

Python 교과목 포트 폴리오

학과: 컴퓨터정보공학과
학번: 20173854
이름: 한상휘

목차

1. 강의계획서(p3 ~ p7)
2. Python이란?(p8 ~ p9)
3. Python 예제풀이
4. 느낀 점, 마무리

2020 학년도 1학기	전공	컴퓨터정보공학과	학부	컴퓨터공학부
과 목 명	파이썬프로그래밍 (2019009-PD)			
강의실 과 강의시간	수:8(3-217), 7(3-217), 8(3-217)		학점	3
교과분류	이론/실습		시수	3
담당 교수	강환수 + 연구실 : 2호관-708 + 전 화 : 02-2610-1941 + E-MAIL : hskang@dongyang.ac.kr + 면담 가능기간 : 화요일 13-16			
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.			
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 적용 할 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, 파이참, 아나콘다와 주피터 노트북			
평가방법	중간고사 30%, 기말고사 40%, 과제물 및 퀴즈 10% 출석 20%(학교 규정, 학업성적 처리 지침에 따름)			
수강안내	1. 파이썬의 개발 환경을 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(9/16)】
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 난수들 이해하고 반복을 제어하는 break문과 continue문을 활용할 수 있다. • 파이참(pycharm)을 활용할 수 있다.
미리읽어오기	교재 4장 조건과 반복
과제,시험,기타	도전 프로그래밍

10 주차	[10주]
학습주제	9장 라이브러리 활용을 위한 모듈과 패키지
목표및 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. 써드파티 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
목표및 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. 써드파티 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
목표및 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
목표및 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. 추상 메소드와 추상 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
목표및 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 13장
과제,시험,기타	

5 주차	[6주]
학습주제	6장 항목의 나열인 리스트와 튜플
목표및 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 중목을 불허하는 집합
목표및 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
목표및 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)
목표및 내용	직무수행능력평가, 서술형 평가
미리읽어오기	교재 1장에서 7장 까지
과제,시험,기타	
9 주차	[9주]
학습주제	8장 조건과 반복, 리스트와 튜플 기반의 미니 프로젝트 I
목표및 내용	8개의 미니 프로젝트를 스스로 생각하고 프로그래밍해 코딩 능력뿐 아니라 문제 해결 능력을 키울 수 있다.
미리읽어오기	교재 8장
과제,시험,기타	

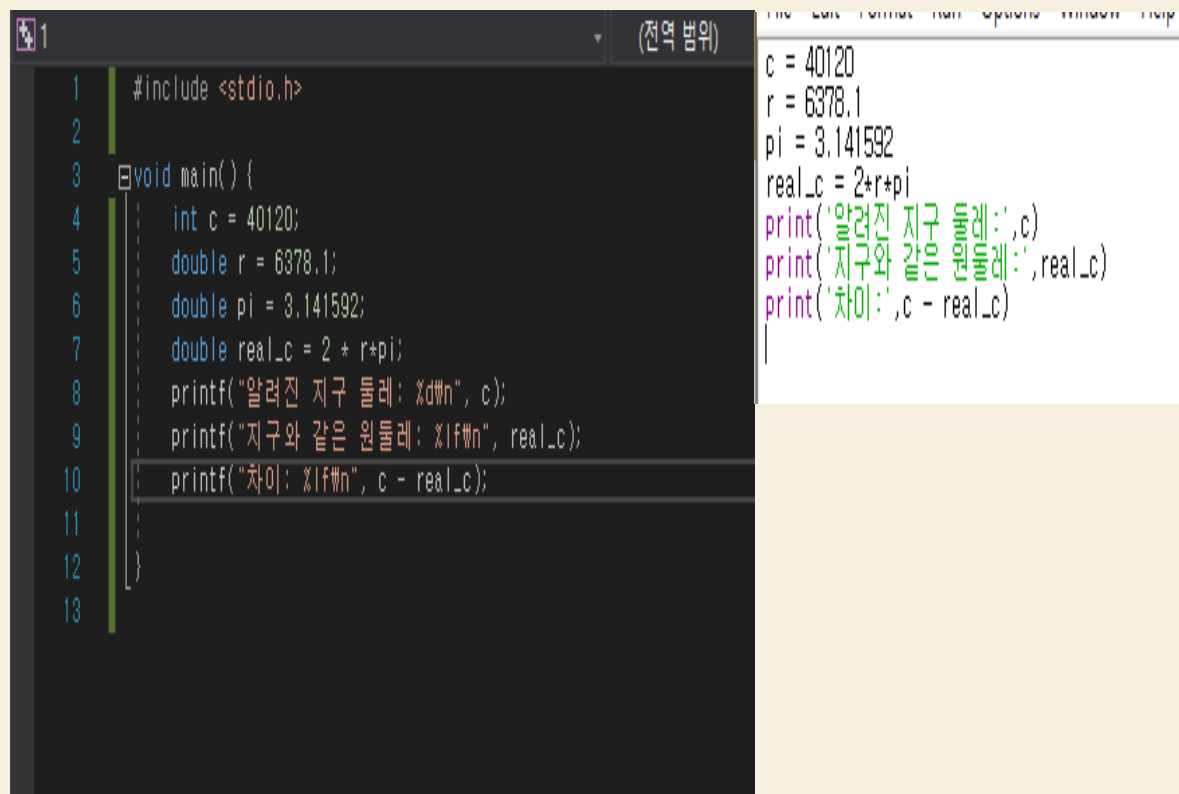


16 주차	[기말고사]
학습주제	직무수행능력평가 2차(기말고사)
목표및 내용	직무수행능력평가, 서술형평가
미리읽어오기	8장에서 13장까지
과제,시험,기타	
수업지원 안내	장애학생을 위한 별도의 수강 지원을 받을 수 있습니다. 언어가 문제가 되는 학생은 글로 된 과제 안내, 확대문자 시험지 제공 등의 지원을 드립니다.

