혹은 다른 변수 이름을 사용
- 대입 연산자의 오른쪽에 수식이 있는 경우, 수식의 계산 결과가 변수에 저장
- 대입 연산자의 왼쪽에는 값이나 수식을 사용하면 안 되고 반드시 하나의 변수 이름만 사용

>>> 변수명 = 저장할 값

오른쪽에 있는 값이나
수식의 결과를 왼쪽 변수에 저장

그림 4-12 대입 연산자의 규칙



02. 연산자

I. 연산자의 종류

■ 대입 연산자

- 대입 연산자의 오른쪽에 다른 변수명이 오면 값이 복사되는데, 그림과 같이 하나의 수식에 대입 연산자를 여러 번 사용 가능
- 다음 수식에서는 변수 c에 10을 먼저 저장하고, 그 값이 다른 변수 b와 a에 차례대로 복사

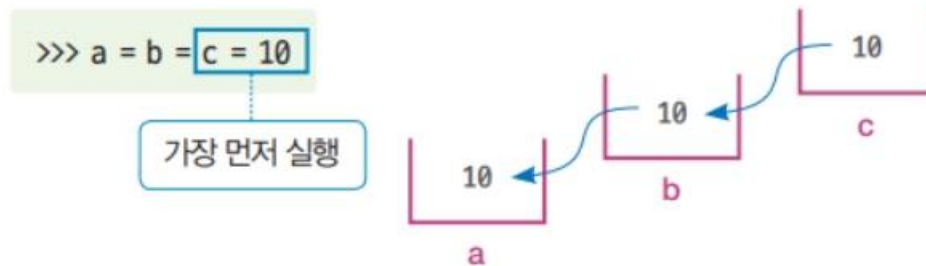


그림 4-13 대입 연산자의 실행 순서

➤ `a = b = c = 10`

- `c = 10`
- `b = c`
- `a = b`



➤ `a, b, c = 10, 10, 10`

- `a = 10`
- `b = 10`
- `c = 10`



02. 연산자

I. 연산자의 종류

■ 대입 연산자

- 대입 연산자와 산술 연산자를 함께 표기한 연산자들이 복합 대입 연산자
- 앞에 연산을 한 후에 뒤의 연산(= 대입)을 하는 복합 대입 연산자

➤ `sum1 += 1` → `sum1 + 1` 후에, 결과를 `sum1`에 저장(=)

표 4-7 복합 대입 연산자의 종류

연산자	예시	의미	실행 결과
<code>+=</code>	<code>x += 5</code>	<code>x = x + 5</code>	x의 값이 5 증가
<code>-=</code>	<code>x -= 5</code>	<code>x = x - 5</code>	x의 값이 5 감소
<code>*=</code>	<code>x *= 5</code>	<code>x = x * 5</code>	x는 5배수 값으로 변경
<code>/=</code>	<code>x /= 5</code>	<code>x = x / 5</code>	x를 5로 나눈 값으로 변경
<code>//=</code>	<code>x //= 5</code>	<code>x = x // 5</code>	x를 5로 나눈 몫으로 변경
<code>%=</code>	<code>x %= 5</code>	<code>x = x % 5</code>	x를 5로 나눈 나머지로 변경

02. 연산자

I. 연산자의 종류

■ 대입 연산자

실습 4-9

복합 대입 연산자 사용하기

- 변수 x와 y에 다음과 같이 복합 대입 연산자를 활용한 수식을 만들어 실행

- ① 변수 x와 y에 각각 정수 10과 5를 저장하고 수식을 만들어 실행

```
>>> x = 10
>>> y = 5
>>> x += y
>>> x *= y
>>> x %= y
```

```
x = x + y
x = x * y
x = x % y
```

- ② 변수 x와 y를 출력하여 변경된 값을 확인

```
>>> print("x = %d\ny = %d" %(x, y))
x = 0
y = 5
```

02. 연산자

I. 연산자의 종류

하나 더 알기

연산자의 우선순위

연산자의 우선순위

앞서 살펴본 여러 가지 연산자들이 수식에 같이 사용될 때는 연산자 우선순위에 따라 실행 순서가 결정된다. 산술 > 비교 > 논리 > 대입 연산자의 순으로 우선순위가 높다. 하지만 어떤 상황에서도 가장 먼저 실행해야 할 부분은 괄호() 안에 있는 수식이다.

