

2020학년도 1학기

학생 교과목 포트폴리오

학부	컴퓨터공학부
학과	컴퓨터정보공학과
과목명	파이썬프로그래밍
학번	20190703
성명	변은주



동양미래대학교
DONGYANG MIRAE UNIVERSITY

머리말

대학교에 온 후, 포트폴리오를 작성해보는 것이 처음이라 내가 만들고자 하는 포트폴리오의 방향이 옳은 방향인지 한참을 고민해보고 많은 시간을 들였다. 그럼, 본론으로 들어가기에 앞서 파이썬에 대한 간단한 설명을 듣고 가자.

프로그래밍 언어의 학습에 대한 순서라는 것은 정해져 있지 않지만 많은 이들이 파이썬으로 기본기를 먼저 잡을 것을 추천한다. 이는 파이썬이 배우기 쉽고, 강력한 프로그래밍 언어이기 때문이다. 파이썬은 효율적인 고수준 데이터 구조를 갖추고 있으며, 간단하지만 효과적인 객체 지향 프로그래밍 접근법 또한 갖추고 있다. 우아한 문법과 동적 타이핑, 그리고 인터프리팅 환경을 갖춘 파이썬은 다양한 분야, 다양한 플랫폼에서 사용될 수 있는 최적의 스크립팅, RAD(rapid application development - 빠른 프로그램 개발) 언어이다.

이 교과목의 담당교수님이신 강환수 교수님께서 해주신 말씀을 듣고 지금까지 내가 학습한 내용들을 다시한번 요약정리하여 이해가 가지 않았던 부분들을 하나하나 되짚어보며 작성했다. 만약 파이썬을 처음 접해보는 사람이 이 포트폴리오를 읽게 된다면 정말 많은 도움이 될 것이라고 조심스럽게 말해본다.

그럼 이제 본격적으로 파이썬에 대해 알아보자!

2020 학년도 1학기	전공	컴퓨터정보공학과	학부	컴퓨터공학부
과 목 명	파이썬프로그래밍(2019009-PC)			
강의실 과 강의시간	화 1(3-217), 2(3-217), 3(3-217)	학점	3	
교과분류	이론/실습	시수	3	

담당 교수	강환수 + 연구실 : 2호관-706 + 전 화 : 02-2610-1941 + E-MAIL : hskang@dongyang.ac.kr + 면담가능기간 : 화요일 13~16
-------	--

학과 교육목표				
과목 개요	2010년 이후 파이썬의 폭발적인 인기는 제4차 산업혁명 시대의 도래와도 밀접한 연관성이 있다. 컴퓨팅 사고력은 누구나가 가져야할 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.			
학습목표 및 성취수준	1. 컴퓨팅 사고력의 중요성을 인지하고 4차 산업혁명에서 파이썬 언어의 필요성을 이해할 수 있다. 2. 기본적인 파이썬 문법을 이해하고 데이터 처리를 위한 자료구조를 이해하여 적용할 수 있다. 3. 문제 해결 방법을 위한 알고리즘을 이해하고 데이터 처리에 적용할 수 있다. 4. 파이썬 프로그램을 이용하여 실무적인 코딩 작업을 할 수 있다.			
	도서명	저자	출판사	비고
주교재	파이썬으로 배우는 누구나 코딩	강환수, 신웅현	홀름과학출판사	
수업시 사용도구	파이썬 기본 도구, 파이참, 아나콘다와 주피터 노트북			
평가방법	중간고사 30%, 기말고사 40%, 과제물 및 퀴즈 10% 출석 20%(학교 규정, 학업성적 처리 지침에 따름)			
수강안내	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.			

2020 학년도 1학기	전공	컴퓨터정보공학과	학부	컴퓨터공학부
과 목 명	파이썬프로그래밍(2019009-PC)			
강의실 과 강의시간	화.1(3-217), 2(3-217), 3(3-217)		학점	3
교과분류	이론/실습		시수	3

담당 교수	강환수 + 연구실 : 2호관-706 + 전 화 : 02-2610-1941 + E-MAIL : hskang@dongyang.ac.kr + 면담가능기간 : 화요일 13~16			
-------	--	--	--	--

학과 교육목표				
과목 개요	2010년 이후 파이썬의 폭발적인 인기는 제4차 산업혁명 시대의 도래와도 밀접한 연관성이 있다. 컴퓨팅 사고력은 누구나가 가져야할 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 매우중요한 언어가 되었다. 본 교과목은 파이썬 프로그래밍의 기초적이고 체계적인 학습을 수행한다. 본 교과목을 통하여 데이터 처리 방법에 대한 효율적인 파이썬 프로그래밍 방법을 학습한다.			
학습목표 및 성취수준	1. 컴퓨팅 사고력의 중요성을 인지하고 4차 산업혁명에서 파이썬 언어의 필요성을 이해할 수 있다. 2. 기본적인 파이썬 문법을 이해하고 데이터 처리를 위한 자료구조를 이해하여 적용할 수 있다. 3. 문제 해결 방법을 위한 알고리즘을 이해하고 데이터 처리에 적용할 수 있다. 4. 파이썬 프로그램을 이용하여 실무적인 코딩 작업을 할 수 있다.			
	도서명	저자	출판사	비고
주교재	파이썬으로 배우는 누구나 코딩	강환수, 신용현	홍릉과학출판사	
수업시 사용도구	파이썬 기본 도구, 파이썬, 아나콘다와 주피터 노트북			
평가방법	중간고사 30%, 기말고사 40%, 과제물 및 퀴즈 10% 출석 20%(학교 규정, 학업성적 처리 지침에 따름)			
수강안내	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.			

1 주차	[개강일(3/16)]
학습주제	교과목 소개 및 강의 계획 1장 파이썬 언어의 개요와 첫 프로그래밍
목표및 내용	<ul style="list-style-type: none"> • 파이썬 언어란 무엇인지 이해하고 이 언어가 인기 있는 이유를 설명할 수 있다. • 파이썬 개발 도구를 설치해 프로그램을 구현할 수 있다. • 파이썬의 특징과 활용 분야를 설명할 수 있다.
미리읽어오기	교재 1장, 파이썬 개발환경 설치 파이썬 IDLE
과제, 시험, 기타	도전 프로그래밍
2 주차	[2주]
학습주제	2장 파이썬 프로그래밍을 위한 기초 다지기
목표및 내용	<ul style="list-style-type: none"> • 파이썬의 재료인 문자열과 수에 대해 이해하고 코드로 구현할 수 있다. • 변수를 이해하고 다양한 대입 연산자를 활용할 수 있다. • 표준 입력으로 문자열을 입력받은 후 원하는 자료로 변환해 활용할 수 있다. • 파이썬 IDLE을 활용할 수 있다.
미리읽어오기	교재 2장 리터럴과 변수의 이해 아나콘다의 주피터 노트북
과제, 시험, 기타	도전 프로그래밍
3 주차	[3주]
학습주제	3장 일상에서 활용되는 문자열과 논리 연산
목표및 내용	<ul style="list-style-type: none"> • 문자열에서 문자나 부분 문자열을 반환하는 여러 방법을 구현할 수 있다. • 문자열 객체에 소속된 다양한 메소드를 이해하고 활용할 수 있다. • 논리 값을 이해하고 다양한 연산을 사용해 실생활에서의 표현에 활용할 수 있다. • 아나콘다의 주피터 노트북을 활용할 수 있다.
미리읽어오기	교재 3장 문자열과 논리연산 파이참(pycharm)
과제, 시험, 기타	도전 프로그래밍
4 주차	[4주]
학습주제	4장 일상생활과 비유되는 조건과 반복
목표및 내용	<ul style="list-style-type: none"> • 조건에 따라 하나를 결정하는 if문을 구현할 수 있다. • 반복을 수행하는 while문과 for문을 구현할 수 있다. • 임의의 수인 난수를 이해하고 반복을 제어하는 break문과 continue문을 활용할 수 있다. • 파이참(pycharm)을 활용할 수 있다.
미리읽어오기	교재 4장 조건과 반복
과제, 시험, 기타	도전 프로그래밍

6 주차	[6주]
학습주제	6장 항목의 나열인 리스트와 튜플
목표 및 내용	<ul style="list-style-type: none"> • 다양한 종류의 항목을 쉽게 나열하는 리스트를 구현할 수 있다. • 리스트에서 부분 참조 방법, 이를 이용한 수정, 리스트 연결, 삽입과 삭제 그리고 리스트 컴프리헨션 등을 구현할 수 있다. • 수정할 수 없는 다양한 종류의 항목 나열을 쉽게 처리하는 튜플을 구현할 수 있다.
미리읽어오기	교재 6장 배열과 리스트
과제, 시험, 기타	도전 프로그래밍
6 주차	[6주]
학습주제	6장 키와 값의 나열인 딕셔너리와 증복을 불러오는 집합
목표 및 내용	<ul style="list-style-type: none"> • 키와 값의 쌍인 항목을 관리하는 딕셔너리를 생성하고 수정하는 방법을 이해하고, 다양한 방법으로 딕셔너리를 구현할 수 있다. • 집합의 특징을 이해하고, 합집합 등과 같은 다양한 집합의 연산을 구현할 수 있다. • 내장 함수 zip() 과 enumerate() , 시퀀스 간의 변환을 이해하고, 구현할 수 있다.
미리읽어오기	교재 6장 집합
과제, 시험, 기타	도전 프로그래밍
7 주차	[7주]
학습주제	7장 특정 기능을 수행하는 사용자 정의 함수와 내장 함수
목표 및 내용	<ul style="list-style-type: none"> • 함수의 내용과 필요성을 이해하고 함수를 직접 정의해 호출할 수 있다. • 인자의 기본 이해와 기본값 지정, 가변 인수와 키워드 인수를 활용할 수 있다. • 간편한 람다 함수와 표준 설치된 내장 함수를 사용할 수 있다.
미리읽어오기	교재 7장 함수의 정의와 호출
과제, 시험, 기타	도전 프로그래밍
8 주차	[중간고사]
학습주제	- 직무수행능력평가 1차(중간고사)
목표 및 내용	직무수행능력평가, 서술형 평가
미리읽어오기	교재 1장에서 7장까지
과제, 시험, 기타	
9 주차	[9주]
학습주제	8장 조건과 반복, 리스트와 튜플 기반의 미니 프로젝트 1
목표 및 내용	8개의 미니 프로젝트를 스스로 생각하고 프로그래밍해 코딩 능력뿐 아니라 문제 해결 능력을 키울 수 있다.
미리읽어오기	교재 8장
과제, 시험, 기타	

