

파이썬 포트폴리오

20191780 강혜원

목차

1 강의계획서

2 자기주도 학습자료

2.1 수업 내용 및 책 내용 요약

2.2 과제

강의계획서

2020학년도 1학기	전공	컴퓨터 공학과	학부	컴퓨터 공학부
과목명	파이썬 프로그래밍 (20190009-PC)			
강의실, 강의시간	화:1(3-217),2(3-217),3(3-217)	학점	3	
교과분류	이론 / 실습	시수	3	

담당 교수	강환수
	+ 연구실: 2호관 - 706
	+ 전화: 02) 2610 - 1941
	+ E-MAIL: hs kang@dngyang.ac.kr
	+ 면담가능기간: 화, 13:00 ~ 16:00

학과 교육목표				
과목 개요	2010년 이후 파이썬의 폭발적인 인기는 제4차 산업혁명 시대의 도래와도 밀접한 연관성이 있다. 컴퓨팅 사고력은 누구나 가져야 할 역량이며, 인공지능, 빅데이터, 사물인터넷 등의 첨단 정보기술이 제4차 산업혁명 시대의 기술을 이끌고 있다. 제4차 산업혁명 시대를 주도하는 핵심 기술은 데이터과학과 머신러닝, 딥러닝이며, 이러한 분야에 적합한 언어인 파이썬은 매우 중요한 언어가 되었다. 본 교과목은 파이썬 프로그래밍의 기초적이고 체계적인 학습을 수행한다. 본 교과목을 통하여 데이터 처리 방법에 대한 효율적인 파이썬 프로그래밍 방법을 학습한다.			
학습 목표 및 성취 수준	1. 컴퓨팅 사고력의 중요성을 인지하고 4차 산업혁명에서 파이썬 언어의 필요성을 이해할 수 있다. 2. 기본적인 파이썬 문법을 이해하고 데이터 처리를 위한 자료 구조를 이해하여 적용할 수 있다. 3. 문제 해결 방법을 위한 알고리즘을 이해하고 데이터 처리에 적용 할 수 있다. 4. 파이썬 프로그램을 이용하여 실무적인 코딩 작업을 할 수 있다.			
	도서명	저자	출판사	비고
주 교재	파이썬으로 배	강환수, 신용	홍릉과학출판	

	우는 누구나 코딩	현	사	
수업 시 사용 도구	파이썬 기본 도구, 파이참, 아나콘다와 주피터 노트북			
평가 방법	중간고사 30%, 기말고사 40%, 과제물 및 퀴즈 10% 출석 20%(학교 규정, 학업성적 처리 지침에 따름)			
수강 안내	1. 파이썬의 개발환경을 설치하고 활용할 수 있다. 2. 파이썬의 기본 자료형을 이해하고 조건과 반복 구문을 활용할 수 있다. 3. 파이썬의 주요 자료인 리스트, 튜플, 딕셔너리, 집합을 활용할 수 있다. 4. 파이썬의 표준 라이브러리와 외부 라이브러리를 이해하고 활용할 수 있다. 5. 파이썬으로 객체지향 프로그래밍을 수행할 수 있다.			

1주차	개강일 (03/16)
학습 주제	교과목 소개 및 강의 계획
목표 및 내용	1장 파이썬 언어의 개요와 첫 프로그래밍
미리 읽어오기	파이썬 언어란 무엇인지 이해하고 이 언어가 인기 있는 이유를 설명할 수 있다.
과제, 시험, 기타	파이썬 개발 도구를 설치해 프로그램을 구현할 수 있다.

2주차	2주
학습 주제	2장 파이썬 프로그래밍을 위한 기초 다지기
목표 및 내용	파이썬의 재료인 문자열과 수에 대해 이해하고 코드로 구현할 수 있다.
미리 읽어오기	변수를 이해하고 다양한 대입 연산자를 활용할 수 있다.
과제, 시험, 기타	표준 입력으로 문자열을 입력받은 후 원하는 자료로 변환해 활용할 수 있다.

3주차	3주
학습 주제	3장 일상에서 활용되는 문자열과 논리 연산
목표 및 내용	문자열에서 문자나 부분 문자열을 반환하는 여러 방법을 구현할 수 있다.
미리 읽어오기	문자열 객체에 소속된 다양한 메소드를 이해하고 활용할 수 있다.
과제, 시험, 기타	논리 값을 이해하고 다양한 연산을 사용해 실생활에서의 표현에 활용할 수 있다.

4주차	4주
학습 주제	4장 일상생활과 비유되는 조건과 반복
목표 및 내용	조건에 따라 하나를 결정하는 if문을 구현할 수 있다.
미리 읽어오기	반복을 수행하는 while문과 for문을 구현할 수 있다.
과제, 시험, 기타	임의의 수인 난수를 이해하고 반복을 제어하는 break문과 continue문을 활용할 수 있다.

5주차	5주
학습 주제	5장 항목의 나열인 리스트와 튜플
목표 및 내용	다양한 종류의 항목을 쉽게 나열하는 리스트를 구현할 수 있다.
미리 읽어오기	리스트에서 부분 참조 방법, 이를 이용한 수정, 리스트 연결, 삽입과 삭제 그리고 리스트 컴프리헨션 등을 구현할 수 있다.
과제, 시험, 기타	수정할 수 없는 다양한 종류의 항목 나열을 쉽게 처리하는 튜플을 구현할 수 있다.

6주차	6주
학습 주제	6장 키와 값의 나열인 딕셔너리와 중복을 불허하는 집합
목표 및 내용	키와 값의 쌍인 항목을 관리하는 딕셔너리를 생성하고 수정하는 방법을 이해하고, 다양한 방법으로 딕셔너리를 구현할 수 있다.
미리 읽어오기	집합의 특징을 이해하고, 합집합 등과 같은 다양한 집합의 연산을 구현할 수 있다.
과제, 시험, 기타	내장 함수 zip()과 enumerate(), 시퀀스 간의 변환을 이해하고, 구현할 수 있다.

7주차	7주
학습 주제	7장 특정 기능을 수행하는 사용자 정의 함수와 내장 함수
목표 및 내용	함수의 내용과 필요성을 이해하고 함수를 직접 정의해 호출할 수 있다.
미리 읽어오기	인자의 기본 이해와 기본값 지정, 가변 인수와 키워드 인수를 활용할 수 있다.
과제, 시험, 기타	간편한 람다 함수와 표준 설치된 내장 함수를 사용할 수 있다.

8주차	중간고사
학습 주제	직무수행능력평가 1차(중간고사)

목표 및 내용	직무수행능력평가, 서술형 평가
미리 읽어오기	교재 1장에서 7장까지
과제, 시험, 기타	

9주차	9주
학습 주제	8장 조건과 반복, 리스트와 튜플 기반의 미니 프로젝트 I
목표 및 내용	8개의 미니 프로젝트를 스스로 생각하고 프로그래밍해 코딩 능력뿐 아니라 문제 해결 능력을 키울 수 있다.
미리 읽어오기	교재 8장
과제, 시험, 기타	

10주차	10주
학습 주제	9장 라이브러리 활용을 위한 모듈과 패키지
목표 및 내용	표준 모듈을 이해하고 사용자 정의 모듈도 직접 구현해 사용할 수 있다.
미리 읽어오기	표준 모듈인 turtle을 사용해 기본적인 도형을 그릴 수 있다.
과제, 시험, 기타	썬드파티 모듈 numpy와 matplotlib 등을 설치해 활용할 수 있다.

11주차	11주
학습 주제	10장 그래픽 사용자 인터페이스 Tkinter와 Pygame
목표 및 내용	GUI를 이해하고 GUI 표준 모듈인 Tkinter를 사용해 필요한 위젯을 구성하고 윈도우를 생성할 수 있다.
미리 읽어오기	이벤트 처리를 이해하고 Tkinter에서 이벤트 처리를 구현할 수 있다.
과제, 시험, 기타	썬드파티 GUI 모듈인 pygame을 설치해 기본적인 윈도우를 구현할 수 있다.

12주차	12주
학습 주제	11장 실행 오류 및 파일을 다루는 예외 처리와 파일 입출력
목표 및 내용	예외 처리의 필요성을 이해하고 try except 구문을 사용해 예외를 처리할 수 있다.
미리 읽어오기	프로그램에서 파일을 생성하는 필요성을 이해하고 필요한 파일을 만들 수 있다.
과제, 시험, 기타	이미 생성된 파일에서 내용을 읽어 처리할 수 있다

13주차	13주
학습 주제	12장 일상생활의 사물 코딩인 객체지향 프로그래밍
목표 및 내용	객체와 클래스를 이해하고 필요한 클래스를 정의하고 객체를 만들어 활용할 수 있다.
미리 읽어오기	클래스 속성과 인스턴스 속성, 정적 메소드와 클래스 메소드를 이해하고 정의할 수 있다.
과제, 시험, 기타	상속을 이해하고 부모 클래스와 자식 클래스를 정의할 수 있다.