Python이란?

1. 간결하고 생산성높은 창의적 용어

- 최근 주목받고 있는 언어로 다른 프로그래밍 언어(C, JAVA 등)에 비해 문법이 간결하여 입문자가 이해하기 쉽고 다양한 분야에서 활용할 수 있기 때문에 파이썬을 배우는 전공자가 늘어나고 있는 추세이다.



```
#include <stdio.h>

void main() {
    int c = 40120;
    double r = 6378.1;
    double pi = 3.141592;
    double real_c = 2 * r * pi;
    printf("알려진 지구 둘레: %d\n", c);
    printf("지구와 같은 원둘레: %lf\n", real_c);
    printf("차이: %lf\n", c - real_c);
}
```

```
c = 40120
r = 6378.1
pi = 3.141592
real_c = 2*r*pi
print('알려진 지구 둘레:', c)
print('지구와 같은 원둘레:', real_c)
print('차이:', c - real_c)
```

직접 예제를 보고 파이썬과 c언어의 코딩을 비교해보자.

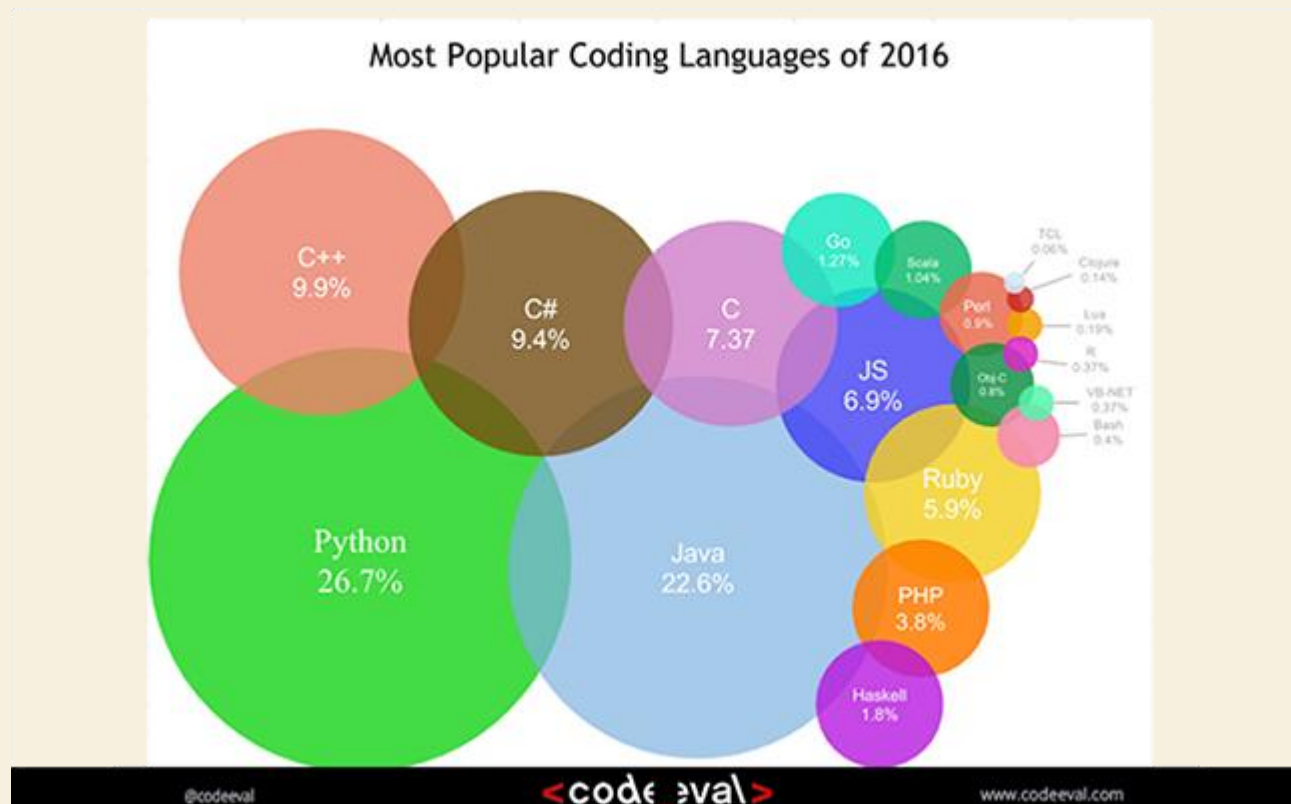
왼쪽에 있는 코딩은 c언어의 코딩이고 오른쪽에 있는 것은 파이썬의 코딩이다. 큰 차이 없어보일수도 있지만 c언어는 변수의 형식을 지정해주면서 선언했고, 파이썬은 그렇지 않다. 출력을 할때에도 차이를 보인다.