10 주차	[10주]
학습주제	9장 라이브러리 활용을 위한 모듈과 패키지
특요점 내용	<ul style="list-style-type: none"> 표준 모듈을 이해하고 사용자 정의 모듈도 직접 구현해 사용할 수 있다. 표준 모듈인 turtle을 사용해 기본적인 도형을 그릴 수 있다. 썬드파티 모듈 numpy와 matplotlib 등을 설치해 활용할 수 있다.
미리읽어오기	교재 9장
과제 시험 기타	도전 프로그래밍
11 주차	[11주]
학습주제	10장 그래픽 사용자 인터페이스 Tkinter와 Pygame
특요점 내용	<ul style="list-style-type: none"> GUI를 이해하고 GUI 표준 모듈인 Tkinter를 사용해 필요한 위젯을 구성하고 윈도우를 생성할 수 있다. 이벤트 처리를 이해하고 Tkinter에서 이벤트 처리를 구현할 수 있다. 썬드파티 GUI 모듈인 pygame을 설치해 기본적인 윈도우를 구현할 수 있다.
미리읽어오기	교재 10장
과제 시험 기타	도전 프로그래밍
12 주차	[12주]
학습주제	11장 실행 오류 및 파일을 다루는 예외 처리와 파일 입출력
특요점 내용	<ul style="list-style-type: none"> 예외 처리의 필요성을 이해하고 try except 구문을 사용해 예외를 처리할 수 있다. 프로그램에서 파일을 생성하는 필요성을 이해하고 필요한 파일을 만들 수 있다. 이미 생성된 파일에서 내용을 읽어 처리할 수 있다.
미리읽어오기	교재 11장
과제 시험 기타	도전 프로그래밍
13 주차	[13주]
학습주제	12장 일상생활의 사물 코딩인 객체지향 프로그래밍
특요점 내용	<ul style="list-style-type: none"> 객체와 클래스를 이해하고 필요한 클래스를 정의하고 객체를 만들어 활용할 수 있다. 클래스 속성과 인스턴스 속성, 정적 메소드와 클래스 메소드를 이해하고 정의할 수 있다. 상속을 이해하고 부모 클래스와 자식 클래스를 정의할 수 있다. 후상 메소드와 후상 클래스를 이해하고 정의할 수 있다.
미리읽어오기	교재 12장
과제 시험 기타	도전 프로그래밍
14 주차	[14주]
학습주제	13장 GUI 모듈과 객체지향 기반의 미니 프로젝트 II
특요점 내용	학습한 파이썬 문법 구조와 프로그래밍 기법을 활용해 8개의 미니 프로젝트를 스스로 생각하고 프로그래밍해 코딩 능력뿐 아니라 문제 해결 능력을 키울 수 있다.
미리읽어오기	교재 1장
과제 시험 기타	

16 주차	[기말고사]
학습주제	직무수행능력평가 2차(기말고사)
목표 및 내용	직무수행능력평가, <u>서술형평가</u>
미리읽어오기	8장에서 13장까지
과제, 시험, 기타	
수업지원 안내	장애학생을 위한 별도의 수강 지원을 받을 수 있습니다. 언어가 문제가 되는 학생은 글로 된 과제 안내, 확대문자 시험지 제공 등의 지원을 드립니다.

목차

1. 파이썬의 기본 이해

1-1. 파이썬이란?	1
1-2. 파이썬의 특징	2
1-3. 파이썬의 활용	4
1-4. 에디터의 사용	7

2. 파이썬의 자료형

2-1. 숫자형	11
2-2. 문자열 자료형	12

3. 파이썬의 제어문

3-1. if문	26
3-2. while문	28
3-3. for문	30

4. 문제풀이

4-1. 연습문제	32
4-2. 과제	36

5. 느낀점	42
--------	----

1-1 파이썬이란?

파이썬(Python)은 1990년 암스테르담의 귀도 반 로섬(Guido Van Rossum)이 개발한 인터프리터 언어이다. 귀도는 파이썬이라는 이름을 자신이 좋아하는 코미디 쇼인 "몬티 파이썬의 날아다니는 서커스 (Monty Python's Flying Circus)"에서 따왔다고 한다.

※ 인터프리터 언어란 한 줄씩 소스 코드를 해석해서 그때그때 실행해 결과를 바로 확인할 수 있는 언어이다.

파이썬의 사전적 의미는 고대 신화에 나오는 파르나소스 산의 동굴에 살던 큰 뱀을 뜻하며, 아폴로 신이 델파이에서 파이썬을 퇴치했다는 이야기가 전해지고 있다. 대부분의 파이썬 책 표지와 아이콘이 뱀 모양으로 그려져 있는 이유가 여기에 있다.



파이썬은 컴퓨터 프로그래밍 교육을 위해 많이 사용하지만, 기업의 실무를 위해서도 많이 사용하는 언어이다. 구글에서 만든 소프트웨어의 50% 이상이 파이썬으로 작성되었고 한다. 이외에도 많이 알려진 예를 몇 가지 들자면 온라인 사진 공유 서비스 인스타그램(Instagram), 파일 동기화 서비스 드롭박스(Dropbox)등이 있다.

또한 파이썬 프로그램은 공동 작업과 유지 보수가 매우 쉽고 편하다. 그 때문에 이미 다른 언어로 작성된 많은 프로그램과 모듈이 파이썬으로 재구성되고 있다. 국내에서도 그 가치를 인정받아 사용자 층이 더욱 넓어지고 있고, 파이썬을 사용해 프로그램을 개발하는 업체들 또한 늘어 가고 있는 추세이다.

1-2 파이썬의 특징

→ 파이썬은 인간다운 언어이다

프로그래밍이란 인간이 생각하는 것을 컴퓨터에 지시하는 행위라고 할 수 있다. 파이썬은 사람이 생각하는 방식을 그대로 표현할 수 있는 언어이다.

따라서 프로그래머는 굳이 컴퓨터의 사고 체계에 맞추어서 프로그래밍을 하려고 애쓸 필요가 없다.

→ 파이썬은 무료이지만 강력하다

오픈 소스인 파이썬은 당연히 무료이다. 사용료 걱정없이 언제 어디서든 파이썬을 다운로드하여 사용할 수 있다.

또한 프로그래머는 만들고자 하는 프로그램의 대부분을 파이썬으로 만들 수 있다. 물론 시스템 프로그래밍이나 하드웨어 제어와 같은 매우 복잡하고 반복 연산이 많은 프로그램은 파이썬과 어울리지 않는다.

하지만 파이썬은 이러한 약점을 극복할 수 있게끔 다른 언어로 만든 프로그램을 파이썬 프로그램에 포함시킬 수 있다.

→ 파이썬은 간결하다

```
# simple.py
languages = ['python', 'perl', 'c', 'java']

for lang in languages:
    if lang in ['python', 'perl']:
        print("%Gs need interpreter" % lang)
    elif lang in ['c', 'java']:
        print("%Gs need compiler" % lang)
    else:
        print("should not reach here")
```

이 예제는 프로그래밍 언어를 판별하여 그에 맞는 문장을 출력하는 파이썬 프로그램 예제이다.

위에 보이는 예제와 같이 파이썬 프로그램은 줄을 맞추지 않으면 실행되지 않는다. 코드를 예쁘게 작성하려고 줄을 맞추는 것이 아니라 프

로그래밍이 실행되게 하려면 꼭 줄을 맞추어야 하는 것이다. 이렇듯 줄을 맞추어 코드를 작성하는 행위는 가독성에 크게 도움이 된다.

※ 이렇게 코드의 줄을 맞추는 것을 "들여쓰기"라고 부른다. 파이썬에서 들여쓰기를 하지 않으면 프로그램이 실행되지 않는다.

1-3 파이썬의 활용

▶ 파이썬으로 할 수 있는 일

→ 시스템 유틸리티 제작

파이썬은 운영체제(윈도우, 리눅스 등)의 시스템 명령어를 사용할 수 있는 각종 도구를 갖추고 있기 때문에 이를 바탕으로 갖가지 시스템 유틸리티를 만드는 데 유리하다. 실제로 여러분은 시스템에서 사용 중인 서로 다른 유틸리티성 프로그램을 하나로 뭉쳐서 큰 힘을 발휘하게 하는 프로그램들을 무수히 만들어낼 수 있다.