14주차	14주
학습 주제	13장 GUI 모듈과 객체지향 기반의 미니 프로젝트 II
목표 및 내용	학습한 파이썬 문법 구조와 프로그래밍 기법을 활용해 8개의 미니 프로젝트를 스스로 생각하고 프로그래밍해 코딩 능력뿐 아니라 문제 해결 능력을 키울 수 있다.
미리 읽어오기	교재 1장
과제, 시험, 기타	

15주차	기말고사
학습 주제	직무수행능력평가 2차(기말고사)
목표 및 내용	직무수행능력평가, 서술형평가
미리 읽어오기	8장에서 13장까지
과제, 시험, 기타	

수업 지원 안내	장애학생을 위한 별도의 수강 지원을 받을 수 있습니다. 언어가 문제가 되는 학생은 글로 된 과제 안내, 확대문자 시험지 제공 등의 지원을 드립니다.
----------	---

CHAPTER 01. 파이썬 언어의 개요와 첫 프로그래밍.

1. 파이썬 언어와 컴퓨팅 사고력

1.1 파이썬 언어란?

파이썬 (PYTHON)

- 오픈 소스(open source) 프로그래밍 언어.

- 1991년, 네덜란드의 귀도 반 로섬 (Guido van Rossum)이 개발, 현재는 파이썬 소프트웨어 재단(PSF)이 관리.

- 미국과 우리나라의 대학 등 전 세계적으로 가장 많이 가르치는 프로그래밍 언어 중 하나. 가장 빠르게 성장하는 프로그래밍 언어로 선택되기도 함.

- 큰 장점: 배우기 쉽고 간결, 개발 속도가 빠르고 강력, 라이브러리가 풍부하고 다양한 개발환경 제공

1.2 컴퓨팅 사고력과 파이썬

- 현재, 사람들은 4차 산업혁명 시대에 살고 있음. 4차 산업혁명 시대란 모든 사물이 연결된 초연결 사회에서 생산되는 빅데이터를 기존 산업과 융합하여 인공지능, 클라우드 등의 첨단 기술로 처리하는 정보, 지능화 혁명 시대를 말함.

- 문제 해결 능력, 창의·융합 사고 능력, 의사소통 능력과 협업 능력, 자기주도 학습 능력 등을 가장 갖추어야 할 덕목으로 삼고 있음. 미래의 인재는 컴퓨터 과학 원리와 개념을 이용해 자신의 영역과 융합할 수 있는 역량을 갖추어야 함.

- 그러나 컴퓨팅 사고력은 모든 사람에게 필요한 기본 역량.

- 프로그래밍의 절차: 이해, 설계, 구현, 공유.

2. 파이썬 설치와 파이썬 셸 실행

2.1 파이썬 개발 도구 설치와 파이썬 셸의 실행

2.2 파이썬 셸에서 첫 대화형 프로그래밍

- 모든 명령어는 첫 칸부터 입력, 만일 첫 칸, 혹은 그 이상을 공간으로 두고 입력하면 오류 발생.

2.3 편집기에서 첫 파일 프로그래밍

- 셸 만이 아니라 편집기를 사용하여 모든 프로그램을 짰 후 실행시킬 수 있음.

3. 제 4차 산업혁명 시대, 모두에게 필요한 파이썬

3.1 쉽고 강력한 언어

- 파이썬은 간결하고 배우기 쉬우며, 무료이고 생산성이 높고, 강력한 라이브러리를 제공하며, 프로그램과 호환되는 풀 언어임.

3.2 빅데이터 처리와 머신 러닝 등 다양한 분야에 적합한 언어

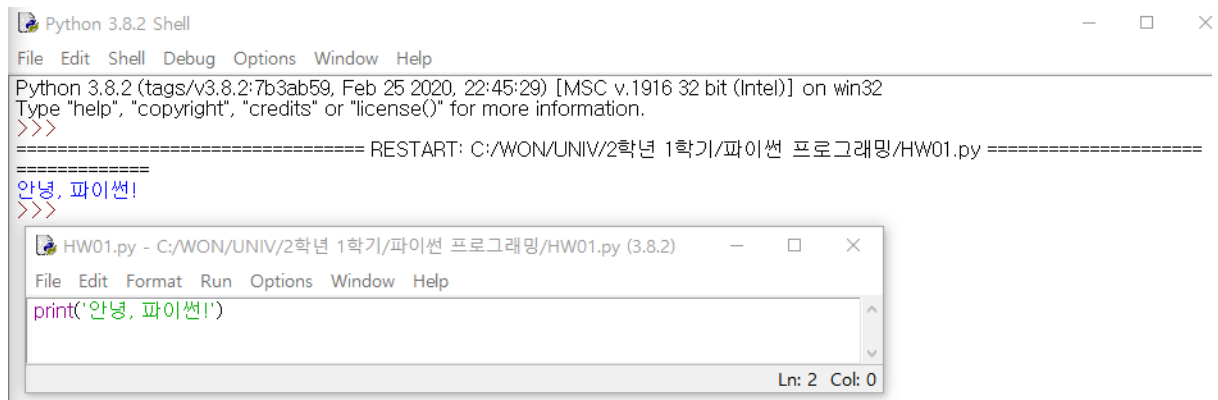
- 그러나 속도가 조금 느리다는 단점이 있음.

3.3 다양한 종류의 파이썬과 개발 환경

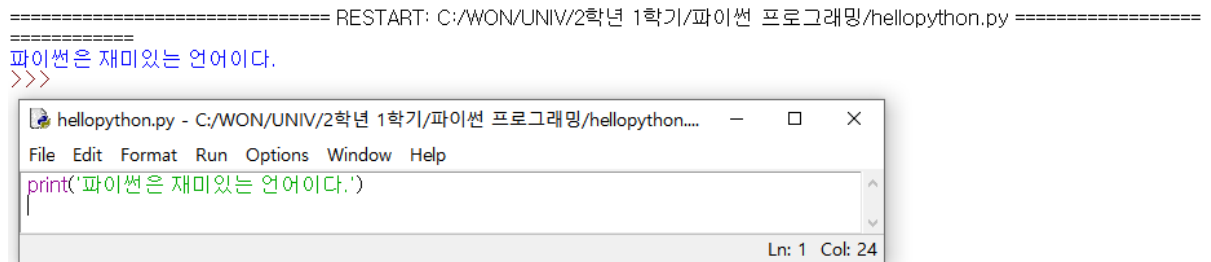
3.4 인터프리트 방식의 언어, 파이썬

과제물 [1주차] 도전! 프로그래밍 (29pg)

① 파이썬 IDLE에서 다음을 출력하는 코드를 작성하시오.

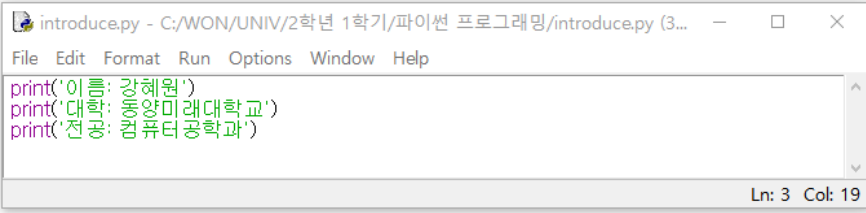


② 파이썬 셸에서 다음을 출력하는 프로그램을 지정된 파일에 저장해 실행하시오. (파일: hellopython.py)



③ 파이썬 셸에서 자신을 소개하는 프로그램을 지정된 파일에 저장해 실행하시오. (파일: introduce.py)

```
===== RESTART: C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/introduce.py =====
=====
이름: 강혜원
대학: 동양미래대학교
전공: 컴퓨터공학과
>>>
```



CHAPTER 02. 파이썬 프로그래밍을 위한 기초 다지기

1. 다양한 자료: 문자열과 수

1.1 자료의 종류와 문자열 표현

– 문자열: 문자 하나 또는 문자가 모인 단어나 문장 또는 단락 등을 말함. 일련(sequence)의 문자(character) 모임이라 할 수 있음. 파이썬에서 작은 따옴표나 큰 따옴표로 앞뒤를 둘러싸 ‘문자열’, 또는 “문자열” 처럼 표현함. 따옴표로 둘러싼 것은 모두 문자열.

– 출력 함수 print(): print(출력될 자료)는 문자열이나 숫자 등의 자료를 콘솔에 출력하는 일을 수행함. 출력 이후에는 다음 줄로 이동해 출력을 준비한다.

– 따옴표: 문자열에서 따옴표는 앞뒤를 동일하게 사용. 예를 들어 ‘가능’, “가능”, ‘불가능’, “불가능” 임. 불가능일 경우에는 구문 오류, Syntax 오류 발생.

1.2 문자열의 연산자 +, *와 주석

– 문자열을 연결할 때는 공백, 또는 +를 이용함. 예를 들어 print(“문자열” ‘연결’)의 결과는 ‘문자열 연결’ 이고, print(“문자열” + ‘연결’)의 결과는 ‘문자열연결’ 임.

- 문자열을 반복할 때는 *를 사용함. 반복 횟수인 정수가 필요하므로 문자열만 두 개라면 작동되지 않지만 print('반복' * 3)일 경우에는 '반복 반복 반복'으로 결과가 나타남.

- 여러 줄을 처리할 때는 삼중 따옴표 (' ' ' ' ' ')를 사용함.

- 코드 내부에 문법과 상관 없는 파일 이름이나 소스 설명 등을 담을 때 주석 (comments)을 사용함. 파이썬 주석은 #으로 시작하고 그 줄의 끝까지 유효함. #주석은 한 줄만 가능하며, 삼중 따옴표는 여러 줄 주석에 사용함. 단, 대화형에서는 제외함.

1.3 정수와 실수의 이해

- 숫자는 간단히 정수와 실수로 나눔. 15, 7, 20 등은 정수, 소수점이 있는 3.14, 2.718 등은 실수.