이 외에도 파이썬은 머신러닝, 그래픽, 웹 개발 등 여러 업계에서 선호하는 언어로 꾸준히 성장하고 있다.

수업 중에 주로 사용하는 프로그램은 편집기인 python IDLE이다.

Python이란?

2. 파이썬의 장, 단점



파이썬은 코드 경진대회를 제공하는 코드이벨에서 '2016 프로그래밍 인기순위 1위'로 뽑혔으며 프로그래밍 인기 순위를 집계하는 여타 다른 사이트에서도 5위안에 들어갈 만큼 인기가 높은 언어이다. 어떤 장점이 있어 이렇게 인기가 높을까?

파이썬은 문법이 간결하고 표현 구조가 인간의 사고 체계와 닮아있어 초보자도 쉽게 배울 수 있다. 또한 외부에 다양한 라이브러리가 있어 웹 개발, 데이터분석, 머신러닝, 그래픽 등 다양한 분야에서 사용할 수 있다.

물론 파이썬도 단점이 있는데 속도가 느리고 모바일 앱 개발 환경에서 사용하기 힘들며 컴파일시 타입 검사가 이뤄지지 않아 개발자가 실수할 여지가 좀 더 많다는 점 등이 있다.

Python 예제 풀이

```
print('다섯 문자 이상의 문자열 입력')
a = input()
while len(a) < 5 :
    print('입력한 문자가 다섯자 미만입니다.')
    print('다섯 문자 이상의 문자열 입력')
    a = input()
print('입력 문자열:' + a[:])
print('첫 문자:' + a[0])
print('마지막 문자:' + a[-1])
print('첫 문자를 제외한 부분 문자열:' + a[1:])
print('마지막 문자를 제외한 부분 문자열:' + a[:-1])
print('맨 앞과 뒤의 두 문자씩을 제외한 부분 문자열:' + a[1:-1])
print('문자 하나씩을 건너뛴 부분 문자열:' + a[::2])
print('역문자열:' + a[::-1])
|
```

파이썬의 코드가 간략하긴 하지만 지금까지 배웠던 다른 프로그래밍 언어와 크게 다르진 않다. 그래서 다른 프로그래밍 언어를 배웠다면 파이썬을 이해하는데 큰 어려움이 없다.

왼쪽의 예제 코드는 문자열 입력 시 부분 문자열을 출력하는 코딩인데 print를 통해 출력을 하고 input을 통해 사용자가 주는 정보를 입력받는다.

결과값은 아래와 같다.

```
다섯 문자 이상의 문자열 입력
String
입력 문자열:String
첫 문자:S
마지막 문자:g
첫 문자를 제외한 부분 문자열:tring
마지막 문자를 제외한 부분 문자열:Strin
맨 앞과 뒤의 두 문자씩을 제외한 부분 문자열:trin
문자 하나씩을 건너뛴 부분 문자열:Srn
역문자열:gnirtS
```

Python 예제 풀이

```
url = 'https://docs.python.org/3/tutorial'  
print(url[:url.find(':')])  
print(url[url.find('docs'):url.find('/3/')])  
print(url[url.find('tu'):])
```

```
https  
docs.python.org  
tutorial  
■
```

앞 예제의 응용 버전으로 find() 함수를 통해 입력한 부분을 찾아주고 그 부분의 앞, 뒤 부분 문자열을 출력해주는 프로그램이다.

Python 예제 풀이

```
a, b = input('실수 두개 입력 ').split()
x, y = float(a), float(b)

print('{} > {} 결과: {}'.format(x, y, x > y))
print('{} >= {} 결과: {}'.format(x, y, x >= y))
print('{} < {} 결과: {}'.format(x, y, x < y))
print('{} <= {} 결과: {}'.format(x, y, x <= y))
print('{} == {} 결과: {}'.format(x, y, x == y))
print('{} != {} 결과: {}'.format(x, y, x != y))
```

```
실수 두개 입력 -9 -2
-9.0 > -2.0 결과: False
-9.0 >= -2.0 결과: False
-9.0 < -2.0 결과: True
-9.0 <= -2.0 결과: True
-9.0 == -2.0 결과: False
-9.0 != -2.0 결과: True
```

이 예제는 숫자 두개를 여러방식으로 비교하여 참, 거짓을 가리는 프로그램으로 input()을 통해 숫자 두개를 입력받고 split()함수가 공백을 기준으로 두 숫자를 나누어 각각의 a, b변수에 할당시킨다. 이때, float를 통해 a, b변수를 실수형으로 변환시켜주어야 한다. 이 과정을 뺀다면 음수를 인식하지 못해 결과값에 에러가 발생할 수 있다.

Format함수는 print에 있는 {}값에 괄호안에 값을 할당해서 출력하는 일을 한다. {}값 안에 숫자를 써서 변수가 들어갈 순서를 정할 수도 있다.