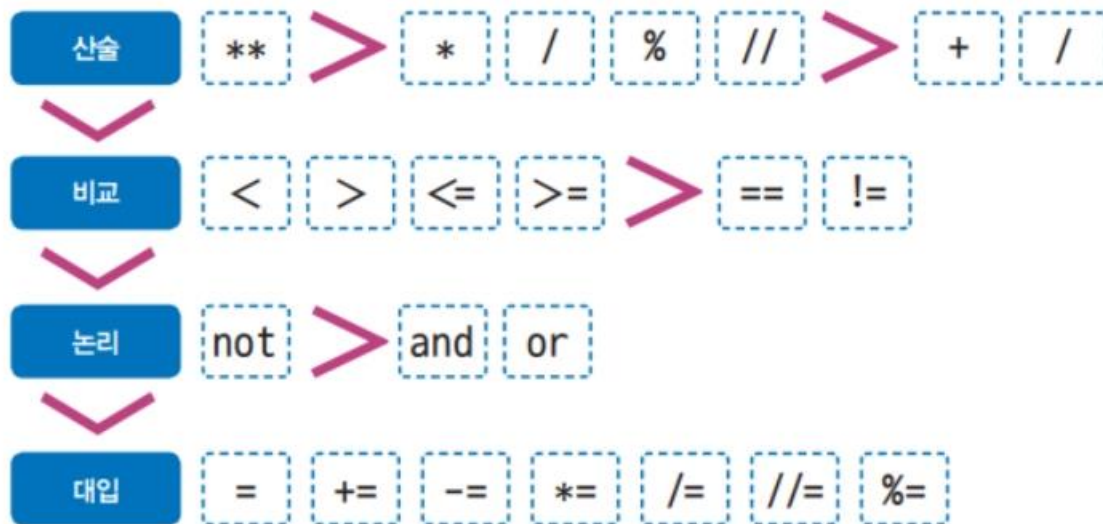


그림 4-14 연산자의 우선순위

02. 변수의 개념과 활용

[문제]

3. 아래 코드를 순차적으로 실행 할 때의 각 실행 결과를 우측에 적으시오.

```
➤ num = 20
```

- | | | |
|--------------|-------------|-------|
| ① num *= 2; | print(num); | _____ |
| ② num //= 2; | print(num); | _____ |
| ③ num += 2; | print(num); | _____ |
| ④ num %= 10; | print(num); | _____ |
| ⑤ num /= 2; | print(num); | _____ |

4. 아래 코드를 순차적으로 실행 할 때의 각 실행 결과를 우측에 적으시오.

```
➤ kum = 782  
➤ won = 0
```

- | | | |
|---------------------|-------------|-------|
| ① won = kum // 500; | print(won); | _____ |
| ② kum %= 500; | print(kum); | _____ |

02. 연산자

I. 연산자의 종류

■ 비트 연산자

- 2진수의 각 비트 간의 논리 연산을 수행

➤ bin()는 전달된 값을 2진수로 병환해주는 함수

연산자	의미	설명
&	비트 논리곱(and)	둘 다 1이면 1
	비트 논리합(or)	둘 중 하나만 1이면 1
^	비트 논리적 배타합(xor)	둘이 같으면 0, 다르면 1
~	비트 부정	1은 0으로, 0은 1로 변경
<<	비트 이동(왼쪽)	비트를 왼쪽으로 시프트(Shift)
>>	비트 이동(오른쪽)	비트를 오른쪽으로 시프트(Shift)