- 실수는 문자 e를 사용해 지수승으로 표현할 수 있음. 대문자 E도 가능.

1.4 정수와 실수의 다양한 연산

- 파이썬 셸은 간단한 계산기로 사용할 수 있음. 수의 연산인 더하기, 빼기, 곱하기, 나누기의 사칙연산이 가능. 다만 곱하기는 *, 나누기는 /를 사용. 이러한 연산자를 모두 산술 연산자라 부름.

- 나누기 결과는 항상 실수이므로 0으로 나누면 실행 오류(ZeroDivisionError) 발생.

- 정수 나눗셈(floor division) 연산자(//)는 나눈 몫이 결과. 따라서 나누기 양수에서는 나누기 / 연산에서 소수부 없이 정수만 남음. (일종의 버림 연산)

- 나머지 연산자(%)는 나눈 나머지가 결과. 예를 들어 $5\%3$ 은 2.
- 지수승 연산자(**)는 거듭제곱을 계산. 예를 들어 $5 ** 3$ 은 125.
- 이러한 계산 시, 연산자에는 먼저 실행되는 우선순위가 있음. 괄호계산, **, 부호의 $+-$, $*$, $/$, $//$, $\%$, $+-$ 순서.
- 종종 계산 순서가 반대인 연산자들이 있으니 이 점은 유의하여 사용할 것.
- 대화형 모드에서 마지막에 실행된 결과값은 특별한 저장 공간인 _에 대입. 숫자, 문자열 등 유효한 연산식이면 모두 가능.
- 함수 eval()은 실행 가능한 연산식 문자열을 실행한 결과를 반환함.

2. 변수와 키워드, 대입 연산자

2.1 자료형

- 정수와 실수, 문자열 등을 자료형이라 함. 정수는 int, 실수는 float, 문자열은 str로 사용. 대화형 모드에서 자료형을 직접 알아보려면 type함수를 사용.

2.2 변수와 대입연산자

- 변수란 변하는 자료를 저장하는 메모리 공간. 자료를 잠을 수 있는 그릇이며 그릇의 이름인 태그가 붙어 있다고 생각하면 이해가 쉬움. 영문자와 숫자, _를 사용할 수 있으며 제한이 있음. 이를 주의 바람.
- 값을 변수에 저장하기 위해서는 대입 연산자(=)가 필요. =의 의미는 왼쪽 화

살표, 오른쪽 값을 왼쪽 변수에 저장하라는 의미. 대입 연산자는 수학에서의 =와 다름, 수학의 의미는 ==로 사용.

- 프로그래밍 언어 문법에서 사용하는 이미 예약된 단어를 키워드라 함. 총 개수는 33개.

3. 자료의 표준 입력과 자료 변환 함수

3.1 표준 입력과 다양한 변환 함수

- 셸이나 콘솔에서 사용자의 입력을 받아 처리하는 방식을 표준 입력이라 함.

- input으로 표준 입력을 문자열로 받아 반환 가능

- str()은 정수와 실수를 문자열로, int()는 정수 형태의 문자열을 정수로, float는 소수점이 있는 실수형태의 문자열을 실수로 반환. 만일 자료값이 다를 시에는 오류가 일어남.

3.2 16진수, 10진수, 8진수, 2진수의 활용

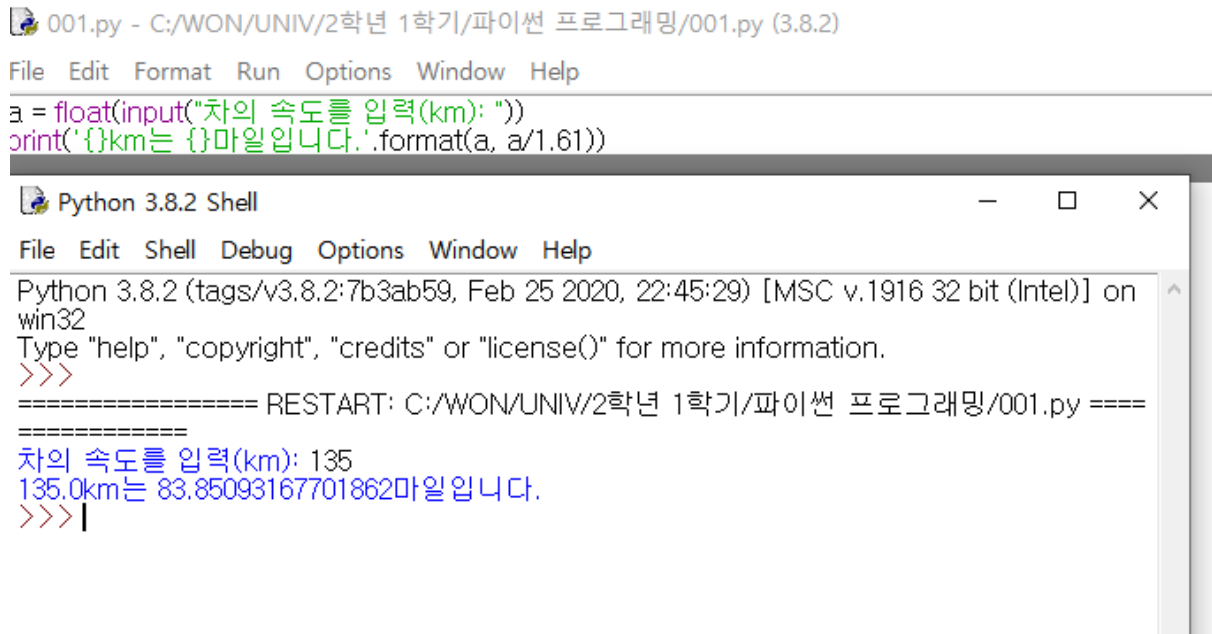
- 16진수는 0x, 8진수는 0o, 2진수는 0b로 시작. 대소문자 구분을 하지 않음.

- 10진수를 바로 16, 8, 2진수로 변환할 때는 bin, oct, hex 함수를 사용.

- int('숫자', 진수)를 입력하면 10진수의 정수로 바꿀 수 있음.

과제물 [2주차] 도전! 프로그래밍 (68, 69pg)

② 킬로미터 단위로 거리를 입력받아 마일 단위로 변환해 출력하는 프로그램을 작성하시오.



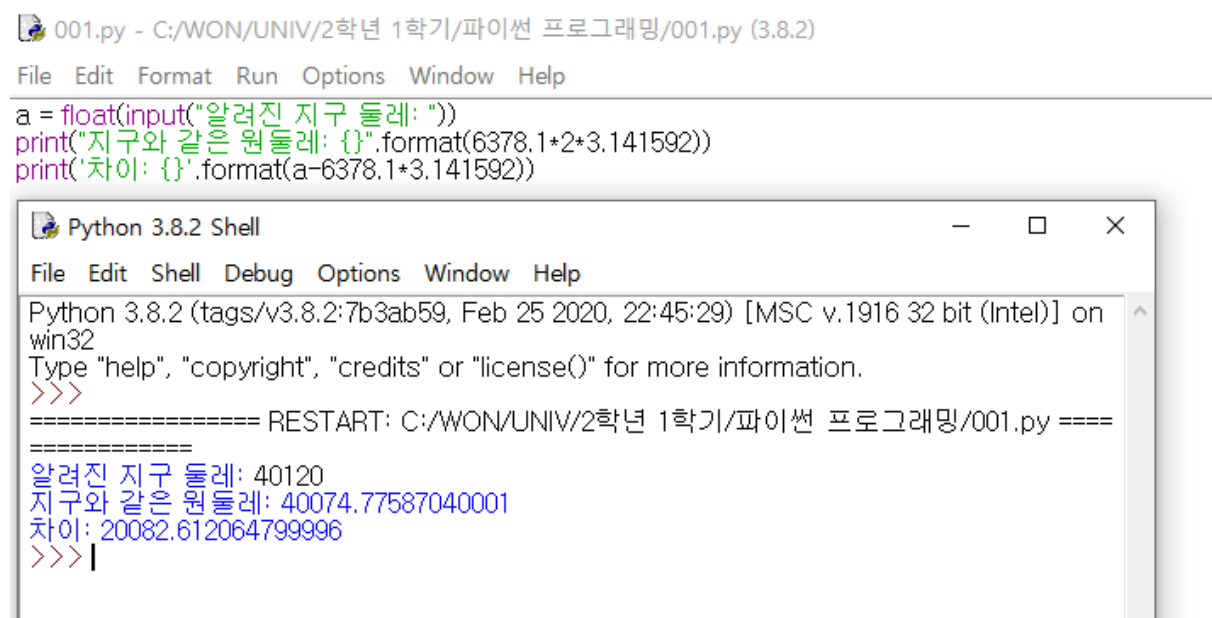
The screenshot shows a Python 3.8.2 IDE window titled "001.py - C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/001.py (3.8.2)". The code in the editor is:

```
a = float(input("차의 속도를 입력(km): "))
print('{}km는 {}마일입니다.'.format(a, a/1.61))
```

Below the editor is a "Python 3.8.2 Shell" window showing the execution output:

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/001.py =====
>>> 차의 속도를 입력(km): 135
135.0km는 83.85093167701862마일입니다.
>>> |
```

④ 다음 조건을 참고해 지구를 원이라고 보고 원의 둘레를 계산해 실제와의 차이를 알아보는 프로그램을 작성하시오.



The screenshot shows a Python 3.8.2 IDE window titled "001.py - C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/001.py (3.8.2)". The code in the editor is:

```
a = float(input("알려진 지구 둘레: "))
print("지구와 같은 원둘레: {}".format(6378.1*2*3.141592))
print('차이: {}'.format(a-6378.1*3.141592))
```

Below the editor is a "Python 3.8.2 Shell" window showing the execution output:

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/001.py =====
>>> 알려진 지구 둘레: 40120
지구와 같은 원둘레: 40074.77587040001
차이: 20082.612064799996
>>> |
```

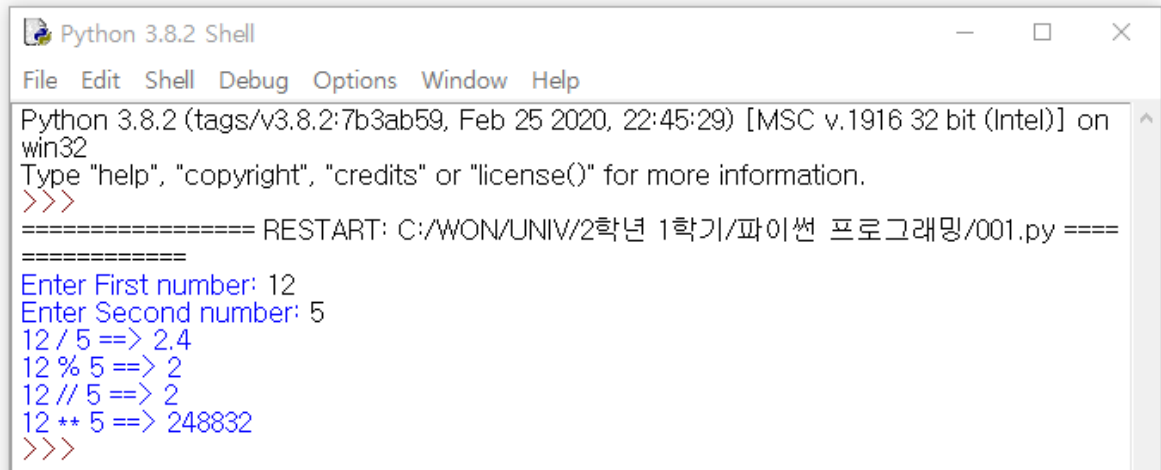
⑥ 두 정수를 입력받은 후 산술 연산 /, %, //, ** 4개를 수행해 결과를 출력하는

프로그램을 작성하시오.

001.py - C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/001.py (3.8.2)

File Edit Format Run Options Window Help

```
a = int(input("Enter First number: "))
b = int(input("Enter Second number: "))
print("{} / {} ==> {}".format(a, b, a/b))
print("{} % {} ==> {}".format(a, b, a%b))
print("{} // {} ==> {}".format(a, b, a//b))
print("{} ** {} ==> {}".format(a, b, a**b))
```



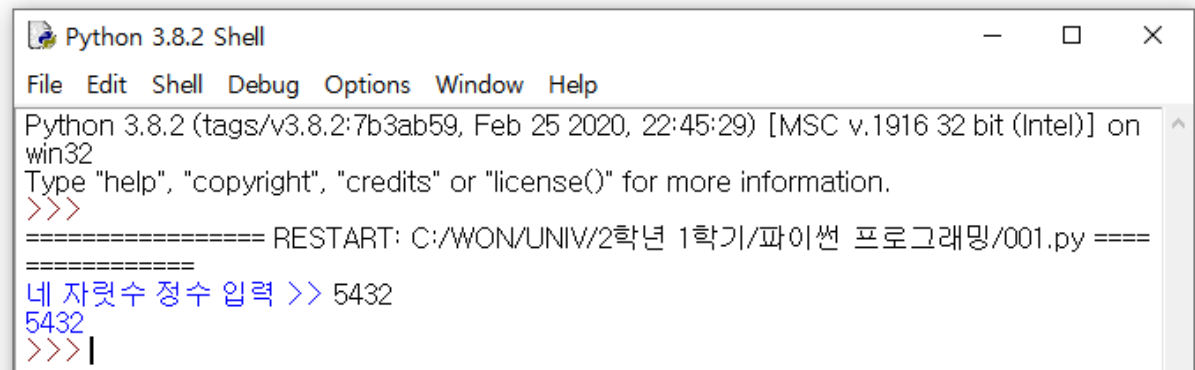
```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/001.py =====
Enter First number: 12
Enter Second number: 5
12 / 5 ==> 2.4
12 % 5 ==> 2
12 // 5 ==> 2
12 ** 5 ==> 248832
>>>
```

⑧ 네 자릿수 정수를 입력받은 후 그 정수를 역순으로 출력하는 프로그램을 작성하시오.

001.py - C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/001.py (3.8.2)

File Edit Format Run Options Window Help

```
num = int(input("네 자릿수 정수 입력 >> "))
a = int(num/1000)
b = int((num-a*1000)/100)
c = int((num-a*1000-b*100)/10)
d = int(num-a*1000-b*100-c*10)
print("{}{}{}{}".format(a, b, c, d))
```



```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/001.py =====
네 자릿수 정수 입력 >> 5432
5432
>>> |
```


CHAPTER 03. 일상에서 활용되는 문자열과 논리 연산

1. 문자열 다루기

1.1 문자열의 str 클래스와 부분 문자열 참고 슬라이싱

- 파이썬에서의 문자열은 문자의 나열, 텍스트 시퀀스(text sequence)라고도 함. 문자열의 자료형은 `class str.`, 문자열 상소는 `class str`의 객체.
- 함수 `len()`으로 문자열의 길이를 구할 수 있음.
- 문자열을 구성하는 문자는 0부터 시작되는 첨자(index)를 대괄호 안에 기술해 참조(indexing)가 가능함. `-1`, `-2` 등 작아지는 첨자도 역순으로 참조. 첨자가 유효 범위를 벗어나면 `IndexError`가 발생.

1.2 문자열의 부분 문자열 참조 방식

- 문자열에서 일부분을 참조하는 방법을 슬라이싱(slicing)이라고 함. 문자열에서 잘라 일부분을 참조한다고 생각하면 쉬움. 기존의 문자열은 변화 없음. `[start:end]`를 사용하여 start번째의 글자부터 end번째 글자까지의 문자열을 반환.
- 이는 음수도 사용이 가능하며, start번째의 글자부터 `end-1`까지의 문자열을 반환.
- 순방향의 0, 양수, 역방향의 음수를 혼합하여 사용하는 것도 가능.
- start와 end로 구성하는 문자열이 없다면 아무것도 없는 빈 문자열 반환.

- start가 없다면 처음부터, end가 없다면 끝까지를 의미. 앞뒤를 모두 비우면 문자열 전체 반환.

- start:end:step으로 문자 사이의 간격 조정 가능. Step이 1일 경우에는 생략. 음수도 가능하나, start는 end보다 작아야 함.

1.3 문자 함수와 이스케이프 시퀀스

- 파이썬은 유니코드 문자 사용, 한글을 비롯한 세계의 언어를 사용하는데 아무런 지장이 없음. ord()로 유니코드 번호를 알 수 있으며, chr()로 유니코드 번호를 호출하여 문자를 반환할 수 있음.

- 역슬래시(\)로 시작하는 조합으로 표현하는 문자를 이스케이프 시퀀스 문자 (escape sequence characters)라고 함. \n, \', \" 등등이 있음.

- 이스케이프 시퀀스는 문자열 내부에도 사용할 수 있음.

- 내장 함수 min()과 max는 인자의 최소, 최대값을 반환하는 함수. 인자가 문자열 1개이면 문자열 내의 문자 중에서 최소, 최대인 문자 반환, 문자열 2개 이상이면 문자열 중 최소, 최대인 문자열 반환.

2. 문자열 관련 메소드

2.1 문자열 대체와 부분 문자열 출현 횟수, 문자열 삽입

- 클래스에 소속된 함수를 메소드(method)라 함. 메소드의 호출은 문자열 객체명.함수 이름(인자)와 같이 사용.

- 문자열 객체가 str일 경우 str.replace(a, b)를 사용해 str 안의 a내용을 b로 바꿀 수 있음.

- 문자열 str에서 문자나 부분 문자열의 출현 횟수를 알려면 메소드 `str.count(부분 문자열)`을 사용해야 함.

- 문자열의 문자와 문자 사이에 원하는 문자열을 삽입하려면 메소드 `join()`을 사용,

2.2 문자열 찾기

- 클래스 str에서 부분 문자열 sub가 맨 처음에 위치한 첨자를 반환받으려면 메소드 `str.find(sub)`나 `str.index(sub)`를 사용. 메소드 `index`는 찾는 문자열이 없을 경우 `ValueError`를 발생시키지만 `find()`는 오류를 발생시키지 않고 `-1`을 반환.

- `rfind`와 `rindex`는 역순으로 부분 문자열을 찾아 첨자를 반환.

2.3 문자열 나누기

- 문자열 메소드 `str.split()`는 문자열 str에서 공백을 기준으로 문자열을 나눔. 만일 괄호 안에 특정 값이 있다면 그 특정 값을 기준으로 문자열을 구분.

- `split` 메소드를 사용하면 표준 입력 input에서 여러 값을 입력받을 수 있음.

2.4 다양한 문자열 변환 메소드

- 문자열 변환 메소드는 매우 다양.