※ 유틸리티란 컴퓨터 사용에 도움을 주는 여러 소프트웨어를 말한다.

→ GUI 프로그래밍

GUI(Graphic User Interface) 프로그래밍이란 쉽게 말해 화면에 또 다른 윈도우 창을 만들고 그 창에 프로그램을 동작시킬 수 있는 메뉴나 버튼, 그림 등을 추가하는 것이다. 파이썬은 GUI 프로그래밍을 위한 도구들이 잘 갖추어져 있어 GUI 프로그램을 만들기 쉽다. 대표적인 예로 파이썬 프로그램과 함께 설치되는 Tkinter(티케이인터)가 있다. Tkinter를 사용하면 단 5줄의 소스 코드만으로 윈도우 창을 띄울 수 있다.

→ C/C++와의 결합

파이썬은 접착(glue) 언어라고도 부르는데, 그 이유는 다른 언어와 잘 어울려 결합해서 사용할 수 있기 때문이다. C나 C++로 만든 프로그램을 파이썬에서 사용할 수 있으며, 파이썬으로 만든 프로그램 역시 C나 C++에서 사용할 수 있다.

→ 웹 프로그래밍

일반적으로 익스플로러나 크롬, 파이어폭스 같은 브라우저로 인터넷을 사용하는데, 누구나 한 번쯤 웹 서핑을 하면서 게시판이나 방명록에 글을 남겨 본 적이 있을 것이다. 그러한 게시판이나 방명록을 바로 웹 프로그램이라고 한다. 파이썬은 웹 프로그램을 만들기에 매우 적합한 도

구이며, 실제로 파이썬으로 제작한 웹 사이트는 셀 수 없을 정도로 많다.

→ 수치 연산 프로그래밍

사실 파이썬은 수치 연산 프로그래밍에 적합한 언어는 아니다. 수치가 복잡하고 연산이 많다면 C 같은 언어로 하는 것이 더 빠르기 때문이다. 하지만 파이썬은 NumPy라는 수치 연산 모듈을 제공한다. 이 모듈은 C로 작성했기 때문에 파이썬에서도 수치 연산을 빠르게 할 수 있다.

→ 데이터베이스 프로그래밍

파이썬은 사이베이스(Sybase), 인포믹스(Infomix), 오라클(Oracle), 마이에스큐엘(MySQL), 포스트그레스큐엘(PostgreSQL) 등의 데이터베이스에 접근하기 위한 도구를 제공한다.

또한 이런 굵직한 데이터베이스를 직접 사용하는 것 외에도 파이썬에는 재미있는 도구가 하나 더 있다. 바로 피클(pickle)이라는 모듈이다. 피클은 파이썬에서 사용하는 자료를 변형 없이 그대로 파일에 저장하고 불러오는 일을 맡아 한다. 이 책에서는 외장 함수에서 피클을 어떻게 사용하고 활용하는지에 대해서 알아본다.

→ 데이터 분석, 사물 인터넷

파이썬으로 만든 판다스(Pandas) 모듈을 사용하면 데이터 분석을 더 쉽고 효과적으로 할 수 있다. 데이터 분석을 할 때 아직까지는 데이터 분석에 특화된 'R'이라는 언어를 많이 사용하고 있지만, 판다스가 등장한 이후로 파이썬을 사용하는 경우가 점점 증가하고 있다.

사물 인터넷 분야에서도 파이썬은 활용도가 높다. 한 예로 라즈베리파이(Raspberry Pi)는 리눅스 기반의 아주 작은 컴퓨터이다. 라즈베리파이를 사용하면 홈시어터나 아주 작은 게임기 등 여러 가지 재미있는 것들을 만들 수 있는데, 파이썬은 이 라즈베리파이를 제어하는 도구로 사

용된다. 예를 들어 라즈베리파이에 연결된 모터를 작동시키거나 LED에 불이 들어오게 하는 일을 파이썬으로 할 수 있다.

▶ 파이썬으로 할 수 없는 일

→ 시스템과 밀접한 프로그래밍 영역

파이썬으로 리눅스 같은 운영체제, 엄청난 횟수의 반복과 연산이 필요한 프로그램 또는 데이터 압축 알고리즘 개발 프로그램 등을 만드는 것은 어렵다. 즉 대단히 빠른 속도를 요구하거나 하드웨어를 직접 건드려야 하는 프로그램에는 어울리지 않는다.

→ 모바일 프로그래밍

파이썬은 구글이 가장 많이 애용하는 언어이지만 파이썬으로 안드로이드 앱(App)을 개발하는 것은 아직 어렵다. 안드로이드에서 파이썬으로 만든 프로그램이 실행되도록 지원하긴 하지만 이것만으로 앱을 만들기에는 아직 역부족이다. 아이폰 앱을 개발하는 것 역시 파이썬으로 할 수 없다.

1-4 에디터의 사용

파이썬 대화형 인터프리터는 간단한 예제를 풀 때는 편리하지만 여러 줄의 복잡한 소스 코드를 가진 프로그램을 만들 때는 불편하다.

또한 인터프리터를 종료하자마자 프로그램이 사라지기 때문에 다시 사용하지 못한다는 단점이 있다. 그래서 여러 번 사용하기 위한 프로그램을 만들 때는 에디터를 사용한다.

에디터란 소스 코드를 편집할 수 있는 프로그래밍 툴을 말한다. 에디터에는 여러 가지 종류가 있다.

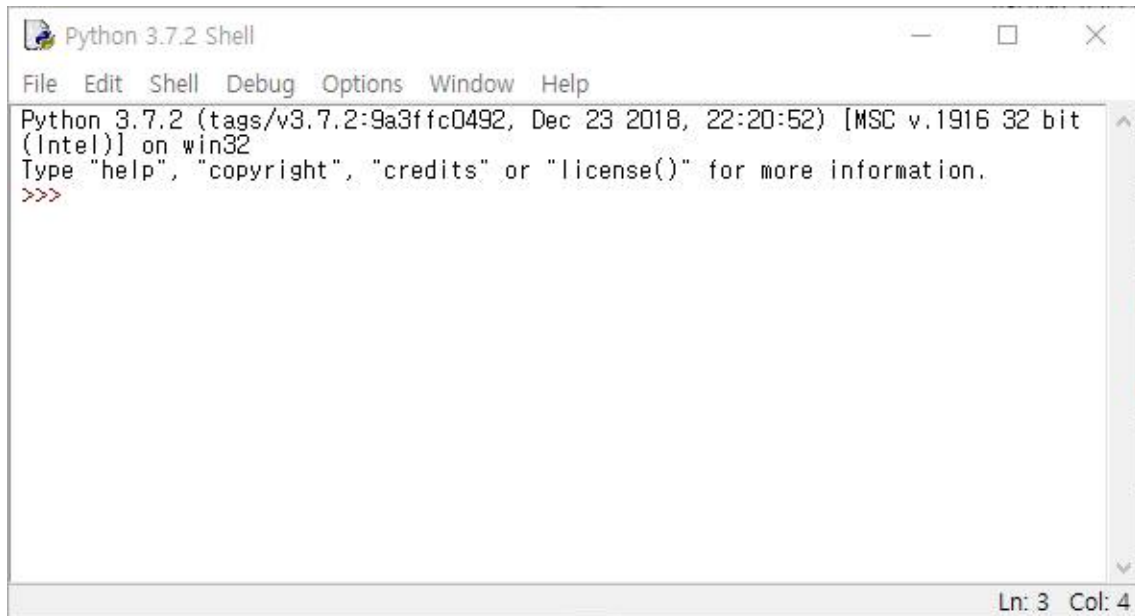
→ IDLE로 파이썬 프로그램 작성하기

파이썬 IDLE(Integrated Development and Learning Environment)은 파이썬 프로그램 작성을 도와주는 통합개발 환경이다.

[시작 → 모든 프로그램 → Python 3.7 → IDLE]을 선택해 파이썬 IDLE을 실행해 보자.



그러면 다음과 같은 IDLE 셸(Shell) 창이 먼저 나타난다.