02. 연산자

I. 연산자의 종류

- 비트 연산자 : 논리곱 &

A	B	A&B
0	0	0
0	1	0
1	0	0
1	1	1

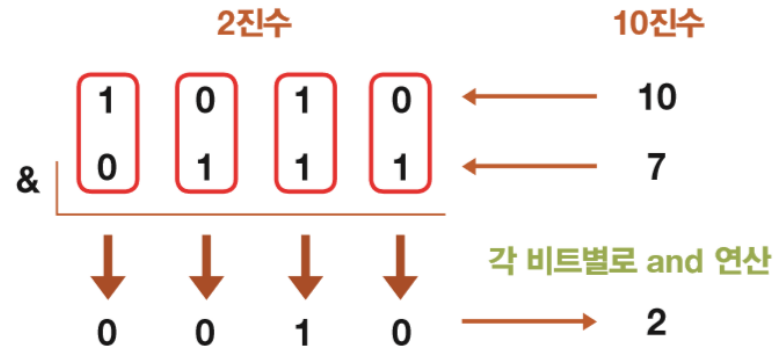


그림 4-2 비트 논리곱의 예

02. 연산자

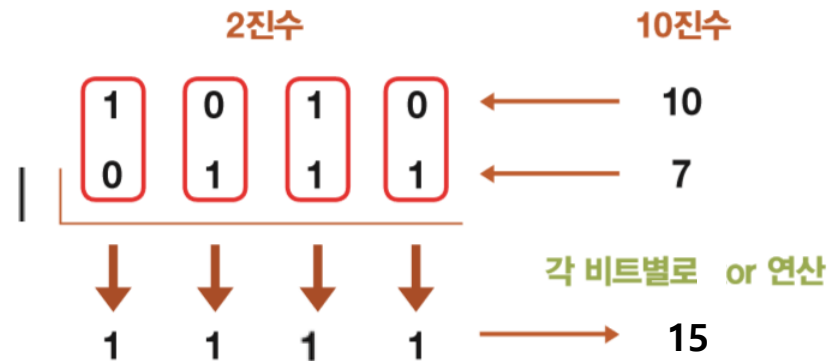
I. 연산자의 종류

- 비트 연산자 : 논리합 |

A	B	A B
0	0	0
0	1	1
1	0	1
1	1	1

그림 4-4 비트

논리합의 예



02. 연산자

I. 연산자의 종류

- 비트 연산자 : 배타적 논리합 \wedge

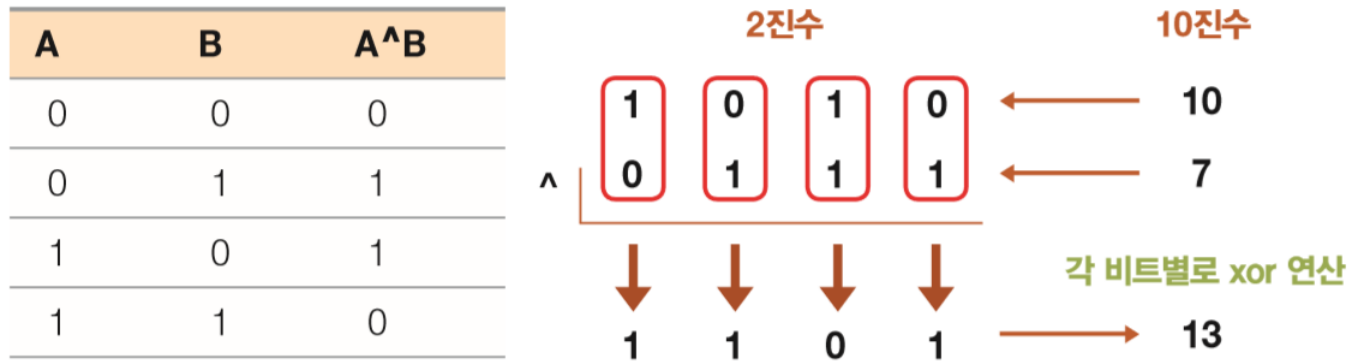


그림 4-4 비트 배타적 논리합의 예

02. 연산자

I. 연산자의 종류

- 비트 연산자 : 비트 마스크

- 원하는 자리만을 추출하기 위해 해당 위치의 비트 값을 논리곱(&) 1 수행

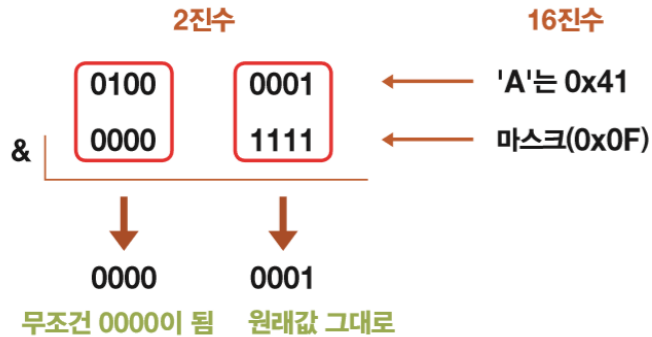


그림 4-5 마스크 0x0F를 사용한 비트 논리곱 결과

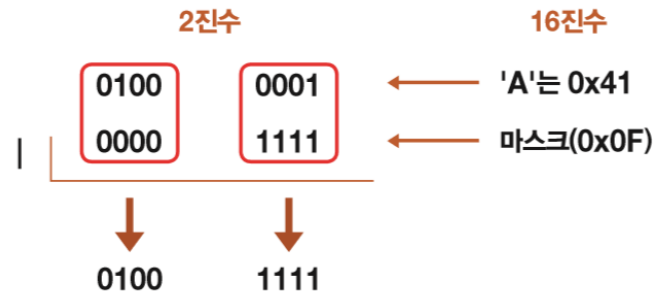