- `.upper`, `.lower`, `.capitalize`, `.title`, `.swapcase` 등.

- 폭을 지정한 다음 중앙에 문자열을 배치할 때는 `.center()`를 사용함. 괄호 안에는 폭이 들어감.
- `ljust`, `rjust`는 각각 좌측 정렬, 우측 정렬, 괄호 안에는 폭이 들어감.
- `strip`메소드는 앞뒤의 공백을 제거. `Lstrip`, `rstrip`, `strip`, `strip(문자)`로 사용 가능.
- `zfill(폭)`을 이용하면 폭의 앞 빈 부분에 0을 채워넣을 수 있음.

2.5 출력을 정형화하는 함수 `format()`

- 문자열의 `format`을 이용하여 출력 처리를 간결하게 할 수 있음. 예를 들어 `str = '{} + {} = {}'.format(3, 4, 3+4)`라면 `str = '3 + 4 = 7'`이 됨.

2.6 C언어의 포매팅 스타일인 `%d`와 `%f`등으로 출력

- 프로그래밍 언어 C의 `printf()`에서 사용하는 형식 지정자 `%d`와 같은 스타일도 지원.

3. 논리 자료와 다양한 연산

3.1 논리 값과 논리 연산

- 파이썬은 논리 값으로 참과 거짓을 의미하는 `true`와 `false`를 키워드로 제공, 이러한 논리 값의 자료형을 `bool`로 제공.
- 논리 값 `true`와 `false`는 `int`로 각각 1, 0으로 변환된다.
- 논리 곱인 `and`와 `&`는 두 항이 모두 참이어야 `true`, `or`와 `|`는 두 항이 모두

거짓이어야 false.

- 배타적 논리합 연산자인 ^은 두 항이 다르면 true, 같으면 false.
- 연산자 not은 뒤에 위치한 논리 값을 바꿈.
- 논리 연산은 not, and, or 순서.

3.2 관계 연산

- 관계 연산자는 수나 문자열 등의 크기 비교에 사용되는 연산자, 결과는 논리 값.
- 파이썬은 논리 값 true와 false를 각각 1, 0으로 산술 연산에 활용할 수 있음.

3.3 멤버십 연산 in

- 파이썬 키워드인 in은 멤버의 소속을 알 수 있는 멤버십 연산, true와 false의 논리값. Not in 문은 부분 문자열이 아니면 true 반환.

3.4 비트 논리 연산

- 비트 연산은 정수로 저장된 메모리에서 비트와 비트의 연산을 말함. 비트 논리 연산을 이해하려면 피 연산자인 정수를 2진수 비트로 변환해 비트와 비트 연산을 수행한 수 다시 그 결과를 10진수로 변환, 즉, 결과도 정수. True를 1, false를 0으로 생각하면 동일.

3.5 비트 이동 연산

- 비트 이동 연산자는 연산자의 방향인 오른쪽, 또는 왼쪽으로 비트 단위로 피연산자인 지정된 횟수만큼 이동시키는 연산자. 빈 자리는 0으로 채워짐.
- 파이썬 연산자는 지수, 단항, 산술, 비트, 관계, 비교, 논리 연산 순서로 연산됨.

과제물 [3주차] 도전! 프로그래밍 (114, 115pg)

② 다섯 문자 이상의 문자열을 입력받아 다음과 같이 입력된 전체 문자열의 출력을 시작으로 결과 예시에 보이는 다양한 부분 문자열을 출력하는 프로그램을 작성하시오.

```
===== RESTART: C:\WONWUNIVW2학년 1학기\파이썬 프로그래밍\W03-02.py =====
>>>
다섯 이상의 문자열 입력 >>>Python String Slicing
입력 문자열: Python String Slicing
첫 문자: P
마지막 문자: g
첫 문자를 제외한 부분 문자열: ython String Slicing
마지막 문자를 제외한 부분 문자열: Python String Slicin
맨 앞과 뒤의 두 문자씩을 제외한 부분 문자열: ython String Slicin
문자 하나씩을 건너뛰 부분 문자열: Pto tigSiig
역문자열: gnicilS gnirtS nohtyP
>>>
```

Ln: 102 Col: 1

```
03-02.py - C:\WONWUNIVW2학년 1학기\파이썬 프로그래밍\W03-02.py (3.8.2)
File Edit Format Run Options Window Help
a = input('다섯 이상의 문자열 입력 >>')
print('입력 문자열: {}'.format(a))
b = list(a)
c = len(a)
d = len(b)
print('첫 문자: {}'.format(b[0]))
print('마지막 문자: {}'.format(b[d-1]))
print('첫 문자를 제외한 부분 문자열: {}'.format(a[1:c]))
print('마지막 문자를 제외한 부분 문자열: {}'.format(a[0:c-1]))
print('맨 앞과 뒤의 두 문자씩을 제외한 부분 문자열: {}'.format(a[1:c-1]))
print('문자 하나씩을 건너뛰 부분 문자열: {}'.format(a[0:c:2]))
print('역문자열: {}'.format(a[::-1]))
```

④ 다음 파이썬 튜토리얼 페이지 주소 문자열에서 메소드 find()와 rfind() 그리고 문자열 슬라이싱을 활용해 부분 문자열 https, docs.python.org, tutorial을 출력하는 프로그램을 작성하시오.

```
===== RESTART: C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/03-04.py =====
https
docs.python.org
tutorial
>>>
```

03-04.py - C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/03-04.py (3.8.2)

File Edit Format Run Options Window Help

```
url = 'https://docs.python.org/3/tutorial'
```

```
a = url.find('/')

```

```
b = url.split('{}'.format(url[a]))

```

```
print(b[0].replace(':', ''))

```

```
print(b[2])

```

```
print(b[4])
|
```

⑥ 실수 2개를 포준 입력으로 받아 다음 결과 예시와 같이 6개의 관계 연산의 결과를 출력하는 프로그램을 작성하시오.

```
===== RESTART: C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/03-06.py =====
```

```
실수 두 개 입력 >> 5.4 2.7
```

```
5.4 > 2.7 결과: True
```

```
5.4 >= 2.7 결과: True
```

```
5.4 < 2.7 결과: False
```

```
5.4 >= 2.7 결과: True
```

```
5.4 == 2.7 결과: False
```

```
5.4 != 2.7 결과: True
```

```
>>>
```

```
>>>
```

03-06.py - C:/WON/UNIV/2학년 1학기/파이썬 프로그래밍/03-06.py (3.8.2)

File Edit Format Run Options Window Help

```
a, b = input('실수 두 개 입력 >> ').split( )
```

```
print('{} > {} 결과: {}'.format(a, b, a>b))
```

```
print('{} >= {} 결과: {}'.format(a, b, a>=b))
```

```
print('{} < {} 결과: {}'.format(a, b, a<b))
```

```
print('{} >= {} 결과: {}'.format(a, b, a>=b))
```

```
print('{} == {} 결과: {}'.format(a, b, a==b))
```

```
print('{} != {} 결과: {}'.format(a, b, a!=b))
```

⑧ 정수 하나와 2의 지수승으로 사용할 정수를 입력받아 연산자 <<와 **를 사용해 각각 계산한 결과가 같음을 보이는 프로그램을 작성하시오.

CHAPTER 04. 일상생활과 비유되는 조건과 반복

1. 조건에 따른 선택 if... else

1.1 조건의 논리 값에 따른 선택 if

– 조건에 따라 문장들을 처리하는 경우 if문을 사용. 결과가 true이면 이어서 진행, 아닐 시에는 실행하지 않음.

1.2 조건에 따라 하나를 선택하는 if... else

– 조건 if의 논리 표현식 결과는 true 아니면 false이며, if else문에서 else는 결과가 false일 때 실행.

1.3 여러 조건 중에서 하나를 선택하는 구문 if... elif

– if, elif, elif, else로 각 조건에 맞는 것으로 이동되며, 아무것도 아닐 경우에는 else로 실행.

1.4 중첩된 조건

2. 반복을 제어하는 for문과 while문

2.1 시퀀스의 내부 값으로 반복을 실행하는 for문

– 반복문에서 내장 함수 range를 활용하면 간결함.

2.2 횟수를 정하지 않은 반복에 적합한 while반복

– while문은 for문에 비해 간결, 반복 조건인 논리 표현식의 값에 따라 반복 수행. 횟수를 정해놓지 않고 어떤 조건이 false가 될 때 까지 반복을 수행하는데 적합.

2.3 중첩된 반복

3. 임의의 수인 난수와 반복을 제어하는 break문, continue문

3.1 임의의 수인 난수 발생과 반복에 활용

– 파이썬에서는 random, randint(시작, 끝)을 사용해 정수 시작과 끝 수 사이에서 임의의 정수를 얻을 수 있음.