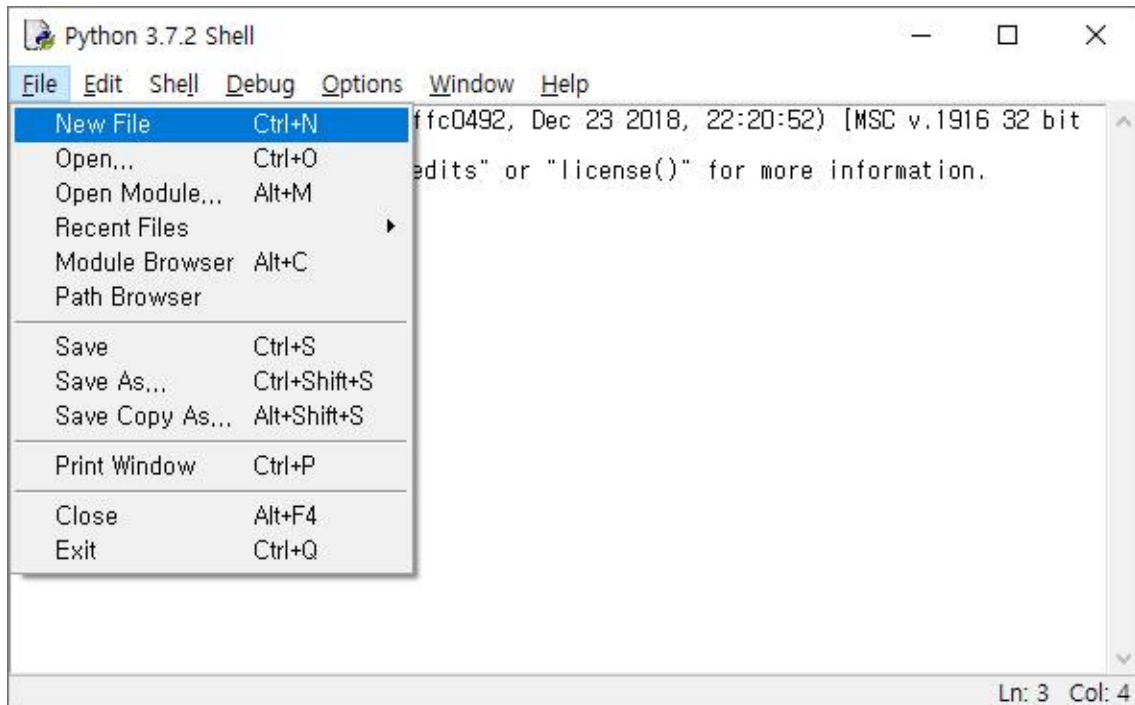


IDLE은 크게 두 가지 창으로 구성된다.

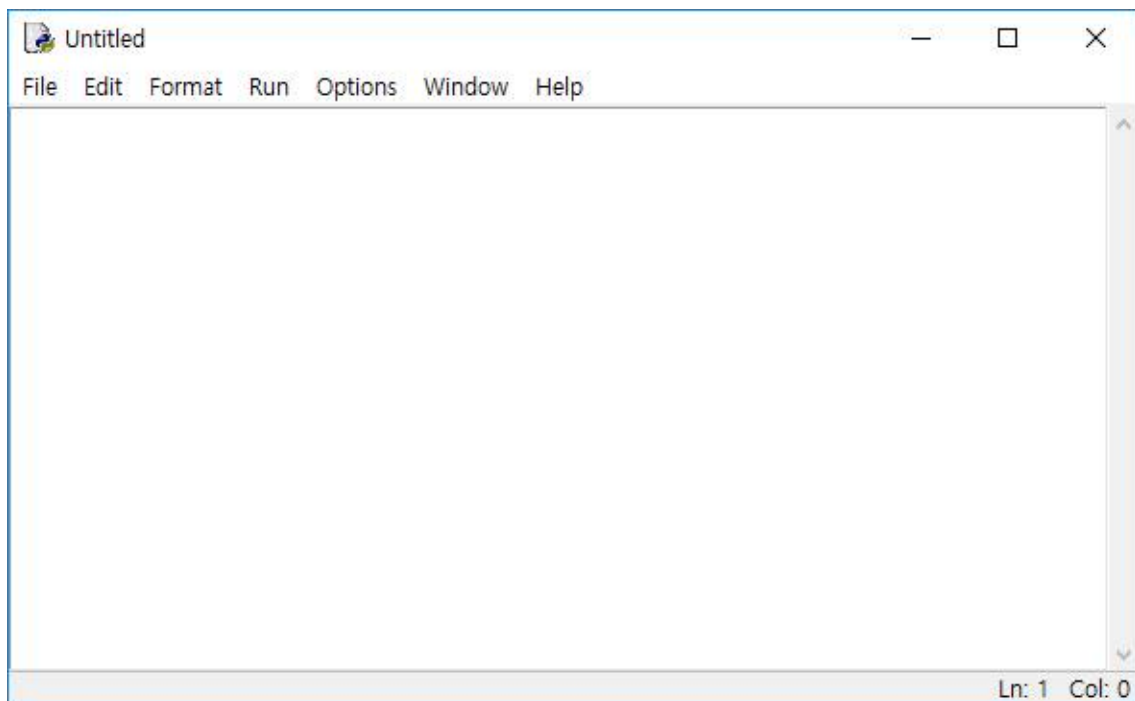
- IDLE 셸 창(Shell Window) - IDLE 에디터에서 실행한 프로그램의 결과가 표시되는 창으로서 파이썬 셸과 동일한 기능을 수행한다. IDLE을 실행하면 가장 먼저 나타나는 창이다.
- IDLE 에디터 창(Editor Window) - IDLE 에디터가 실행되는 창이다.

프로그램은 IDLE 에디터 창에서 작성한다. 이제 IDLE 에디터(Editor)를 실행해 보자.

IDLE 셸 창 메뉴에서 [File → New File]을 선택한다.



그러면 다음과 같이 빈 창이 나타나는데 이 창이 IDLE 에디터이다.



→ 비주얼 스튜디오 코드

비주얼 스튜디오 코드(Visual Studio Code)는 파이참과 더불어 프로그래머들에게 가장 많은 사랑을 받는 파이썬의 대표적인 에디터이다. 비주얼 스튜디오 코드는 공식 다운로드 사이트에서 내려받을 수 있다.

비주얼 스튜디오 코드는 파이썬 전용 에디터가 아니다(파이썬 외에 여러 가지 언어를 지원하는 에디터이다). 따라서 비주얼 스튜디오 코드를 설치한 후 파이썬 편집을 위해 가장 먼저 해야 할 일은 파이썬 Extension을 설치하는 것이다. 파이썬 Extension은 비주얼 스튜디오 코드를 실행한 후 Extension 메뉴를 사용하여 설치할 수 있다.

→ 파이참

파이참은 가장 유명한 파이썬 에디터 중 하나로서 코드를 작성할 때 자동 완성, 문법 체크 등 편리한 기능을 많이 제공한다. 이 에디터는 파이참 공식 다운로드 사이트에서 내려 받을 수 있다.

위와 같은 여러 가지 다양한 에디터들 중 내가 파이썬을 배우며 사용한 것은 바로 파이썬 IDLE이다.

2-1 숫자형

▶ 숫자형이란?

숫자형(Number)이란 숫자 형태로 이루어진 자료형으로, 우리가 이미 잘 알고 있는 것이다.

우리가 흔히 사용하는 것을 생각해 보자. 123 같은 정수, 12.34 같은 실수, 드물게 사용하긴 하지만 8진수나 16진수 같은 것도 있다.

다음 표는 파이썬에서 숫자를 어떻게 사용하는지 간략하게 보여 준다.

항목	사용 예
정수	123, -345, 0
실수	123.45, -1234.5, 3.4e10
8진수	0o34, 0o25
16진수	0x2A, 0xFF

▶ 숫자형을 활용하기 위한 연산자

- 사칙연산[+, -, *, /]
- x의 y제곱을 나타내는 ** 연산자
- 나눗셈 후 나머지를 반환하는 % 연산자
- 나눗셈 후 몫을 반환하는 // 연산자

2-2 문자열 자료형

▶ 문자열이란?

문자열(String)이란 문자, 단어 등으로 구성된 문자들의 집합을 의미한다. 예를 들어 다음과 같은 것들이 문자열이다.

▶ 문자열의 사용방법

→ 큰따옴표(")로 양쪽 둘러싸기

```
"Hello World"
```

→ 작은따옴표(')로 양쪽 둘러싸기

```
'Python is fun'
```

→ 큰따옴표 3개를 연속(""")으로 써서 양쪽 둘러싸기

```
"""Life is too short, You need python"""
```

→ 작은따옴표 3개를 연속('')으로 써서 양쪽 둘러싸기

```
'''Life is too short, You need python'''
```

▶ 여러 줄인 문자열을 변수에 대입하고 싶을 때

→ 줄을 바꾸기 위한 이스케이프 코드 `\n` 삽입하기

```
>>> multiline = "Life is too short\nYou need python"
```

위 예처럼 줄바꿈 문자 `\n`을 삽입하는 방법이 있지만 읽기에 불편하고 줄이 길어지는 단점이 있다.

→ 연속된 작은따옴표 3개('') 또는 큰따옴표 3개(''') 사용하기

위 1번의 단점을 극복하기 위해 파이썬에서는 다음과 같이 작은따옴표 3개('') 또는 큰따옴표 3개(''')를 사용한다.

```
>>> multiline="""
... Life is too short
... You need python
... """
```

▶ 이스케이프 코드란?

문자열 예제에서 여러 줄의 문장을 처리할 때 백슬래시 문자와 소문자 `n`을 조합한 `\n` 이스케이프 코드를 사용했다. 이스케이프 코드란 프로그래밍할 때 사용할 수 있도록 미리 정의해 둔 "문자 조합"이다. 주로 출력물을 보기 좋게 정렬하는 용도로 사용한다. 몇 가지 이스케이프 코드를 정리하면 다음과 같다.

코드	설명
<code>\n</code>	문자열 안에서 줄을 바꿀 때 사용

₩t	문자열 사이에 탭 간격을 줄 때 사용
₩₩	문자 ₩를 그대로 표현할 때 사용
₩'	작은따옴표(')를 그대로 표현할 때 사용
₩"	큰따옴표(")를 그대로 표현할 때 사용
₩r	캐리지 리턴(줄 바꿈 문자, 현재 커서를 가장 앞으로 이동)
₩f	폼 피드(줄 바꿈 문자, 현재 커서를 다음 줄로 이동)
₩a	벨 소리(출력할 때 PC 스피커에서 '뽕' 소리가 난다)
₩b	백 스페이스
₩000	널 문자

▶ 문자열 연산하기

파이썬에서는 문자열을 더하거나 곱할 수 있다. 다른 언어에서는 쉽게 찾아볼 수 없는 재미있는 기능으로, 우리 생각을 그대로 반영해 주는 파이썬만의 장점이라고 할 수 있다. 문자열을 더하거나 곱하는 방법에 대해 알아보자.