그림 4-6 마스크 0x0F를 사용한 비트 논리합 결과

02. 연산자

I. 연산자의 종류

- 비트 연산자 : 시프트 >> <<



그림 4-7 26의 왼쪽으로 2칸 시프트 연산

02. 연산자

I. 연산자의 종류

■ 비트 연산자

- 2진수의 각 비트 간의 논리 연산을 수행

➤ bin()는 전달된 값을 2진수로 병환해주는 함수

```
a = 0b10100101
b = 0b11000011
-----
```

a & b

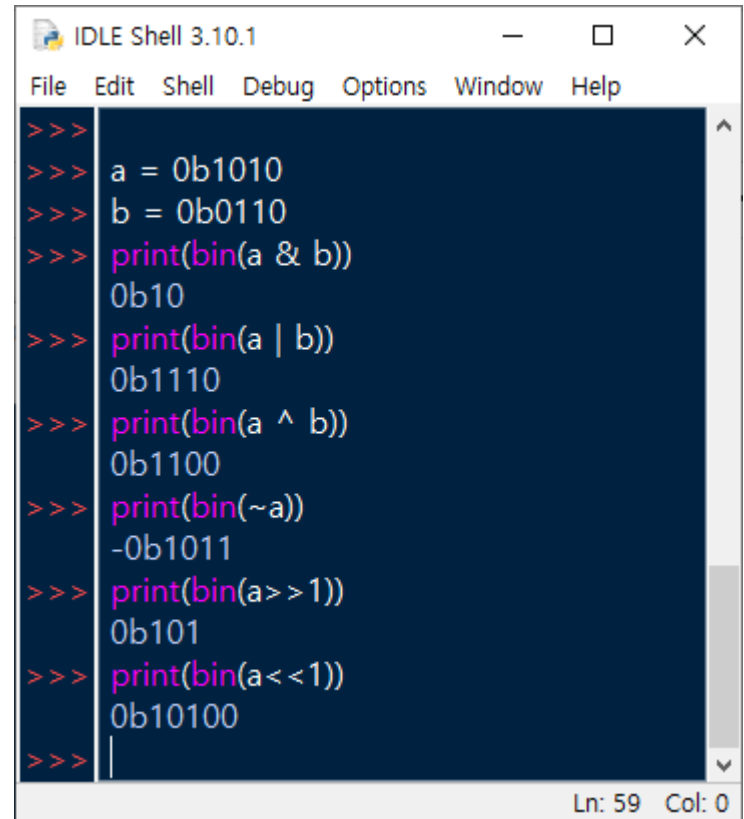
a | b

a ^ b

~a

a >> 1

a << 1



```
IDLE Shell 3.10.1
File Edit Shell Debug Options Window Help
>>>
>>> a = 0b1010
>>> b = 0b0110
>>> print(bin(a & b))
0b10
>>> print(bin(a | b))
0b1110
>>> print(bin(a ^ b))
0b1100
>>> print(bin(~a))
-0b1011
>>> print(bin(a >> 1))
0b101
>>> print(bin(a << 1))
0b10100
>>>
```