Python 예제 풀이

```
a = int(input('정수 입력'))
b = int(input('2의 지수승 입력'))

print('정수 {} << {}, 결과: {}'.format(a, b, a << b))
print('정수 {} * 2**{}, 결과: {}'.format(a, b, a * 2**b))
print('2진수(32비트): {0:32b} 정수: {0}'.format(a))
print('2진수(32비트): {0:32b} 정수: {1} << {2}'.format(a << b, a, b))
print('2진수(32비트): {0:32b} 정수: {1} * 2**{2}'.format(a * 2**b, a, b))
```

정수와 2의 지수승을 입력받아 2의 b승을 a와 곱한 결과가 a를 이진수로 바꾸어서 b만큼 자리수를 이동한 결과가 같음을 보여주는 예제이다.

2**8은 2의 8승을 뜻하는 연산이고 5 << 8은 5를 이진수로 바꿔 왼쪽으로 8비트만큼 이동한다는 뜻이다. 즉, $5 * 2^{**8} = 5 << 8$ 과 같다고 볼 수 있다.

```
정수 입력5
2의 지수승 입력8
정수 5 << 8, 결과: 1280
정수 5 * 2**8, 결과: 1280
2진수(32비트):
2진수(32비트):
2진수(32비트):
101 정수: 5
10100000000 정수: 5 << 8
10100000000 정수: 5 * 2**8
```


Python 예제 풀이

조건 충족 시

```
height = 152 # 탑승을 체크할 키를 대입
if 140 <= height:
    print('롤러코스터 T-Express, 즐기세요!')
```

```
'''
롤러코스터 T-Express, 즐기세요!
'''
```

조건 미충족 시

```
height = 138 # 탑승을 체크할 키를 대입
if 140 <= height:
    print('롤러코스터 T-Express, 즐기세요!')
else:
    print('이용 불가능합니다.')
```

```
'''
이용 불가능합니다.
'''
```

어떠한 조건을 걸어 그 조건을 충족시킬 때, ‘다음 문장을 실행하는 조건문 if’를 가장 기본적인 형태로 풀어낸 예제이다. 위 예제에서는 height 변수가 152이므로 조건인 140보다 높아 롤러코스터를 탑승해도 된다는 문구가 뜨고 아래 예제에서 height 변수가 138이므로 140보다 낮아 if문에서 탈락하고 else문으로 넘어가 탑승이 불가능하다는 문구가 뜨게 된다.

Python 예제 풀이

```
from time import localtime
hour = localtime().tm_hour
mnt = localtime().tm_min

if hour < 10:
    print('지금 시각: %d시 %d분, 조조 할인 된다. ' % (hour, mnt))
else:
    print('지금 시각: %d시 %d분, 조조 할인 안된다. ' % (hour, mnt))
```

지금 시각: 18시 48분, 조조 할인 안된다.

오후 6:48

이전 예제와 똑같은 조건문이지만 라이브러리에서 상속받아 함수를 사용한다는 점이 다르다. From time import localtime이란 구문은 time 라이브러리에서 localtime, 즉 현재시간을 받아와 사용한다는 뜻으로 그 아래에 있는 함수인 localtime().tm_hour(현재 몇시인지 받아옴), localtime().tm_min(현재 몇분인지 받아옴)는 위에 있는 구문을 통해 상속받지 않고서는 사용할 수 없다. 아래 있는 결과값과 옆에 표시된 컴퓨터 시간을 보면 같음을 알 수 있다.

Python 예제 풀이

```
category = int(input('원하는 음료는? 1. 커피 2. 주스 '))

if category == 1:
    menu = int(input('번호 선택? 1.아메리카노 2.카페라테 3.카푸치노 '))
    if menu == 1:
        print('1.아메리카노 선택')
    elif menu == 2:
        print('2. 카페라테 선택')
    elif menu == 3:
        print('3.카푸치노 선택')
else:
    menu = int(input('번호 선택? 1. 키위주스 2. 토마토주스 3.오렌지주스 '))
    if menu == 1:
        print('1.키위주스 선택')
    if menu == 2:
        print('2.토마토주스 선택')
    if menu == 3:
        print('3.오렌지주스 선택')
```

커피 선택 시

```
원하는 음료는? 1. 커피 2. 주스 1
번호 선택? 1.아메리카노 2.카페라테 3.카푸치노 3
3.카푸치노 선택
```

주스 선택 시

```
원하는 음료는? 1. 커피 2. 주스 2
번호 선택? 1. 키위주스 2. 토마토주스 3.오렌지주스 2
2.토마토주스 선택
```

이번 예제는 이중 조건문이다. Category변수를 입력받는 값에 따라 달라지는 첫번째 조건문을 걸고 menu변수를 추가하여 입력받는 값에 따라 두번째 조건문을 걸어 출력값이 달라지게 코딩했다.

Elif는 else와 달리 if의 조건에는 거짓이지만 다른 조건을 달고 싶을 때 사용한다. 아래 else문에 if처럼 if문 다음 elif를 사용하지않고 if를 사용할 수도 있으나 만약

```
a = 7
if a < 10:
    print('한자리수')
if a < 100:
    print('두자리수')
```

한자리수
두자리수

만약 이 코딩과 같은 경우 elif를 썼다면 '한자리수'라는 결과만 출력됐겠지만 if를 사용하여 두조건을 모두 만족하여 '한자리수', '두자리수' 결과가 모두 출력되므로 이런 경우 사용을 조심해야한다.