→ 문자열 더해서 연결하기(Concatenation)

```
>>> head = "Python"
>>> tail = " is fun!"
>>> head + tail
'Python is fun!'
```

위 소스 코드에서 세 번째 줄을 보자. 복잡하게 생각하지 말고 눈에 보이는 대로 생각해 보자. "Python"이라는 head 변수와 " is fun!"이라는 tail 변수를 더한 것이다. 결과는 'Python is fun!'이다. 즉 head와 tail 변수가 +에 의해 합쳐진 것이다.

→ 문자열 곱하기

```
>>> a = "python"
>>> a * 2
'pythonpython'
```

위 소스 코드에서 *의 의미는 우리가 일반적으로 사용하는 숫자 곱하기의 의미와는 다르다. 위 소스 코드에서 a * 2 문장은 a를 두 번 반복하라는 뜻이다. 즉 *는 문자열의 반복을 뜻하는 의미로 사용되었다. 굳이 코드의 의미를 설명할 필요가 없을 정도로 직관적이다.

→ 문자열 길이 구하기

```
>>> a = "Life is too short"
>>> len(a)
17
```

문자열의 길이는 다음과 같이 len 함수를 사용하면 구할 수 있다. len 함수는 print 함수처럼 파이썬의 기본 내장 함수로 별다른 설정 없이

바로 사용할 수 있다.

▶ 문자열 인덱싱과 슬라이싱

인덱싱(Indexing)이란 무엇인가를 "가리킨다"는 의미이고, 슬라이싱(Slicing)은 무엇인가를 "잘라낸다"는 의미이다.

문자열 인덱싱이란?

```
>>> a = "Life is too short, You need Python"
```

위 소스 코드에서 변수 a에 저장한 문자열의 각 문자마다 번호를 매겨 보면 다음과 같다.

```
Life is too short, You need Python
0      1      2      3
0123456789012345678901234567890123
```

"Life is too short, You need Python" 문자열에서 L은 첫 번째 자리를 뜻하는 숫자 0, 바로 다음인 i는 1 이런 식으로 계속 번호를 붙인 것이다. 중간에 있는 short의 s는 12가 된다.

이제 다음 예를 실행해 보자.

```
>>> a = "Life is too short, You need Python"
>>> a[3]
'e'
```

a[3]이 뜻하는 것은 a라는 문자열의 네 번째 문자 e를 말한다.
위 예에서 볼 수 있듯이 a[번호]는 문자열 안의 특정한 값을 뽑아내는

역할을 한다. 이러한 작업을 인덱싱이라고 한다.

문자열 슬라이싱이란?

"Life is too short, You need Python" 문자열에서 단순히 한 문자만을 뽑아내는 것이 아니라 'Life' 또는 'You' 같은 단어를 뽑아내려면 다음과 같이 하면 된다.

```
>>> a = "Life is too short, You need Python"
>>> b = a[0] + a[1] + a[2] + a[3]
>>> b
'Life'
```

위 방법처럼 단순하게 접근할 수도 있지만 파이썬에서는 더 좋은 방법을 제공한다. 바로 슬라이싱(Slicing) 기법이다.

```
>>> a = "Life is too short, You need Python"
>>> a[0:4]
'Life'
```

a[0:4]가 뜻하는 것은 a 문자열, 즉 "Life is too short, You need Python" 문장에서 자리 번호 0부터 4까지의 문자를 뽑아낸다는 뜻이다. 하지만 다음과 같은 의문이 생길 것이다.

a[0]은 L, a[1]은 i, a[2]는 f, a[3]은 e니까 a[0:3]으로도 Life라는 단어를 뽑아낼 수 있지 않을까? 다음 예로 확인해 보자.

```
>>> a[0:3]
'Lif'
```

이렇게 되는 이유는 간단하다. 슬라이싱 기법으로 a[시작 번호:끝 번호]

를 지정할 때 끝 번호에 해당하는 것은 포함하지 않기 때문이다. `a[0:3]`을 수식으로 나타내면 다음과 같다.

```
0 <= a < 3
```

이 수식을 만족하는 것은 `a[0]`, `a[1]`, `a[2]`이다. 따라서 `a[0:3]`은 'Lif'이고 `a[0:4]`는 'Life'가 되는 것이다.

→ 문자열을 슬라이싱하는 방법

```
>>> a[0:5]  
'Life '
```

위 예는 `a[0] + a[1] + a[2] + a[3] + a[4]`와 동일하다. `a[4]`는 공백 문자이기 때문에 'Life'가 아닌 'Life '가 출력된다. 공백 문자 역시 L, i, f, e 같은 문자와 동일하게 취급되는 것을 잊지 말자. 'Life'와 'Life '는 완전히 다른 문자열이다.

슬라이싱할 때 항상 시작 번호가 0일 필요는 없다.

```
>>> a[0:2]  
'Li'  
>>> a[5:7]  
'is'  
>>> a[12:17]  
'short'
```

`a[시작 번호:끝 번호]`에서 끝 번호 부분을 생략하면 시작 번호부터 그 문자열의 끝까지 뽑아낸다.

```
>>> a[19:]  
'You need Python'
```

a[시작 번호:끝 번호]에서 시작 번호를 생략하면 문자열의 처음부터 끝 번호까지 뽑아낸다.

```
>>> a[:17]  
'Life is too short'
```

a[시작 번호:끝 번호]에서 시작 번호와 끝 번호를 생략하면 문자열의 처음부터 끝까지를 뽑아낸다.

```
>>> a[:]  
'Life is too short, You need Python'
```

슬라이싱에서도 인덱싱과 마찬가지로 마이너스(-) 기호를 사용할 수 있다.

```
>>> a[19:-7]  
'You need'
```

위 소스 코드에서 a[19:-7]이 뜻하는 것은 a[19]에서부터 a[-8]까지를 말한다. 이 역시 a[-7]은 포함하지 않는다.

→ 슬라이싱으로 문자열 나누기

다음은 자주 사용하게 되는 슬라이싱 기법 중 하나이다.

```
>>> a = "20190703Byeon"
>>> num = a[:8]
>>> name = a[8:]
>>> num
'20190703'
>>> name
'Byeon'
```

위 예는 문자열 `a`를 두 부분으로 나누는 기법이다. 숫자 8을 기준으로 문자열 `a`를 양쪽으로 한 번씩 슬라이싱했다.

`a[:8]`은 `a[8]`이 포함되지 않고, `a[8:]`은 `a[8]`을 포함하기 때문에 8을 기준으로 해서 두 부분으로 나눌 수 있는 것이다.

위 예에서는 "20190703Byeon" 문자열을 나의 학번을 나타내는 부분인 '20190703'과 이름의 성을 나타내는 부분인 'Byeon'으로 나누는 방법을 보여 준다.

["Pithon"이라는 문자열을 "Python"으로 바꾸려면?]

Pithon 문자열을 Python으로 바꾸려면 어떻게 해야 할까? 제일 먼저 떠오르는 생각은 다음과 같다.

```
>>> a = "Pithon"
>>> a[1]
'i'
>>> a[1] = 'y'
```

즉 a 변수에 "Pithon" 문자열을 대입하고 a[1]의 값이 i니까 a[1]을 y로 바꾸어 준다는 생각이다. 하지만 결과는 어떻게 나올까?

당연히 오류가 발생한다.

문자열 자료형은 그 요솟값을 변경할 수 없다. 그래서 immutable한 자료형이라고도 부른다.

하지만 앞에서 살펴본 슬라이싱 기법을 사용하면 Pithon 문자열을 사용해 Python 문자열을 만들 수 있다.

```
>>> a = "Pithon"
>>> a[:1]
'P'
>>> a[2:]
'thon'
>>> a[:1] + 'y' + a[2:]
'Python'
```

위 예에서 볼 수 있듯이 슬라이싱을 사용하면 "Pithon" 문자열을 'P' 부분과 'thon' 부분으로 나눌 수 있기 때문에 그 사이에 'y' 문자를 추가하여 'Python'이라는 새로운 문자열을 만들 수 있다.

▶ 문자열 포맷 코드

코드	설명
%s	문자열(String)
%c	문자 1개(character)
%d	정수(Integer)

%f	부동소수(floating-point)
%o	8진수
%x	16진수
%%	Literal % (문자 % 자체)

▶ 문자열 관련 함수들

→ 문자 개수 세기(count)

```
>>> a = "hobby"
>>> a.count('b')
2
```

→ 위치 알려주기1(find)

```
>>> a = "Python is the best choice"
>>> a.find('b')
14
>>> a.find('k')
-1
```

문자열 중 문자 b가 처음으로 나온 위치를 반환한다. 만약 찾는 문자나 문자열이 존재하지 않는다면 -1을 반환한다.

→ 위치 알려주기2(index)

```
>>> a = "Life is too short"
>>> a.index('t')
8
>>> a.index('k')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: substring not found
```

문자열 중 문자 t가 맨 처음으로 나온 위치를 반환한다. 만약 찾는 문자나 문자열이 존재하지 않는다면 오류를 발생시킨다. 앞의 find 함수와 다른 점은 문자열 안에 존재하지 않는 문자를 찾으면 오류가 발생한다는 점이다.