Ln: 59 Col: 0

02. 연산자

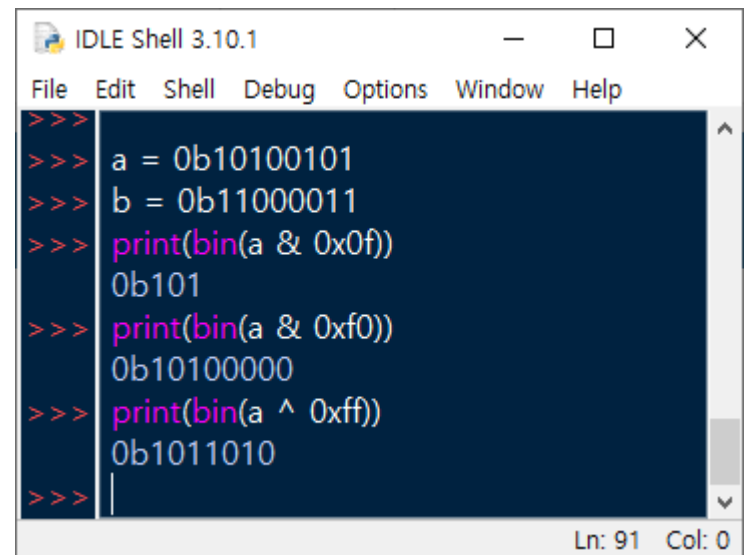
I. 연산자의 종류

■ 비트 연산자

- 2진수의 각 비트 간의 논리 연산을 수행

➤ bin()는 전달된 값을 2진수로 병환해주는 함수

```
a = 0b10100101
b = 0b11000011
-----
a & b
a & 0x0f
a & 0xf0
a ^ 0xff #비트 뒤집기
```



```
IDLE Shell 3.10.1
File Edit Shell Debug Options Window Help
>>> 
>>> a = 0b10100101
>>> b = 0b11000011
>>> print(bin(a & 0x0f))
0b101
>>> print(bin(a & 0xf0))
0b10100000
>>> print(bin(a ^ 0xff))
0b1011010
>>> 
```

Ln: 91 Col: 0

02. 연산자

I. 연산자의 종류

■ 비트 연산자 : 2의 보수 연산

- CPU는 가산기만 가지고 뺄셈도 덧셈으로 처리
- (양수 - 양수)와 같은 뺄셈은 (양수 + 음수) 형태로 바꾸어 덧셈 처리
- 음수는 양수를 2의 보수(2's complement == complement + 1)를 취하는 것으로 표현

➤ $a = 3;$ $b = \sim a + 1;$ $a + b;$

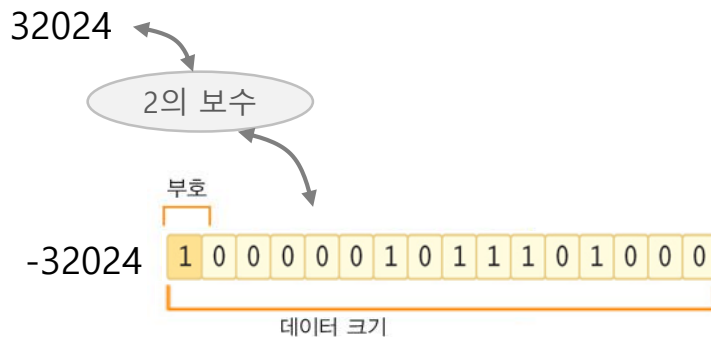


그림 3-7 • 양의 정수 표현방식

```
IDLE Shell 3.10.1
File Edit Shell Debug Options Window Help
>>> a = 0b1010
>>> b = ~a + 1
>>> print(bin(~a))
-0b1011
>>> print(bin(~a+1))
-0b1010
>>> print(bin(b))
-0b1010
>>> print(bin(a + (~a+1) ))
0b0
>>>
```

Ln: 70 Col: 0

02. 변수의 개념과 활용

[문제]

5. 아래 코드를 순차적으로 실행 할 때의 각 실행 결과를 우측에 적으시오.

```
➤ a = 0b11001010  
➤ b = 0b10100011
```

- | | | | |
|---|---|-----------------------------|-------|
| ① | <code>c = a & b;</code> | <code>print(bin(c));</code> | _____ |
| ② | <code>c = a b;</code> | <code>print(bin(c));</code> | _____ |
| ③ | <code>c = a ^ b;</code> | <code>print(bin(c));</code> | _____ |
| ④ | <code>c = a & 0x0f;</code> | <code>print(bin(c));</code> | _____ |
| ⑤ | <code>c = a ^ 0x0f;</code> | <code>print(bin(c));</code> | _____ |
| ⑥ | <code>c = (a << 4) >> 4;</code> | <code>print(bin(c));</code> | _____ |

02. 연산자

II. 연산자의 활용

실습 4-11

거스름돈 계산하기

code04-11.py

- 문구점에서 연필을 사려고 한다. 연필 한 자루가 400원이라고 한다면, 몇 자루의 연필을 살 수 있고 거스름돈은 얼마인지 계산하는 프로그램

- ① 프로그램이 실행되는 순서를 먼저 생각



그림 4-16 연필 개수와 거스름돈 계산 순서

- ② 가진 돈이 3천 원이라고 가정하고 IDLE 셸에서 간단하게 먼저 계산, 나눗셈 연산으로 계산하면 살 수 있는 연필의 개수는 7.5 자루

```
>>> 3000 / 400  
7.5
```