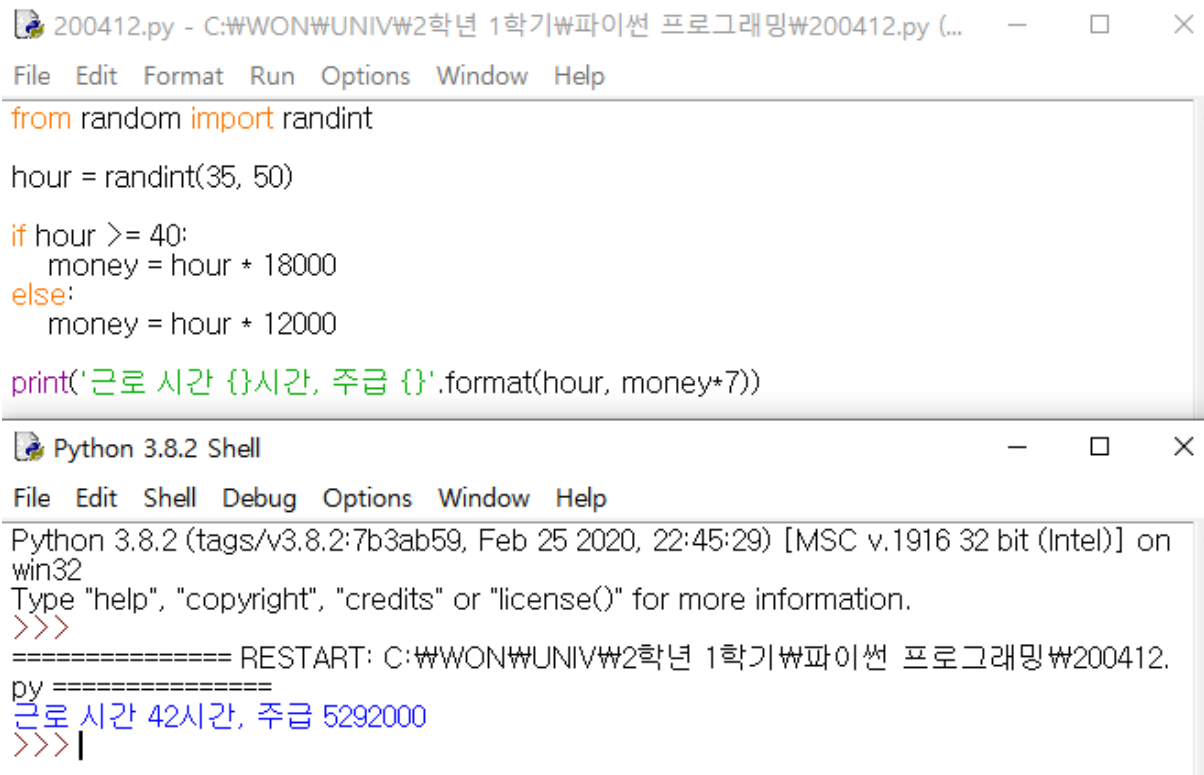
3.2 반복을 제어하는 break문과 continue문

– while의 논리 표현식이 true라면 무한히 반복, 이를 무한 반복이라고 한다. for이나 while반복 내에서 break는 else를 실행하지 않고 무조건 종료.

– continue 문장은 이후의 반복 몸체를 실행하지 않고 당므 반복을 위해 논리 조건을 수행.

과제물 [4주차] 도전! 프로그래밍 (156, 157pg)

② 근로 시급이 12000원이고 일주일에 40시간 이상 근무하면 시급의 1.5배의 급여를 준다고 한다. 일주일 근로 시간을 35~50시간 사이에서 임의의 난수로 정하고, 주급을 계산해 출력하는 프로그램을 작성하시오.



The image shows a screenshot of a Python IDE. The top window, titled '200412.py - C:\WON\UNIV\2학년 1학기\파이썬 프로그래밍\200412.py (...)', contains the following Python code:

```
from random import randint

hour = randint(35, 50)

if hour >= 40:
    money = hour * 18000
else:
    money = hour * 12000

print('근로 시간 {}시간, 주급 {}'.format(hour, money*7))
```

The bottom window, titled 'Python 3.8.2 Shell', shows the execution output:

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\WON\UNIV\2학년 1학기\파이썬 프로그래밍\200412.py =====
근로 시간 42시간, 주급 5292000
>>> |
```

④ 다음을 참고해 인간의 비만도를 측정하는 체질량 지수를 계산해 판정 결과를 출력하는 프로그램을 작성하시오.

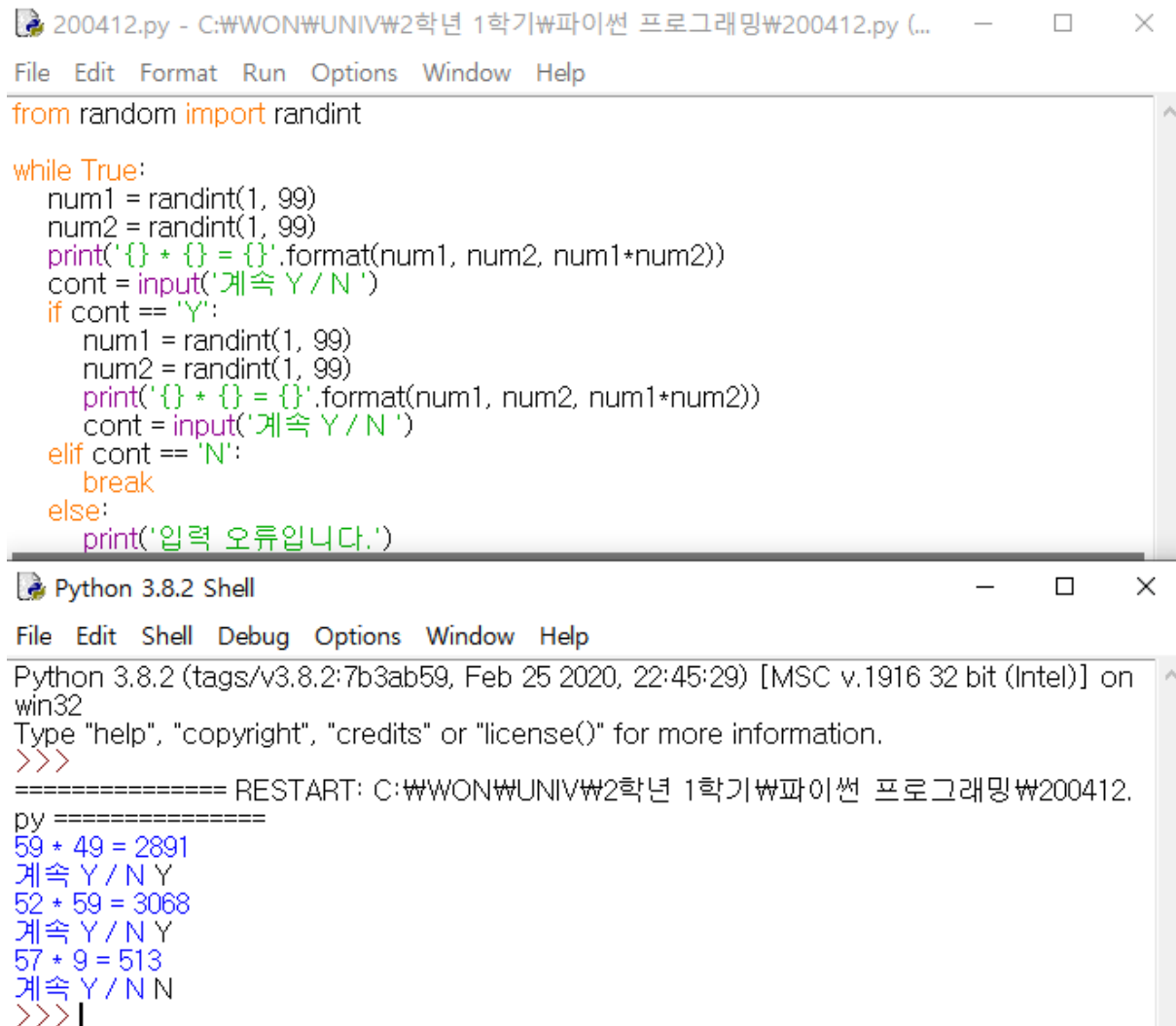
```
200412.py - C:\WONWUNIV\2학년 1학기\파이썬 프로그래밍\200412.py (... - □ ×
File Edit Format Run Options Window Help
h, w = input('당신의 키(cm)와 몸무게(kg(는? >>')).split( )
h = float(h)
w = float(w)

bmi = w / ((h * h) / 10000)

if bmi >= 40:
    print('키: {}cm, 몸무게: {}'.format(h, w))
    print('BMI: {}, 고도비만'.format(bmi))
elif bmi >= 35:
    print('키: {}cm, 몸무게: {}'.format(h, w))
    print('BMI: {}, 중등도 비만'.format(bmi))
elif bmi >= 30:
    print('키: {}cm, 몸무게: {}'.format(h, w))
    print('BMI: {}, 비만'.format(bmi))
elif bmi >= 25:
    print('키: {}cm, 몸무게: {}'.format(h, w))
    print('BMI: {}, 과체중'.format(bmi))
elif bmi >= 18.5:
    print('키: {}cm, 몸무게: {}'.format(h, w))
    print('BMI: {}, 정상'.format(bmi))
else:
    print('키: {}cm, 몸무게: {}'.format(h, w))
    print('BMI: {}, 저체중'.format(bmi))

Python 3.8.2 Shell - □ ×
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on
win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\WONWUNIV\2학년 1학기\파이썬 프로그래밍\200412.
py =====
당신의 키(cm)와 몸무게(kg(는? >>171 72
키: 171.0cm, 몸무게: 72.0
BMI: 24.622960911049553, 정상
>>>
```

⑥ 다음을 참고해 1에서 99까지 정수인 난수를 2개 생성하고 곱하기의 결과를 출력하는 프로그램을 작성하시오.



The image shows a screenshot of a Python IDE with two windows. The top window, titled '200412.py', contains a Python script that generates random numbers and calculates their product. The script uses a while loop to repeatedly generate two random numbers between 1 and 99, calculate their product, and prompt the user to continue (Y) or stop (N). If the user enters 'N', the program prints an error message. The bottom window, titled 'Python 3.8.2 Shell', shows the execution of the script. It displays the product of 59 and 49 (2891), followed by the user input 'Y'. Then it shows the product of 52 and 59 (3068), followed by 'Y'. Next, it shows the product of 57 and 9 (513), followed by 'N'. The prompt '>>>' is shown at the end of the execution.