Python 예제 풀이

```
for i in range(3):
    coffee = input("주문하세요! [아메리카노] [카페라테] [카푸치노] >> ")
    if coffee == '아메리카노':
        print('%s 주문' % coffee)
    elif coffee == '카페라테':
        print('%s 주문' % coffee)
    elif coffee == '카푸치노':
        print('%s 주문' % coffee)
    else:
        print('모르겠어요.')
else:
    print('주문을 마치겠습니다.')
```

반복문중 하나인 for문에 관한 기본 예제이다. For문은 보통 for 변수 in range(반복횟수): 방법으로 사용하는데 뜻 그대로 풀어서 설명하자면 i가 현재 반복횟수이고 한번 반복할때마다 1씩 올라가며 그것을 range괄호 안에 있는 숫자가 될때까지 반복하겠다는 뜻이다. 따라서 이 예제도 주문을 세번 받을때까지 반복되며 3번을 마치고 난 후엔 아래 else문으로 빠져 '주문을 마치겠습니다.'가 출력되는 것이다.

```
주문하세요! [아메리카노] [카페라테] [카푸치노] >> 아메리카노
아메리카노 주문
주문하세요! [아메리카노] [카페라테] [카푸치노] >> 카페라테
카페라테 주문
주문하세요! [아메리카노] [카페라테] [카푸치노] >> 카푸치노
카푸치노 주문
주문을 마치겠습니다.
```

Python 예제 풀이

```
for i in range(2,10):  
    for j in range(1,10):  
        print('%d * %d = %2d' % (i,j,i*j), end= ' ')  
    print()
```

```
2 * 1 = 2 2 * 2 = 4 2 * 3 = 6 2 * 4 = 8 2 * 5 = 10 2 * 6 = 12 2 * 7 = 14 2 *  
8 = 16 2 * 9 = 18  
3 * 1 = 3 3 * 2 = 6 3 * 3 = 9 3 * 4 = 12 3 * 5 = 15 3 * 6 = 18 3 * 7 = 21 3 *  
8 = 24 3 * 9 = 27  
4 * 1 = 4 4 * 2 = 8 4 * 3 = 12 4 * 4 = 16 4 * 5 = 20 4 * 6 = 24 4 * 7 = 28 4 *  
8 = 32 4 * 9 = 36  
5 * 1 = 5 5 * 2 = 10 5 * 3 = 15 5 * 4 = 20 5 * 5 = 25 5 * 6 = 30 5 * 7 = 35 5 *  
8 = 40 5 * 9 = 45  
6 * 1 = 6 6 * 2 = 12 6 * 3 = 18 6 * 4 = 24 6 * 5 = 30 6 * 6 = 36 6 * 7 = 42 6 *  
8 = 48 6 * 9 = 54  
7 * 1 = 7 7 * 2 = 14 7 * 3 = 21 7 * 4 = 28 7 * 5 = 35 7 * 6 = 42 7 * 7 = 49 7 *  
8 = 56 7 * 9 = 63  
8 * 1 = 8 8 * 2 = 16 8 * 3 = 24 8 * 4 = 32 8 * 5 = 40 8 * 6 = 48 8 * 7 = 56 8 *  
8 = 64 8 * 9 = 72  
9 * 1 = 9 9 * 2 = 18 9 * 3 = 27 9 * 4 = 36 9 * 5 = 45 9 * 6 = 54 9 * 7 = 63 9 *  
8 = 72 9 * 9 = 81  
...
```

구구단 출력 프로그램이다. Range(a, b)일때 i는 a부터 시작해서 b-1까지 올라가므로 i의 범위를 2부터 시작해서 9까지 올라가는 것으로 정하고 이중 반복문을 사용하여 j의 범위를 1부터 9까지 올라가는 것으로 정하여 첫 반복문이 한번 반복할 때 두번째 반복문이 9번 반복하게 된다. 따라서 i가 2일때 j가 1부터 9까지 반복한 후 다시 i가 3일때 j가 1부터 반복해서 i가 9가 될때까지 계속 반복하게 된다. 이렇게해서 쉽게 구구단 코딩을 할 수 있다.

Python 예제 풀이

```
winnumber = 11, 17, 28, 30, 33, 35
print(' 모의 로또 당첨 번호 '.center(28, '='))
print(winnumber)
print()
print(' 내 번호 확인 '.center(30, '-'))
cnt = 0
import random
for i in range(6):
    n = random.randint(1,45)
    if n in winnumber:
        print(n, 'O', end = ' ')
        cnt += 1
    else:
        print(n, 'X ', end = ' ')
print()
print(cnt, '개 맞음')
```

```
===== 모의 로또 당첨 번호 =====
(11, 17, 28, 30, 33, 35)

----- 내 번호 확인 -----
16 X 28 O 39 X 38 X 32 X 19 X
1 개 맞음
>>> |
```

로또번호 추첨 프로그램이다. 임의의 당첨번호를 정하고 당첨 개수를 세기 위한 변수 cnt를 선언한다. 그리고 randint함수를 쓰기위해 random을 상속 받고 6번 반복하는 for문을 만든 후 random.randint(1,45)에서 발생된 난수를 담은 변수 n을 선언한다. Random.randint(1,45)는 1~45사이의 임의의 난수를 발생하는 함수이다. If문을 통해 임의의난수가 당첨번호에 포함되는 경우 숫자와 함께 O를 프린트하고 당첨개수를 한 개 올린다. 그렇지 않을 시 숫자와 함께 X를 프린트한다.