02. 연산자

II. 연산자의 활용

실습 4-11

거스름돈 계산하기

code04-11.py

- ③ 연필은 자루 단위로만 살 수 있기 때문에 다른 계산 방법을 사용해야 함, 몫과 나머지를 계산하는 산술 연산자(//, %)를 사용해서 다시 계산

```
>>> 3000 // 400
7
>>> 3000 % 400
200
```

- ④ 몫과 나머지 계산 방법과 실행 순서에 맞게 전체 프로그램을 작성

```
01 money = int(input("원 단위로 액수를 입력하세요. : ")) # 가진 돈의 액수
02 pencil = 400 # 연필 가격
03 print("연필 개수: %d 자루" % (money // pencil)) # 연필 개수 계산 후 출력
04 print("거스름돈 : %d 원" % (money % pencil)) # 거스름돈 계산 후 출력
```

- ③ 실행 결과 확인

02. 연산자

II. 연산자의 활용

실습 4-12

박테리아 개체 수 계산하기

code04-12.py

- 만약 시간당 두 배로 분열하는 박테리아 100개가 있다면, 한 시간 후 200개, 두 시간 후에는 400개로 증식
- 하루가 지나고 나면 박테리아 개체 수는 얼마로 늘어나 있을지 계산

①

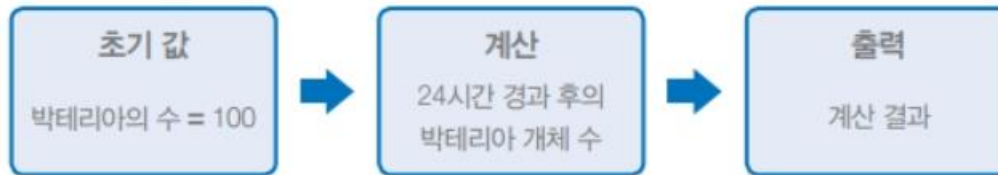


그림 4-20 박테리아 개체 수 계산 순서

02. 연산자

II. 연산자의 활용

실습 4-12

박테리아 개체 수 계산하기

code04-12.py

- ② 시간(x)이 경과함에 따라 증식하는 개체 수에 대해 지수 함수식 생성

```
>>> 100                                # 시작 개체 수
100
>>> 100 * 2                            # 1시간 후 = 100*2
200
>>> (100 * 2) * 2                       # 2시간 후 = 100*22
400
>>> (100 * 2 * 2) * 2                   # 3시간 후 = 100*23
800                                     # 100 * (2 ** x)이므로 x 시간 후 = 100*2x
```

- ③ 실행 순서에 맞게 전체 프로그램을 만들고 저장

```
01 number = 100                        # 시작 개체 수
02 x = 24                              # 경과 시간
03 result = number * (2 ** x)          # x 시간 후 박테리아의 개체 수 계산
04 print("%d개의 박테리아의 %d시간 후 개체 수 = %d" % (number, x, result))
```

- ④ 실행 결과 확인

02. 연산자

II. 연산자의 활용

하나 더 알기 `format()` 함수

- 숫자 데이터의 형식을 지정할 때 이용

표 4-9 `format()` 함수 사용법

예시	의미	실행 결과
<code>format(123456789, ',')</code>	세자리마다 쉼표(.) 넣기	<code>'1,234,567'</code>
<code>format(123456789, '15')</code>	자릿수를 15로 하기	<code>'123456789'</code>
<code>format(123456789, '<15')</code>	자릿수를 15로 하고 왼쪽 정렬하기	<code>'123456789 '</code>
<code>format(123456789, '015')</code>	자릿수를 15로 하고 빈 자리를 0으로 채우기	<code>'000000123456789'</code>
<code>format(123456789, 'x')</code>	16진수로 표시하기	<code>'75bcd15'</code>
<code>format(123456789, 'e')</code>	지수 표기법으로 표시하기	<code>'1.234568e+08'</code>
<code>format(12345.6789, '.2f')</code>	소수점 이하 2자리로 실수 표시하기	<code>'12345.68'</code>

02. 연산자

II. 연산자의 활용

실습 4-13

친구들과 피자 나눠 먹기

code04-13.py

- 세 명이 모여 피자를 주문, 피자를 2판 주문한다면 한 사람당 몇 조각씩?