```
200412.py - C:\WON\UNIV\2학년 1학기\파이썬 프로그래밍\200412.py (...)
```

```
File Edit Format Run Options Window Help
```

```
from random import randint
```

```
while True:
```

```
    num1 = randint(1, 99)
```

```
    num2 = randint(1, 99)
```

```
    print('{} * {} = {}'.format(num1, num2, num1*num2))
```

```
    cont = input('계속 Y / N ')
```

```
    if cont == 'Y':
```

```
        num1 = randint(1, 99)
```

```
        num2 = randint(1, 99)
```

```
        print('{} * {} = {}'.format(num1, num2, num1*num2))
```

```
        cont = input('계속 Y / N ')
```

```
    elif cont == 'N':
```

```
        break
```

```
    else:
```

```
        print('입력 오류입니다.')
```

```
Python 3.8.2 Shell
```

```
File Edit Shell Debug Options Window Help
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on
```

```
win32
```

```
Type "help", "copyright", "credits" or "license()" for more information.
```

```
>>>
```

```
===== RESTART: C:\WON\UNIV\2학년 1학기\파이썬 프로그래밍\200412.
```

```
py =====
```

```
59 * 49 = 2891
```

```
계속 Y / N Y
```

```
52 * 59 = 3068
```

```
계속 Y / N Y
```

```
57 * 9 = 513
```

```
계속 Y / N N
```

```
>>> |
```

⑧ 난수로 발생시킨 1에서 100사이의 첫 번째 피연산자와 사용자가 표준 입력한 산술 연산자 문자 그리고 표준 입력한 두 번째 피연산자를 계산해 출력하는 프로그램을 작성하시오.