→ 문자열 삽입(join)

```
>>> ",".join('abcd')
'a,b,c,d'
```

abcd 문자열의 각각의 문자 사이에 ','를 삽입한다.

→ 소문자를 대문자로 바꾸기(upper)

```
>>> a = "hi"
>>> a.upper()
'HI'
```

upper 함수는 소문자를 대문자로 바꾸어 준다. 만약 문자열이 이미 대문자라면 아무 변화도 일어나지 않을 것이다.

→ 대문자를 소문자로 바꾸기(lower)

```
>>> a = "HI"  
>>> a.lower()  
'hi'
```

lower 함수는 대문자를 소문자로 바꾸어 준다.

→ 왼쪽 공백 지우기(lstrip)

```
>>> a = " hi "  
>>> a.lstrip()  
'hi '
```

문자열 중 가장 왼쪽에 있는 한 칸 이상의 연속된 공백들을 모두 지운다. lstrip에서 l은 left를 의미한다.

→ 오른쪽 공백 지우기(rstrip)

```
>>> a= " hi "  
>>> a.rstrip()  
' hi'
```

문자열 중 가장 오른쪽에 있는 한 칸 이상의 연속된 공백을 모두 지운다. rstrip에서 r는 right를 의미한다.

→ 양쪽 공백 지우기(strip)

```
>>> a = " hi "  
>>> a.strip()  
'hi'
```

문자열 양쪽에 있는 한 칸 이상의 연속된 공백을 모두 지운다.

→ 문자열 바꾸기(replace)

```
>>> a = "Life is too short"
>>> a.replace("Life", "Your leg")
'Your leg is too short'
```

replace(바뀌게 될 문자열, 바꿀 문자열)처럼 사용해서 문자열 안의 특정한 값을 다른 값으로 치환해 준다.

→ 문자열 나누기(split)

```
>>> a = "Life is too short"
>>> a.split()
['Life', 'is', 'too', 'short']
>>> b = "a:b:c:d"
>>> b.split(':')
['a', 'b', 'c', 'd']
```

3-1 if문

▶ if문의 기본 구조

다음은 if와 else를 사용한 조건문의 기본 구조이다.

if 조건문:

수행할 문장1

수행할 문장2

...

else:

수행할 문장A

수행할 문장B

...

조건문을 테스트해서 참이면 if문 바로 다음 문장(if 블록)들을 수행하고, 조건문이 거짓이면 else문 다음 문장(else 블록)들을 수행하게 된다. 그러므로 else문은 if문 없이 독립적으로 사용할 수 없다. 조건문을 작성 때에도 들여쓰기는 꼭 주의해야 한다.

▶ 조건문이란 무엇인가?

if 조건문에서 "조건문"이란 참과 거짓을 판단하는 문장을 말한다.

→ 비교연산자

비교연산자	설명
$x < y$	x가 y보다 작다
$x > y$	x가 y보다 크다

<code>x == y</code>	x와 y가 같다
<code>x != y</code>	x와 y가 같지 않다
<code>x >= y</code>	x가 y보다 크거나 같다
<code>x <= y</code>	x가 y보다 작거나 같다

→ **and, or, not**

조건을 판단하기 위해 사용하는 다른 연산자로는 and, or, not이 있다. 각각의 연산자는 다음처럼 동작한다.

연산자	설명
<code>x or y</code>	x와 y 둘중에 하나만 참이어도 참이다
<code>x and y</code>	x와 y 모두 참이어야 참이다
<code>not x</code>	x가 거짓이면 참이다

→ **x in s, x not in s**

in	not in
<code>x in 리스트</code>	<code>x not in 리스트</code>
<code>x in 튜플</code>	<code>x not in 튜플</code>
<code>x in 문자열</code>	<code>x not in 문자열</code>

3-2 while문

▶ while문의 기본 구조

반복해서 문장을 수행해야 할 경우 while문을 사용한다. 그래서 while문을 반복문이라고도 부른다.

```
while <조건문>:  
    <수행할 문장1>  
    <수행할 문장2>  
    <수행할 문장3>  
    ...
```

▶ while문 강제로 빠져나가기

```
print("카운트다운 시작합니다.")  
while i>=10 :  
    if i==5 :  
        break  
    print(i)  
    i-=1
```

▶ while문의 맨 처음으로 돌아가기

while문 안의 문장을 수행할 때 입력 조건을 검사해서 조건에 맞지 않으면 while문을 빠져나간다. 그런데 프로그래밍을 하다 보면 while문을 빠져나가지 않고 while문의 맨 처음(조건문)으로 다시 돌아가게 만들고 싶은 경우가 생기게 된다. 이때 사용하는 것이 바로 continue문이다.

```
>>> a = 0
>>> while a < 10:
    a = a + 1
    if a % 2 == 0: continue
    print(a)
```

▶ 무한 루프

무한 루프란 무한히 반복한다는 의미이다. 우리가 사용하는 일반 프로그램 중에서 무한 루프 개념을 사용하지 않는 프로그램은 거의 없다. 그만큼 자주 사용한다는 뜻이다.

다음은 무한 루프의 기본 형태이다.

```
while True:
    수행할 문장1
    수행할 문장2
    ...
```

3-3 for문

▶ for문의 기본 구조

for문의 기본 구조는 다음과 같다.

```
for 변수 in 리스트(또는 튜플, 문자열):  
    수행할 문장1  
    수행할 문장2  
    ...
```

▶ for문과 continue

while문에서 살펴본 continue문을 for문에서도 사용할 수 있다. 즉 for문 안의 문장을 수행하는 도중에 continue문을 만나면 for문의 처음으로 돌아가게 된다.

```
marks = [90, 25, 67, 45, 80]  
  
number = 0  
for mark in marks:  
    number = number + 1  
    if mark < 60:  
        continue  
    print("%d번 학생 축하합니다. 합격입니다. " % number)
```

점수가 60점 이하인 학생일 경우에는 `mark < 60`이 참이 되어 `continue`문이 수행된다. 따라서 축하 메시지를 출력하는 부분인 `print`문을 수행하지 않고 for문의 처음으로 돌아가게 된다.

▶ for문과 range 함수

```
>>> add = 0
>>> for i in range(1, 11):
    add = add + i

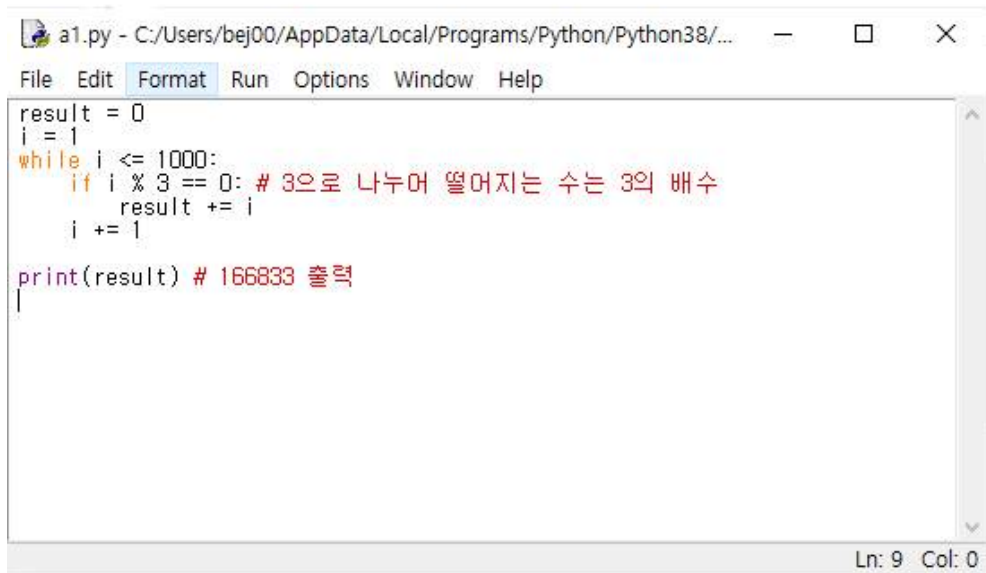
>>> print(add)
55
```

range(1, 11)은 숫자 1부터 10까지(1 이상 11 미만)의 숫자를 데이터로 갖는 객체이다. 따라서 위 예에서 i 변수에 리스트의 숫자가 1부터 10까지 하나씩 차례로 대입되면서 add = add + i 문장을 반복적으로 수행하고 add는 최종적으로 55가 된다.

연습문제

Q1. while문을 사용해 1부터 1000까지의 자연수 중 3의 배수의 합을 구해 보자.