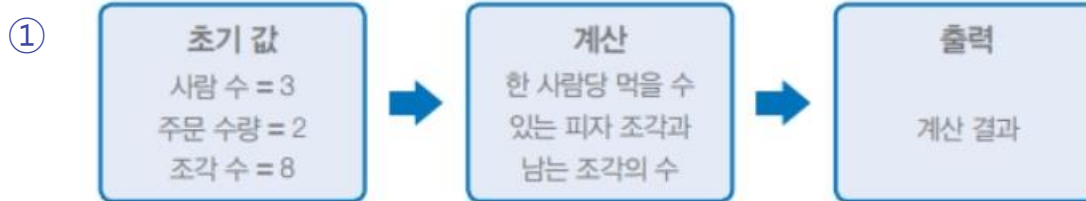


그림 4-22 피자 나눠먹기 계산 순서

- ② 실행 순서에 맞게 전체 프로그램을 만들고 저장

```
01 person = 3                # 사람 수
02 order = 2                  # 피자 주문 수량
03 piece = 8                  # 피자당 조각 수
04 result = (order * piece) // person  # 한 사람당 먹을 수 있는 조각 수
05 remain = (order * piece) % person  # 남는 조각 수
06
07 print("%d 조각씩 먹을 수 있고, %d 조각이 남습니다." % (result, remain))
```

02. 연산자

II. 연산자의 활용

실습 4-14

장학금 대상인지 확인하기

code04-14.py

- 내년도 장학금을 타려면 올해 학점 평균은 3.0 이상, 봉사시간은 10시간 이상 필요

①

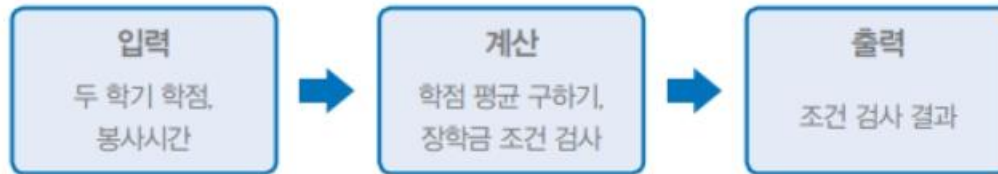


그림 4-23 장학금 조건 구하기 계산 순서

- ② 장학금을 받는 조건은 학점 평균과 봉사시간이 모두 정해진 기준 이상이어야 하므로, 비교 연산식 2개를 논리 연산자 and로 결합



그림 4-24 장학금 대상 기준

02. 연산자

II. 연산자의 활용

실습 4-14

장학금 대상인지 확인하기

code04-14.py

- ③ 실행 순서와 문제의 조건식에 맞게 프로그램을 만들고 저장

```
01 grade1 = float(input("1학기 학점 입력 : "))
02 grade2 = float(input("2학기 학점 입력 : "))
03 time = int(input("봉사시간 입력 : "))
04
05 average = (grade1 + grade2) / 2           # 학점 평균 구하기
06 result = (average >= 3.0) and (time >= 10) # 장학금 대상인지 검사하기
07 print("장학금 대상 여부 =", result)       # 결과 출력하기
```

- ④ 실행 결과 확인

```
1학기 학점 입력 : 3.34
2학기 학점 입력 : 2.85
봉사시간 입력 : 12
장학금 대상 여부 = True
```

02. 연산자

II. 연산자의 활용

하나 더 알기 주석의 이용

- 프로그램을 작성할 때 참고할 사항이나 필요한 설명을 자유롭게 기술
- 주석은 '#' 기호부터 그 줄의 마지막 내용까지에 해당하고, 프로그램 실행에서 제외되는 부분
- 주석을 여러 행에 나누어 작성하려면 큰따옴표나 작은따옴표 3개(''', ''')를 이용
- 편집기에서는 단축키를 이용해서 주석 기호(##)를 자동으로 붙일 수 있는데, 주석 처리할 문장을 선택하고 Alt + 3 / 제거하는 단축키는 Alt + 4

```
01  '''=====
02  다각형의 내각 계산 프로그램
03  ====='''
04  n = int(input("3에서 6 사이의 정수를 입력하세요 : "))      # 다각형 모양 선택
05  sumAngle = 180 / (n - 2)                                     # 내각의 합 계산
```

02. 변수의 개념과 활용

[Python실습] 거스름 돈 반환 최소화

◆ 키보드로 다음을 입력받아

- >청구한 금액 입력(원)?
- >받은 금액 입력(원)?