```
from random import randint

while True:
    num1 = randint(1, 99)
    print('첫 값은 {}이다.'.format(num1))
    a = input('산술 연산의 종류를 입력하십시오 >> ')
    num2 = input('두 번째 피연산자를 입력하십시오 >> ')
    if a == '+':
        print('{} {} {} = {}'.format(num1, a, num2, num1+num2))
        num1 = randint(1, 99)
        print('첫 값은 {}이다.'.format(num1))
        a = input('산술 연산의 종류를 입력하십시오 >> ')
        num2 = ('두 번째 피연산자를 입력하십시오 >> ')
    elif a == '*':
        num2 = ('두 번째 피연산자를 입력하십시오 >> ')
        print('{} {} {} = {}'.format(num1, a, num2, num1*num2))
        num1 = randint(1, 99)
        print('첫 값은 {}이다.'.format(num1))
        a = input('산술 연산의 종류를 입력하십시오 >> ')
        num2 = ('두 번째 피연산자를 입력하십시오 >> ')
    elif a == '-':
        num2 = ('두 번째 피연산자를 입력하십시오 >> ')
        print('{} {} {} = {}'.format(num1, a, num2, num1-num2))
        num1 = randint(1, 99)
        print('첫 값은 {}이다.'.format(num1))
        a = input('산술 연산의 종류를 입력하십시오 >> ')
        num2 = ('두 번째 피연산자를 입력하십시오 >> ')
    elif a == '/':
        num2 = ('두 번째 피연산자를 입력하십시오 >> ')
        print('{} {} {} = {}'.format(num1, a, num2, num1/num2))
        num1 = randint(1, 99)
        print('첫 값은 {}이다.'.format(num1))
        a = input('산술 연산의 종류를 입력하십시오 >> ')
        num2 = ('두 번째 피연산자를 입력하십시오 >> ')
    else a == '&':
        break
```

CHAPTER 05. 항목의 나열인 리스트와 튜플

1. 여러 자료 값을 편리하게 처리하는 리스트

1.1 리스트의 개념과 생성

– 여러 항목을 하나의 단위로 묶어 손쉽게 사용하는 복합 자료형을 제공하는데, 그 중 대표적인 것이 리스트(list).

- 리스트는 대괄호 사이에 항목을 기술. Print(리스트 이름)으로 리스트 전체 출력 가능.

1.2 리스트의 항목 참조

- 리스트에서 첨자를 사용해 항목 하나하나를 참조할 수 있음, 첨자는 정수이며, 문자열에서의 첨자 방식과 동일.

1.3 리스트의 항목 수정

- 리스트의 메소드 count는 값을 갖는 항목의 수, index는 인자인 값의 항목이 위치한 첨자를 반환. 동일한 값이 여러 개이면 첫 번째로 나타난 위치의 첨자.
- 리스트의 첨자를 이용한 항목을 대입 연산자의 오른쪽에 위치시켜 리스트의 항목을 수정할 수 있음.

1.4 리스트 내부에 다시 리스트를 포함하는 중첩 리스트

- 리스트 내부에 다시 리스트가 항목으로 올 수 있음. 중첩된 리스트는 for문을 이용하여 참조할 수 있음.

2. 리스트의 부분 참조와 항목의 삽입과 삭제

2.1 리스트의 부분 참조와 항목의 삽입과 삭제

- 리스트도 전체나 일부분을 참조할 수 있음. 리스트[start:end:step]와 같이 사용, start에서 end-1까지 step 간격의 요소로 구분된 부분 리스트 반환.

- start, end는 생략하면 처음부터 마지막 항목까지 참조. Step을 생략하면 1.

2.2 리스트의 부분 수정

- 리스트의 일부분을 다른 리스트로 수정하려면 슬라이스 방식에 대입.
- 리스트의 슬라이스에 동일한 리스트의 슬라이스를 대입해도 아무런 문제 없음. 다른 리스트의 슬라이스를 대입해도 상관 없음.

2.3 리스트의 항목 삽입과 삭제

- 리스트의 첨자 위치에 항목을 삽입하려면 리스트.insert(첨자, 항목)을 이용. 삽입되는 항목은 무엇이든 가능, 빈 리스트에도 삽입 가능.
- 리스트에서 하나의 항목을 삭제하려면 메소드 remove(값), pop(첨자), pop을 사용. 메소드 remove(값)은 리스트에서 지정된 값의 항목을 삭제. 그러나 삭제할 값이 리스트에 없다면 오류 발생.
- 인자가 없는 pop()은 마지막 항목을 삭제하고 삭제된 값 반환, 첨자를 사용한 pop(첨자)는 지정된 첨자의 항목을 삭제하고 반환. 그러므로 pop의 호출로 삭제된 값을 대입하거나 출력할 수 있음.
- 문장 del은 뒤에 위치한 변수나 항목을 삭제. 변수 자체를 메모리에서 제거하므로 참조 불가.
- .clear로 리스트의 모든 항목 삭제 가능.

2.4 리스트의 추가, 연결과 반복

- 리스트 메소드 `리스트.extend(list)`는 리스트에 인자인 `list`를 가장 뒤에 추가.
- 더하기 연산자 `+`는 리스트를 연결해줌.
- 리스트와 정수와 `*`로 곱하면 항목이 지정된 정수만큼 반복된 리스트를 반환.

2.5 리스트 항목의 순서와 정렬

- `reverse()`는 항목 순서를 반대로 뒤집음.
- `sort()`는 리스트 항목의 순서를 오름차순으로 정렬. `Sort(reverse=True)`로 호출하면 내림차순으로 정렬.
- 내장 함수 `sorted`는 리스트의 항목 순서를 오름차순으로 정렬한 새로운 리스트 반환. 원래 리스트는 변하지 않음.

2.6 리스트 컴프리헨션

- 리스트를 함축하는 것을 리스트 컴프리헨션이라 함. 리스트를 만드는 간결한 방법 제공.
- 즉, 이를 사용하여 한 리스트의 모든 항목 각각에 대해 어떤 조건을 적용한 후 그 반환값을 항목으로 갖는 다른 리스트를 쉽게 만들 수 있음.
- 함축, 축약, 내포, 내장 등으로도 불림.

2.7 리스트 대입과 복사

- 리스트에서 완전히 새로운 리스트를 만들어 복사하려면 슬라이스 :나 copy, list 함수를 이용해야 함.
- 문장 is는 피연산자인 변수 두 개가 동일한 메모리를 공유하는지 검사. 같으면 true, 다르면 false

3. 항목의 순서나 내용을 수정할 수 없는 튜플

3.1 괄호로 정의하는 시퀀스 튜플

- 튜플은 문자열, 리스트와 같은 항목의 나열인 시퀀스. 리스트와 달리 항목의 순서, 내용 수정이 불가능.

- 빈 튜플은 tuple()로 만들.

- 튜플도 모두 콤마로 구분된 항목들의 리스트로 표현, 각각의 항목은 정수, 실수, 문자열, 리스트, 튜플 등 제한이 없음.

- 튜플은 괄호 사이에 항목을 기술, 괄호는 생략될 수 있음.

- 튜플도 리스트와 같이 첨자 참조와 슬라이스가 가능. 그러나 수정은 불가능.

3.2 튜플 연결과 반복, 정렬과 삭제

- 리스트와 같이 +와 *는 튜플을 연결하고 항목이 횟수만큼 반복된 튜플을 반환.
- 내장 함수 sorted(튜플)은 튜플 항목의 순서를 오름차순으로 정렬한 새로운

리스트를 반환, 따라서 반환 값을 리스트 변수에 대입해 사용할 수 있음.
Reverse 사용 가능.

CHAPTER 06. 일상에서 활용되는 문자열과 논리 연산

1. 키와 값인 쌍의 나열인 딕셔너리

1.1 딕셔너리의 개념과 생성

- 말 그대로 사전을 생각하면 이해하기 쉬움.
- 키와 값의 쌍인 항목을 나열한 시퀀스. 딕셔너리는 콤마로 구분된 항목들의 리스트로 표현.
- 딕셔너리는 중괄호로 생성, 변수에 대입 가능. 자료형은 클래스 dict.
- 변수 = {}로 빈 딕셔너리를 만들 수 있음.

1.2 다양한 인자의 함수 dict()로 생성하는 딕셔너리

- 내장 함수 dict함수에서 인자로 리스트나 튜플 1개를 사용해 딕셔너리를 만들 수 있음.
- 함수 dict의 리스트나 튜플 내부에서 일련의 키-값 쌍으로 [키, 값] 리스트 형식과 (키, 값) 튜플 형식을 모두 사용할 수 있음.
- l가 단순 문자열이면 키=값 항목 나열로도 지정할 수 있음.

1.3 딕셔너리 키는 수정 불가능한 객체로 사용

- 딕셔너리의 키는 수정 불가능한 객체는 모두 가능. 따라서 정수, 실수도 가능.
- 새로운 키로 대입하면 항상 새로운 키-값 항목이 삽입.
- 이미 있는 키로 항목을 참조하면 값이 반환.
- 수정 불가능한 튜플은 딕셔너리의 키로 사용될 수 있음. 그러나 수정 가능한 리스트는 키로 사용할 수 없음.

1.4 딕셔너리 항목의 순회

- 딕셔너리 메소드 `keys()`는 키로만 구성된 리스트를 반환. 그러므로 `for`문에서 시퀀스 위치에 메소드 `keys`를 사용하면 딕셔너리의 모든 항목을 참조하는 구문을 사용할 수 있음.
- 딕셔너리 메소드 `items()`는 (키, 값) 쌍의 튜플이 들어 있는 리스트를 반환. 각 튜플의 첫번째 항목은 키, 두번째 항목은 키값. 그러므로 `for`문에서 변수 위치에 키 값을 저장할 2개의 변수와 시퀀스 위치에 메소드 `items()`를 사용하면 딕셔너리의 모든 항목을 참조하는 간단한 구문을 사용할 수 있음.
- 딕셔너리 메소드 `values()`는 값으로 구성된 리스트를 반환.
- 반복`for`문에서는 시퀀스 위치에 있는 딕셔너리 변수만으로도 모든 키를 순회할 수 있음.

1.5 딕셔너리 항목의 참조와 삭제

- 딕셔너리 메소드 `get(키)`는 키의 해당 값을 반환. 메소드 `get(키)`는 딕셔너리에 키가 없어도 오류가 발생하지 않고 아무것도 없다는 의미인 `none`을 반환, 만

일 키 뒤에 다른 인자를 넣으면 딕셔너리에 키가 없을 때 이 지정된 값을 반환.

- 딕셔너리 메소드 `pop(키)`는 키인 항목을 삭제하고 삭제되는 키의 해당 값을 반환, 삭제할 키가 없다면 두 번째에 지정한 값이 반환. 인자로 키만 적었는데 삭제할 키가 없다면 `KeyError` 발생.

- 딕셔너리 메소드인 `popitem()`은 임의의 (키, 값)의 튜플을 반환하고 삭제. 만일 데이터가 하나도 없다면 오류가 발생.

- 딕셔너리를 문장 `del`에 이어 키로 지정하면 해당 항목이 삭제.

1.6 딕셔너리 항목의 전체 삭제와 변수 제거

- 딕셔너리 메소드 `clear()`는 기존의 모든 키:값 항목을 삭제.

- 딕셔너리를 문장 `del`에 이어 키로 지정하면 변수 자체가 메모리에서 제거. 따라서 제거 이후에는 변수를 참조할 수 없음.

1.7 딕셔너리 결합과 키의 멤버십 검사 연산자 `in`

- 딕셔너리의 메소드 `update(다른 딕셔너리)`는 인자인 다른 딕셔너리를 합병.

- 인자 딕셔너리에 원 딕셔너리와 동일한 키가 있다면 인자 딕셔너리의 값으로 대체.

- 문장 `in`으로 딕셔너리에 키가 존재하는지 간단히 검사할 수 있음. 값의 존재 여부는 확인할 수 없으므로 값으로 조회하면 항상 `false`. `Not in` 또한 가능.

2. 중복과 순서가 없는 집합

2.1 수학에서 배운 집합을 처리하는 자료형

– 집합은 중복되는 요소가 없으며 순서도 없는 원소의 모임(collection). 파이썬에서 집합은 수학과 같이 원소를 콤마로 구분하여 중괄호로 둘러싸 표현한다.

– 원소는 불변 값으로 중복될 수 없으며 서로 다른 값이어야 함, 즉, 원소는 중복을 허용하지 않으며 원소의 순서는 의미가 없음.

– 집합의 원소는 정수, 실수, 문자열, 튜플 등 수정 불가능한 것이어야 함, 리스트나 딕셔너리처럼 가변적인 것은 허용되지 않음. 집합의 원소는 중복을 허용하지 않으므로 멤버십 검사와 중복 제거에 주로 사용될 수 있음.

2.2 내장 함수 set()을 활용한 집합 생성

– 집합은 내장 함수 set()으로도 생성할 수 있음.

– set(원소로 구성된 리스트_or_튜플_or_문자열)

– 인자가 없으면 빈 집합인 공집합이 생성, 인자가 있으면 하나이며 리스트와 튜플, 문자열 등이 올 수 있음.

– 함수 set에서 인자는 리스트와 튜플, 문자열처럼 반복적이면 가능. 그러나 리스트나 튜플의 항목은 변할 수 없는 것이어야 함. 가변적인 것은 허용되지 않음.

– 내장 함수 set으로 만들어지는 집합 a는 공집합, 집합의 자료형 이름은 클래스 set.

- 빈 리스트와 튜플이 각 [], (), 빈 딕셔너리는 {}. 수학의 {}는 공집합이지만 파이썬의 {}은 빈 딕셔너리.

- 함수 set에서는 인자로 리스트, 튜플 자체를 사용할 수 있음, 결과는 시퀀스 항목에서 중복을 제거한 원소로 구성.

- 함수 set의 인자로 문자열이 사용되면 각각의 문자가 유니코드 집합이 생성, 단, 집합이므로 순서는 의미 없음.

- 함수 set의 인자에서 리스트나 튜플의 항목으로 수정될 수 있는 리스트나 딕셔너리는 허용되지 않음, TypeError 발생.

2.3 중괄호로 직접 원소를 내열해 집합 생성

- 집합을 생성하는 다른 방법은 중괄호 안에 직접 원소를 콤마로 구분해 나열하는 방법. 집합의 원소는 문자, 문자열, 숫자, 튜플과 같이 변할 수 없는 것이어야 함.

- 리스트나 딕셔너리와 같이 가변적인 것은 원소로 사용할 수 없음.

2.4 집합의 원소 추가와 삭제

- 원소의 추가는 add(원소).

- 원소의 삭제는 remove(원소), 삭제하려는 원소가 없으면 KeyError 발생. Discard(원소)로도 원소를 삭제할 수 있으며, 원소가 없어도 오류가 발생하지 않음. 임의의 원소를 삭제하려면 pop()을 사용해야 함.

- 집합의 모든 원소를 삭제하려면 메소드 clear()를 사용.

2.5 집합의 주요 연산인 합집합, 교집합, 차집합, 여집합

– 양 쪽 모든 원소를 합하는 합집합은 연산자 `|`와 메소드 `union`을 사용. `Union`은 합집합을 반환, `a` 수정. 메소드 `a.update(b)`도 합집합과 같은 효과, 합집합 결과가 호출하는 집합 `a`에 반영돼 수정.

– 양쪽 모든 집합에 속하는 원소로 구성되는 교집합은 연산자 `&`와 메소드 `intersection` 사용. 집합 `a`, `b`모두에 영향을 끼치지 않음.

– 차집합은 연산자 `-`와 메소드 `difference`를 사용. 피연산자의 순서에 따라 결과가 달라짐, 교환 법칙은 성립되지 않음.

– 산술 연산의 `+=`와 `*=`처럼 위에서 알아본 집합 연산 네 가지에도 축약 대입 연산자 제고. 합집합 `= update = |=`, 교집합 `= &=`, `intersection_update`, 차집합 `= -=` `= difference_update`, 여집합 `= ^=` `= symmetric_difference_update()`

2.6 함수 `len()`과 소속 연산 `in`

– 함수 `len()`으로 집합에 들어 있는 원소의 개수를 확인할 수 있음.

– 소속 연산자 `in`은 집합에서 유용하게 사용될 수 있음, 특정 원소가 집합에 있는지를 확인하기 위해 `in` 사용.