코드)

A screenshot of a Python IDE window titled 'a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Python38/...'. The code inside is:

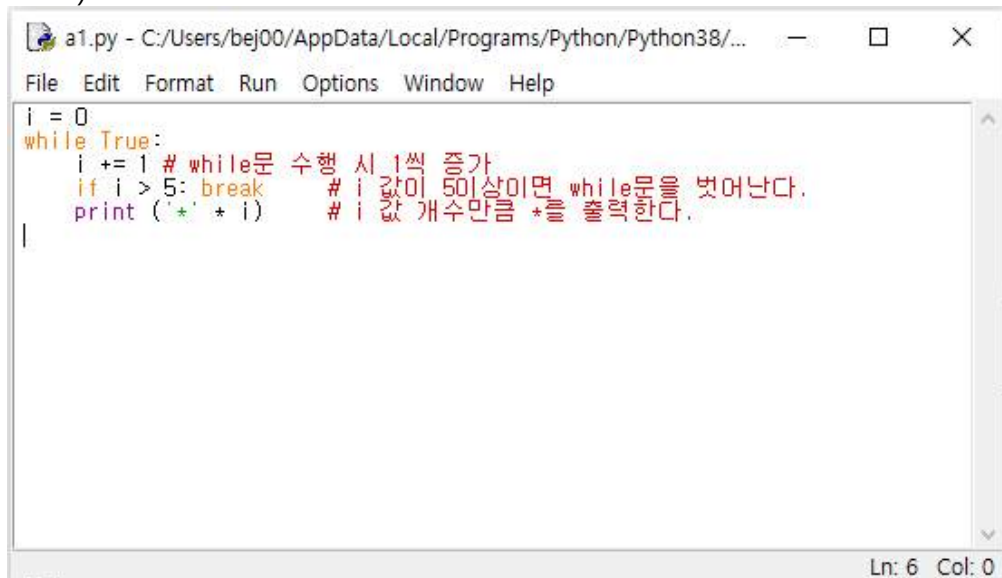
```
result = 0
i = 1
while i <= 1000:
    if i % 3 == 0: # 3으로 나누어 떨어지는 수는 3의 배수
        result += i
    i += 1

print(result) # 166833 출력
```

The status bar at the bottom right shows 'Ln: 9 Col: 0'.

Q2. while문을 사용하여 다음과 같이 별(*)을 표시하는 프로그램을 작성해 보자.

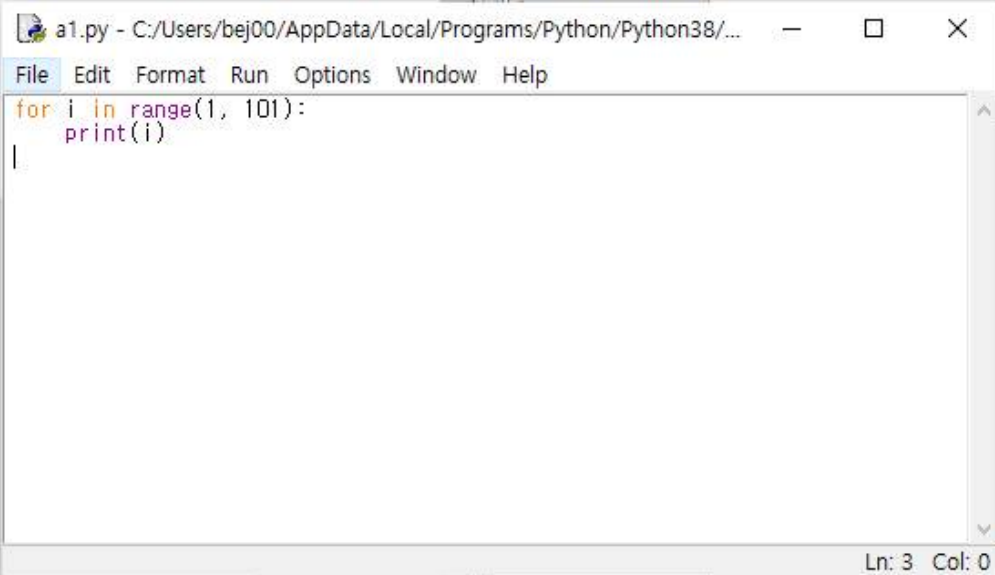
코드)

A screenshot of a Python IDE window titled 'a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Python38/...'. The code inside is:

```
i = 0
while True:
    i += 1 # while문 수행 시 1씩 증가
    if i > 5: break # i 값이 5이상이면 while문을 벗어난다.
    print('*', * i) # i 값 개수만큼 *를 출력한다.
```

The status bar at the bottom right shows 'Ln: 6 Col: 0'.

Q3. for문을 사용해 1부터 100까지의 숫자를 출력해 보자.
코드)

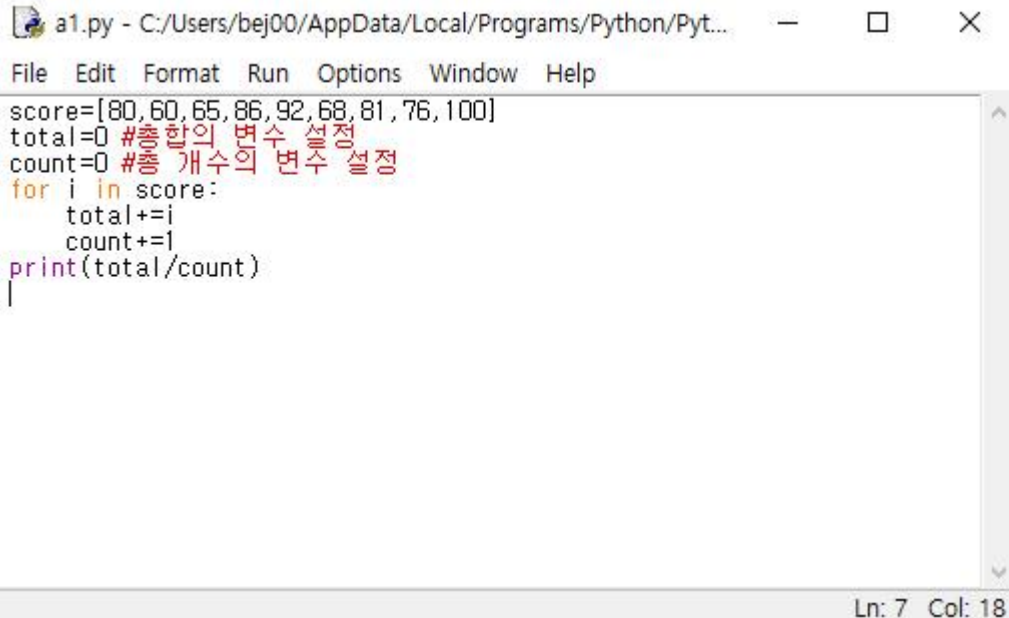


```
a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Python38/...  
File Edit Format Run Options Window Help  
for i in range(1, 101):  
    print(i)  
|  
  
Ln: 3 Col: 0
```

Q4. A 학급에 총 10명의 학생이 있다. 이 학생들의 중간고사 점수는 다음과 같다.

[80, 60, 65, 86, 92, 68, 81, 76, 100]

이때 10명의 학생들 평균 점수를 구해보자.



```
a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Pyt...  
File Edit Format Run Options Window Help  
score=[80,60,65,86,92,68,81,76,100]  
total=0 #총합의 변수 설정  
count=0 #총 개수의 변수 설정  
for i in score:  
    total+=i  
    count+=1  
print(total/count)  
|  
  
Ln: 7 Col: 18
```

Q5. 숫자를 입력받고 짝수인지 홀수 인지 맞춰보자.
코드)

```
*a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Pyt...
File Edit Format Run Options Window Help
count = 0
while True:
    num = int(input("숫자 입력: "))
    if num % 2 == 0:
        msg = "짝수입니다."
        count += 1
    else:
        msg = "홀수입니다."
        count += 1
    print(msg)
    if count >= 3:
        print("종료되었습니다.")
        break
```

Ln: 14 Col: 57

Q6. 초를 입력받고 분과 초를 출력해보자.
코드)

```
a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Pyt...
File Edit Format Run Options Window Help
time = int(input("초를 입력해주세요"))
if time >= 3600:
    hour = time // 3600
    min = (time % 3600) // 60
    sec = (time % 3600) % 60
    print(hour, "시간", min, "분", sec, "초")
elif time < 3600:
    min = time // 60
    sec = time % 60
    print(min, "분", sec, "초")
elif time < 60:
    sec = time
    print(sec, "초")
|
```

Ln: 16 Col: 0

Q7. 화폐매수를 단위에 맞춰 구해보자.

코드)

```
a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Pyt...
File Edit Format Run Options Window Help
money = int(input("금액을 입력하세요"))

오만원 = money // 50000
만원 = (money % 50000) // 10000
오천원 = ((money % 50000) % 10000) // 5000
천원 = (((money % 50000) % 10000) % 5000) // 1000
오백원 = (((((money % 50000) % 10000) % 5000) % 1000) // 500)
백원 = ((((((money % 50000) % 10000) % 5000) % 500) // 100)

print("오만원", 오만원, "매", "\n",
      "만원", 만원, "매", "\n",
      "오천원", 오천원, "매", "\n",
      "천원", 천원, "매", "\n",
      "오백원", 오백원, "매", "\n",
      "백원", 백원, "매", "\n",
      "\n")

Ln: 15 Col: 0
```

Q8. 국어와 수학과 영어이 점수를 입력받고 총점과 평균을 출력해보자.