Python 예제 풀이

```
days = ['monday', 'tuesday', 'wednesday']

while True:
    user = input('월, 화, 수 중 하나 영어 단어 입력 >> ')
    if user not in days:
        print("잘못 입력했어요!")
        continue
    print("입력: %s, 철자가 맞습니다." % user)
    break
print('종료 '.center(15, '*'))
```

```
>>> 월, 화, 수 중 하나 영어 단어 입력 >> mon
잘못 입력했어요!
>>> 월, 화, 수 중 하나 영어 단어 입력 >> mond
잘못 입력했어요!
>>> 월, 화, 수 중 하나 영어 단어 입력 >> monda
잘못 입력했어요!
>>> 월, 화, 수 중 하나 영어 단어 입력 >> monay
잘못 입력했어요!
>>> 월, 화, 수 중 하나 영어 단어 입력 >> monday
입력: monday, 철자가 맞습니다.
***** 종료 *****
>>>
```

반복문 중 하나인 while문에 관련된 예제이다. 철자를 틀리게 입력하면 continue문구를 통해 다시 처음으로 돌아가서 철자를 바르게 입력할 때 까지 무한으로 반복된다. 철자를 바르게 입력하면 break문을 통해 루프를 그 만두고 빠져나온다.

Python 예제 풀이

```
coffee = ['에스프레소', '아메리카노', '카페라테', '카페모카']
print(coffee)
print(type(coffee))
num = 0
for s in coffee:
    num += 1
    print('%d. %s' % (num, s))
```

```
['에스프레소', '아메리카노', '카페라테', '카페모카']
<class 'list'>
1. 에스프레소
2. 아메리카노
3. 카페라테
4. 카페모카
... |
```

List관련 예제이다. 커피관련 단어들을 coffee라는 리스트에 담아 출력하고 type()함수는 괄호 안 변수의 타입을 리턴한다. For문을 통해 리스트에 있는 내용들을 하나씩 출력하고 앞에 1씩 증가하는 변수 num을 추가하여 번호를 매겼다.

Python 예제 풀이

```
pl = ['C', 'C++', 'Python', 'Java']  
print(pl[0])  
print(pl[2])  
print()  
  
for i in range(len(pl)):  
    print(pl[i])
```

```
C  
Python  
  
C  
C++  
Python  
Java  
....
```

리스트의 순서는 0부터 시작한다. 따라서 왼쪽 예제에서 첫번째 있는 'C'를 출력하려면 pl[1]이 아닌 pl[0]을 출력해야 한다. For문을 리스트의 길이만큼 반복하여 pl[0], pl[1], pl[2], pl[3]을 순서대로 출력했다.

Python 예제 풀이

```
File Edit Format Run Options Window Help
food = ['짜장면', '짬뽕', '우동', '울면']
print(food)
# 탕수육 주문 추가
food.append('탕수육')
print(food)
# 짬뽕을 굴짬뽕으로 주문 변경
food[1] = '굴짬뽕'
print(food)

# 우동을 물만두로 주문 변경
food[food.index('우동')] = '물만두'
print(food)
```

처음 리스트를 선언한 다음 food.append('탕수육')을 통해 food리스트에 탕수육 원소를 추가했다. 이때 추가된 원소는 리스트의 맨 뒤쪽으로 위치한다. 원소를 수정하는 법 두가지가 나왔는데 첫번째는 해당 원소의 순서를 직접 지정하여 재정의 하는것이고, 두번째는 food.index함수를 통해 원소의 순서를 리턴받아 재정의하는 것이다.

```
====
[ '짜장면', '짬뽕', '우동', '울면', '탕수육' ]
[ '짜장면', '짬뽕', '우동', '울면', '탕수육' ]
[ '짜장면', '굴짬뽕', '우동', '울면', '탕수육' ]
[ '짜장면', '굴짬뽕', '물만두', '울면', '탕수육' ]
>>> |
```


Python 예제 풀이

```
wlist = ['밥', '삼', '길', '죽', '꿈', '차', '떡', '복', '말']
print('wlist[:] = ', wlist[:])
print('wlist[:] = ', wlist[:])
print('wlist[::-1] = ', wlist[::-1])
# 오름차순
print('wlist[:] = ', wlist[:])
print(wlist[:3])
print(wlist[1::3])
print(wlist[2::3])
# 내림차순
print(wlist[::-2])
print(wlist[-1:-8:-3])
# 오름과 내림차순의 혼재
print(wlist[1:-1:])
print(wlist[-2:-9:-3])
```