◆ 다음 내용을 출력하는 프로그램을 완성하시오.

단, 거스름 돈은 최소 개수가 되도록 하시오.

- >거스름 총 금액 :
- >5만원권 %d매, 1만원권 %d매, 5천원권 %d매, 천원권 %d매
500원동전 %d개, 100원동전 %d개, 50원동전 %d개, 10원동전 %d개, 1원동전 %d개

```
청구한 금액 입력(원)? 12837
받은 금액 입력(원)? 100000
거스름 총 금액 : 87163
5만원권 1매
1만원권 3매
5천원권 1매
1천원권 2매
5백원동전 0개
1백원동전 1개
50원동전 1개
10원동전 1개
1원동전 3개
```

```
>>> |
```

02. 변수의 개념과 활용

[Python실습] 거스름 돈 반환 최소화

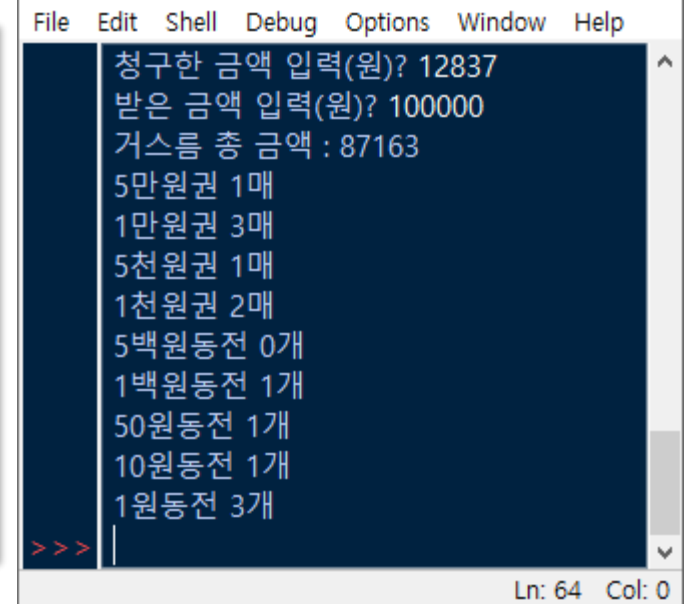
◆ 키보드로 금액을 입력받아 반환 할 화폐별로 반환 수 출력

- >청구한 금액 입력(원)?
- >받은 금액 입력(원)?
- >거스름 총 금액 :
- >5만원권 %d매, 1만원권 %d매, 5천원권 %d매, 천원권 %d매
500원동전 %d개, 100원동전 %d개, 50원동전 %d개, 10원동전 %d개, 1원동전 %d개

```
totwon = 0
inwon = 0
outwon = 0

totwon = int(input("청구한 금액 입력(원)? "))
inwon = int(input("받은 금액 입력(원)? "))

outwon = inwon - totwon ##거스름 금액
print("거스름 총 금액 : %d" %outwon)
```



```
File Edit Shell Debug Options Window Help
청구한 금액 입력(원)? 12837
받은 금액 입력(원)? 100000
거스름 총 금액 : 87163
5만원권 1매
1만원권 3매
5천원권 1매
1천원권 2매
5백원동전 0개
1백원동전 1개
50원동전 1개
10원동전 1개
1원동전 3개
>>>
```

Ln: 64 Col: 0

02. 변수의 개념과 활용

[과제] 거스름 돈 최소 개수 계산 (화폐 개수가 0이면 출력하지 않음)

◆ 키보드로 다음을 입력받아

- >청구한 금액 입력(원)?
- >받은 금액 입력(원)?

➤ if 조건연산식:
처리문장

◆ 다음 내용을 출력하는 프로그램을 완성하시오.

단, 거스름 돈은 최소 개수가 되도록 하시오. (화폐 개수가 0이면 출력하지 않음)

- >거스름 총 금액 :
- >5만원권 %d매, 1만원권 %d매, 5천원권 %d매, 천원권 %d매
500원동전 %d개, 100원동전 %d개, 50원동전 %d개, 10원동전 %d개, 1원동전 %d개

```
청구한 금액 입력(원)? 12837
받은 금액 입력(원)? 100000
거스름 총 금액 : 87163
5만원권 1매
1만원권 3매
5천원권 1매
1천원권 2매
1백원동전 1개
50원동전 1개
10원동전 1개
1원동전 3개
>>>
```

Ln: 16 Col: 0

Thank You !

[Python]