코드)

```
*a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Pytho...
File Edit Format Run Options Window Help
while True:
    korea = int(input("국어 점수를 입력해주세요"))
    math = int(input("수학 점수를 입력해주세요"))
    english = int(input("영어 점수를 입력해주세요"))
    total = korea + math + english
    avr = total / 3

    print("국어점수 :", korea, "수학점수 :", math,
          "영어점수 :", english, "\n총점 :", total,
          "평균 :", avr)

    Q = int(input("계속 입력하시겠습니까? 예:1 / 아니오 : 0"))
    if Q == 1:
        continue # 다시 while문의 처음으로
    else:
        print("종료하겠습니다.")
        break

Ln: 18 Col: 0
```


2주차 과제

1.

```
==== RESTART: C:/Users/bej00/
문자열 1: python
문자열 2: 언어
python 언어
>>>
```


 a1.py - C:/Users/bej00/AppData/L

File Edit Format Run Options V

```
num1=input('문자열 1: ')
num2=input('문자열 2: ')
print(num1,num2)
```

2.

```
==== RESTART: C:/Users/bej00/AppData/Local/Progra
아메리카노 몇개 주문하세요? 10
총 가격은 35000 입니다
>>>
```

 *a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Pytho

File Edit Format Run Options Window Help

```
num = eval(input("아메리카노 몇개 주문하세요? "))
total=num*3500
print("총 가격은 ",total,"입니다")
```

3.

```
==== RESTART: C:/Users/bej00/AppData/Local/Programs/Python/Pyt
온도 입력 >>> 38
정확 계산: 섭씨: 38 , 화씨: 100.4
정확 계산: 섭씨: 38 , 화씨: 106
차이: -5.5999999999999994
>>>
```

 a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Python38/a1.py (3.8.2)

File Edit Format Run Options Window Help

```
celsius = eval(input("온도 입력 >>> "))
fahrenheit1=celsius*9/5+32
fahrenheit2=celsius*2+30
error=fahrenheit1-fahrenheit2
print("정확 계산: 섭씨: ",celsius, ",","화씨: ", fahrenheit1)
print("정확 계산: 섭씨: ",celsius, ",","화씨: ", fahrenheit2)
print("차이: ",error)
```

4.

```
==== RESTART: C:/Users/bej00/AppData/Local/Programs/Pyth
입력할 실수의 진수 base는? 16
16첫 번째 16진수 실수 입력 >> a
16두 번째 16진수 실수 입력 >> b
실수1: 10.0 , 실수2: 11.0
합: 21
차: -1
곱하기: 110
나누기: 0.9090909090909091
>>>
```

 a1.py - C:/Users/bej00/AppData/Local/Programs/Python/Python38/a1.p

File Edit Format Run Options Window Help

```
base=int(input('입력할 실수의 진수 base는? '))
a=input(str(base)+'첫 번째 16진수 실수 입력 >> ')
b=input(str(base)+'두 번째 16진수 실수 입력 >> ')
data=int(a,base);
data2=int(b,base);
print('실수1:', float(data),',','실수2:', float(data2))
print("합: ",data+data2)
print("차: ",data-data2)
print("곱하기: ",data*data2)
print("나누기: ",data/data2)
```

3주차 과제

1.

```
문자열 입력 >> Python is a good language!
참조할 첨자: 0~ 25
참조할 첨자 입력 >> 12
문자열: Python is a good language! 길이: 26
참조 문자: g
>>>
```

a1.py - C:\Users\bej00\AppData\Local\Programs\Pytl

File Edit Format Run Options Window Help

```
str=input('문자열 입력 >> ')
n=len(str)
print('참조할 첨자: 0~',n-1)
a=int(input('참조할 첨자 입력 >> '))
print('문자열: ',str,'길이: ',n)
print('참조 문자: ',str[a])
```

2.

```
위 철학 메소드 replace()를 사용해 다음 철학으로 다시 저장
Explicit is better than implicit.
>>>
```

a1.py - C:\Users\Wbej00\AppData\Local\Programs\Python\Python38\wa1.py...

File Edit Format Run Options Window Help

```
str='Beautiful is better than ugly.'  
print('위 철학 메소드 replace()를 사용해 다음 철학으로 다시 저장')  
a=str.replace(str,'Explicit is better than implicit.')  
print(a)
```

3.

```
시각 정보 입력 >> 17:25:46
입력 시각 정보: 17 : 25 : 46
17시 25분 46초
>>>
```

a1.py - C:\Users\bej00\AppData\Local\Programs\Python\Python3

File Edit Format Run Options Window Help

```
hours,mins,secs=input('시각 정보 입력 >> ').split(':')
print('입력 시각 정보: ',hours,':',mins,':',secs)
print(hours+'시',mins+'분',secs+'초')
```

4.

```
10진 수: -7, 2진 수: -0000111, 8진 수: -0o7, 16진 수: -0x7
10진 수: -6, 2진 수: -0000110, 8진 수: -0o6, 16진 수: -0x6
10진 수: -5, 2진 수: -0000101, 8진 수: -0o5, 16진 수: -0x5
10진 수: -4, 2진 수: -0000100, 8진 수: -0o4, 16진 수: -0x4
10진 수: -3, 2진 수: -0000011, 8진 수: -0o3, 16진 수: -0x3
10진 수: -2, 2진 수: -0000010, 8진 수: -0o2, 16진 수: -0x2
10진 수: -1, 2진 수: -0000001, 8진 수: -0o1, 16진 수: -0x1
10진 수: 0, 2진 수: 00000000, 8진 수: 0o0, 16진 수: 0x0
```

a1.py - C:\Users\bej00\AppData\Local\Programs\Python\Python38\python.exe a1.py (3.8.2)

File Edit Format Run Options Window Help

```
a=-7
while a<1:
    print('10진 수: {0:3d}, 2진 수: {0:08b}, 8진 수: {0:#o}, 16진 수: {0:#x}'.format(a))
    a=a+1
```

4주차 과제

1.

```
==== RESTART: C:\Users\bej00\AppData\Local\Temp\
```

```
1
```

```
겨울
```

```
*a1.py - C:\Users\bej00\AppData\Local\Programs\Python\Python38-64\python.exe
```

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

```
a=int(input())
if a==11 or a==12 or a==1 or a==2 or a==3 :
    print("겨울")
elif a==4 or a==5 :
    print("봄")
elif a==6 or a==7 or a==8 :
    print("여름")
elif a==9 or a==10 :
    print("가을")
```

2.

```
==== RESTART: C:\Users\bej00\AppData\Local\Temp\
[36, 62, 2] 중에서 최대: 62
```

```
*a1.py - C:\Users\bej00\AppData\Local\Programs\Python\Python38-64\python.exe
```

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

```
import random
count = 3
x = random.sample(range(0, 100 + 1), count)
print(x, '중에서 최대: ', max(x))
```


3.

```
===== RESTART: C:\Users\bej00\AppData\Local\Programs\Python\Python38\python.exe a1.py =====
섭씨: 20, 화씨: 68.0, 화씨(약식): 70, 차이: 2.00
섭씨: 23, 화씨: 73.4, 화씨(약식): 76, 차이: 2.60
섭씨: 26, 화씨: 78.8, 화씨(약식): 82, 차이: 3.20
섭씨: 29, 화씨: 84.2, 화씨(약식): 88, 차이: 3.80
섭씨: 32, 화씨: 89.6, 화씨(약식): 94, 차이: 4.40
섭씨: 35, 화씨: 95.0, 화씨(약식): 100, 차이: 5.00
섭씨: 38, 화씨: 100.4, 화씨(약식): 106, 차이: 5.60
섭씨: 41, 화씨: 105.8, 화씨(약식): 112, 차이: 6.20

*a1.py - C:\Users\bej00\AppData\Local\Programs\Python\Python38\python.exe a1.py (3.8.2)*
File Edit Format Run Options Window Help
celsius=20
while celsius<=41:
    fahrenheit1=celsius*9/5+32
    fahrenheit2=celsius*2+30
    error=fahrenheit1-fahrenheit2
    error=format(abs(error),'.2f')
    print("섭씨: ",celsius," ", "화씨: ", fahrenheit1," ", "화씨(약식): ", fahrenheit2," ", "차이: ", error)
    celsius+=3
```

4.

```
===== RESTART: C:\Users\bej00\AppData\Local\Programs\Python\Python38\python.exe a1.py =====
소수(prime number)인지를 판별한 2 이상의 정수 입력 >> 25
정수 25 는 소수가 아니다.
```

```
a1.py - C:\Users\bej00\AppData\Local\Programs\Python\Python38\python.exe a1.py (3.8.2)
File Edit Format Run Options Window Help
a=int(input("소수(prime number)인지를 판별한 2 이상의 정수 입력 >> "))
b=0
for i in range(2,a):
    if a%i==0:
        b=1
if b==0:
    print("정수 ",a,"는 소수이다.")
else:
    print("정수 ",a,"는 소수가 아니다.")
|
```

느낀점

교과목 포트폴리오 작성이 처음이라 이것을 작성하는 의의가 무엇인지 제대로 인지하지 못했었다. 그래서인지 어떻게 만들어야 하는 것인지 감도 잡히지 않았고 포트폴리오의 양식이라는 것이 세부적으로 나와있지 않았기에 방향을 잡는 데에 더 많은 시간이 들었던 것 같다.

하지만 추후에 포트폴리오라는 것이 나만의 기록 중 하나라는 사실을 다시 한번 되새겼고 나 자신이 이 교과목에 대해 공부해 보며 부족한 부분을 요약정리하는 것이 가장 중요한 부분이라는 것을 깨달았다.

이후 작성하는 것이 매우 수월해졌고 포트폴리오 작성이 교과목 학습에 있어 엄청난 도움이 된다는 사실이 놀라웠다.

머리에 있던 지식들을 재정리해보는 유익한 시간이었고, 시간이 된다면 다른 교과목들의 포트폴리오도 작성해보고 싶다는 생각이 들었다.