어떠한 리스트를 자를때, 슬라이싱할때는 [:]를 쓰는데 이때 [시작:끝:주소간 차이] 라고 볼 수 있다. 이때 숫자를 생략한다면 그냥 오름차순 리스트가 된다. 따라서 wlist[:]와 wlist[:]의 출력값은 같고[::-1]은 주소간 차이가 -1이므로 내림차순으로 출력한다. [1::3]은 두번째 주소부터 3칸씩 건너 뛴 내용을 출력하고 [-2:-9:-3]은 wlist의 끝에서 두번째 '복' 부터 시작하여 아홉번째 '밥' 까지 3칸씩 역으로 건너뛰어 오름차순으로 출력한다는 의미이다.

```
>>> wlist[:] = ['밥', '삼', '길', '죽', '꿈', '차', '떡', '복', '말']
>>> wlist[:] = ['밥', '삼', '길', '죽', '꿈', '차', '떡', '복', '말']
>>> wlist[::-1] = ['말', '복', '떡', '차', '꿈', '죽', '길', '삼', '밥']
>>> wlist[:] = ['밥', '삼', '길', '죽', '꿈', '차', '떡', '복', '말']
>>> wlist[:3] = ['밥', '삼', '길']
>>> wlist[1::3] = ['삼', '죽', '떡']
>>> wlist[2::3] = ['길', '차', '복']
>>> wlist[::-2] = ['말', '죽', '떡', '차', '꿈', '삼', '밥']
>>> wlist[-1:-8:-3] = ['말', '죽', '떡', '차', '꿈', '삼', '밥']
>>> wlist[1:-1:] = ['삼', '길', '죽', '꿈', '차', '떡', '복']
>>> wlist[-2:-9:-3] = ['말', '죽', '떡', '차', '꿈', '삼', '밥']
```

Python 예제 풀이

```
# for문으로 리스트 생성
a = []
for i in range(10):
    a.append(i)
print(a)

# 컴프리헨션으로 리스트 생성
seq = [i for i in range(10)]
print(seq)

# for문으로 리스트 생성
s = []
for i in range(10):
    if i%2 == 1:
        s.append(i**2)
print(s)

# 컴프리헨션으로 리스트 생성
squares = [i**2 for i in range(10) if i%2 == 1]
print(squares)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[1, 9, 25, 49, 81]
[1, 9, 25, 49, 81]
```

개인적으로 처음 접했을 때 이해가 조금 힘들었던 컴프리헨션 예제이다. 리스트에 내용을 추가할 때 append함수를 사용하지만 내용간 규칙이 있는 리스트라면 컴프리헨션으로 생성하는게 좀더 효과적이다. For문을 이용하여 list 선언과 동시에 리스트의 내용을 추가하기 때문에 코딩의 길이를 많이 줄일 수 있다.

Python 예제 풀이

```
day1 = ('monday', 'tuesday', 'wednesday')
day2 = ('thursday', 'friday', 'saturday')
# ('sunday')로 하면 튜플이 아니고 문자열
day3 = ('sunday',)

day = day1 + day2 + day3
print(type(day))
print(day)

day = day1+day2+day3*3
print(day)
```

튜플에 관한 예제이다.

튜플은 리스트와 흡사하지만 소괄호를 사용하고 초기화 한 이후 수정이 불가능하다는 차이가 있다. 굳이 튜플을 사용하는 이유는 리스트는 변경 가능성을 대비하기에 튜플에 비해 메모리를 많이 차지하지만 튜플은 값이 바뀔 일이 없기에 구조가 단순하고 읽어오는 속도가 빠르다. 또 값이 변조될 일이 없기 때문에 참조만 하는 값들에는 튜플이 적합하다 할 수 있다.

```
<class 'tuple'>
('monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday')
('monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday',
'sunday', 'sunday')
>>> |
```

Python 예제 풀이

```
bts1 = {'그룹명': '방탄소년단', '인원수': 7, '리더': '김남준'}
bts1['소속사'] = '빅히트 엔터테인먼트';
print(bts1)
bts2 = dict(['그룹명', '방탄소년단'], ['인원수', 7])
print(bts2)
bts3 = dict((( '리더', '김남준'), ('소속사', '빅히트 엔터테인먼트'))))
print(bts3)

bts = dict(그룹명='방탄소년단', 인원수=7, 리더='김남준', 소속사='빅히트 엔터테인
# 구성원 추가
bts['구성원'] = ['RM', '진', '슈가', '제이홉', '지민', '뷔', '정국']

print(bts) # 전체 출력
print(bts['구성원']) # 구성원 출력
```

```
{'그룹명': '방탄소년단', '인원수': 7, '리더': '김남준', '소속사': '빅히트 엔터테
인먼트'}
{'그룹명': '방탄소년단', '인원수': 7}
{'리더': '김남준', '소속사': '빅히트 엔터테인먼트'}
{'그룹명': '방탄소년단', '인원수': 7, '리더': '김남준', '소속사': '빅히트 엔터테
인먼트', '구성원': ['RM', '진', '슈가', '제이홉', '지민', '뷔', '정국']}
['RM', '진', '슈가', '제이홉', '지민', '뷔', '정국']
>>> |
```

Dict() 에 관한 예제이다. 딕셔너리는 key와 value로 이루어져 있으며 콜론:을 기준으로 Key : value 쌍으로 이루어진다. 따라서 값을 찾을 때 리스트나 튜플처럼 순차적인 숫자로 찾지 않고 key값을 통해 값을 찾는다. {}를 사용해 딕셔너리를 정의할수 있고

변수명['key'] = 'value';

를 통해 수정, 삭제, 추가할 수 있다.
튜플과 리스트는 슬라이싱이 가능하지만 딕셔너리는 불가능하다

Python 예제 풀이

```
color = dict(검은색='black', 흰색='white', 녹색='green', 파란색='blue')
print(color)

# 항목 조회
print(color.get('녹색'))
print(color.get('노란색'))
print()

# 항목 추가
color['노란색'] = 'yellow'
print(color)
print()

# 항목 삭제
c = '흰색'
print('삭제: %s %s' % (c, color.pop('흰색')))
print(color)
c = '빨간색'
print('삭제: %s %s' % (c, color.pop(c, '없어요'))))

print('임의 삭제: {} '.format(color.popitem()))
print('임의 삭제 후: {} '.format(color))

c = '검은색'
del color[c]
print('{} 삭제 후: {}'.format(c, color))

# 모든 항목 삭제
color.clear()
print(color)
```

```
{'검은색': 'black', '흰색': 'white', '녹색': 'green', '파란색': 'blue'}
green
None

{'검은색': 'black', '흰색': 'white', '녹색': 'green', '파란색': 'blue', '노란색': 'yellow'}

삭제: 흰색 white
{'검은색': 'black', '녹색': 'green', '파란색': 'blue', '노란색': 'yellow'}
삭제: 빨간색 없어요
{'검은색': 'black', '노란색': 'yellow'}
삭제 후: {'검은색': 'black', '녹색': 'green', '파란색': 'blue'}
임의 삭제 후: {'검은색': 'green', '파란색': 'blue'}
```

딕셔너리명.get('key')를 통해 해당 key의 value 값을 불러 올 수 있다.

딕셔너리명.pop('key', 'String')을 통해 해당 key 와 value값을 삭제할 수 있고 만약 딕셔너리에 해당 키가 없을 시 'String'을 출력한다.
Del 딕셔너리명['key']를 통해서도 해당 키와 value값을 삭제할 수 있으며

딕셔너리명.clear()를 통해 해당 딕셔너리의 모든 값을 삭제할 수 있다.

Python 예제 풀이

```
from random import randrange
from random import sample

# randrange() 함수와 집합을 이용, 중복 제거
mylotto = set()
while True:
    num = randrange(1, 46)
    print(num, end=' ')
    mylotto.add(num)
    if len(mylotto) == 6:
        break
print()
print('집합: {}'.format(mylotto))
print('정렬 리스트: {}'.format(sorted(mylotto)))
print()

# sample() 함수를 이용하면 매우 간편
lotto = sample(range(1, 46), 6)
print('sample 함수 리스트: {}'.format(lotto))
print('sample 함수 정렬 리스트: {}'.format(sorted(lotto)))
```

```
4 39 18 6 32 32 9
집합: {32, 4, 6, 39, 9, 18}
정렬 리스트: [4, 6, 9, 18, 32, 39]

sample 함수 리스트: [19, 6, 11, 42, 40, 9]
sample 함수 정렬 리스트: [6, 9, 11, 19, 40, 42]
>>> |
```

Sample함수 관련 예제이다. Sample함수를 사용하기위해선 random에서 import 해야 하며 사용 방법은

Sample(range(시작범위, 끝범위), 반복횟수) 이다. Sample함수를 이용한다면 반복문과 randrange를 이용해야하는 보통의 난수반복추출보다 훨씬 간단하고 효과적으로 사용할 수 있다.

Python 예제 풀이

```
planets = set('해달별')
fruits = set(['감', '귤'])
nuts = {'밤', '잣'}
things = {('밤', '잣'), ('감', '귤'), '해달'}
# things = {['밤', '잣'], ('감', '귤'), '해달'} # 오류발생

print(planets)
print(fruits)
print(nuts)
print(things)
|
```

```
{'별', '해'}
{'귤', '감'}
{'밤', '잣'}
{('밤', '잣'), ('감', '귤'), '해달'}
```

Set는 집합이며 set를 사용하는 방법에는

A = Set('내용')

B = set(['내', '용'])

C = {'내', '용'}

이런 방법이 있다. 세트는 순서가 없고 중복이 되지 않으므로 그점에 주의하며 A같은 경우에는 '내용'에서 '내'와 '용'이 떨어져 각자 원소가 된다.

느낀 점

처음 파이썬을 접할 때는 마냥 걱정만 앞선 상태였다. 군대에 갔다와 학업을 2년이나 놓은 상태로 있다가 와보니 기억이 나는 것이 거의 없었기 때문이다. 그 상태에서 한번도 배워본 적이 없는 파이썬이라는 과목까지 새로 배우게 된다고 생각하니 '과연 내가 따라갈 수 있을까...'라는 생각이 많이 들었다.

하지만 막상 파이썬을 어느정도 배우고 나서 이러한 생각들이 많이 바뀌었다. 여타 프로그래밍 언어들과 크게 다르지 않고 오히려 더 간결하고 명확하니 배우는 게 쉽고 재밌었다. 이번 포트폴리오 작성 간 느낀 점은 내가 배운 것을 정리하는 것이 생각보다 쉽지 않다는 점이었다. 정리가 잘 되지 않으니 교수님의 의도와 달리 잘 정리해서 하나의 책처럼 만든 것이 아닌 그냥 예제만 정리해 놓은 다소 부끄러운 포트폴리오가 된 것 같다.

앞으로 포트폴리오를 쓸 일이 많을텐데 틈틈히 혼자 써보는 연습을 하여 수준높은 포트폴리오를 작성할 수 있도록 해야겠다는 필요성을 느꼈다.

감사합니다.