

1장. 파이썬이 뭔데?

- 파이썬기초 -

프로그래밍 언어?

MONITOR FOR 6802 1.4 9-14-80 TSC ASSEMBLER PAGE 2

```
C000                    ORG     ROM+$0000 BEGIN MONITOR
C000 8E 00 70    START    LDS     #STACK

*****
* FUNCTION: INITA - Initialize ACIA
* INPUT: none
* OUTPUT: none
* CALLS: none
* DESTROYS: acc A

0013            RESETA   EQU     %00010011
0011            CTLREG   EQU     %00010001

C003 86 13        INITA   LDA A   #RESETA   RESET ACIA
C005 B7 80 04                STA A   ACIA
C008 86 11                LDA A   #CTLREG   SET 8 BITS AND 2 STOP
C00A B7 80 04                STA A   ACIA

C00D 7E C0 F1                JMP     SIGNON   GO TO START OF MONITOR

*****
* FUNCTION: INCH - Input character
* INPUT: none
* OUTPUT: char in acc A
* DESTROYS: acc A
* CALLS: none
* DESCRIPTION: Gets 1 character from terminal
```

컴퓨터와 대화를 하기위한 도구

(컴퓨터가 어떤 목적을 수행하도록 일련의 과정을 안내)

초기에는 0과 1의 조합인 기계어 사용

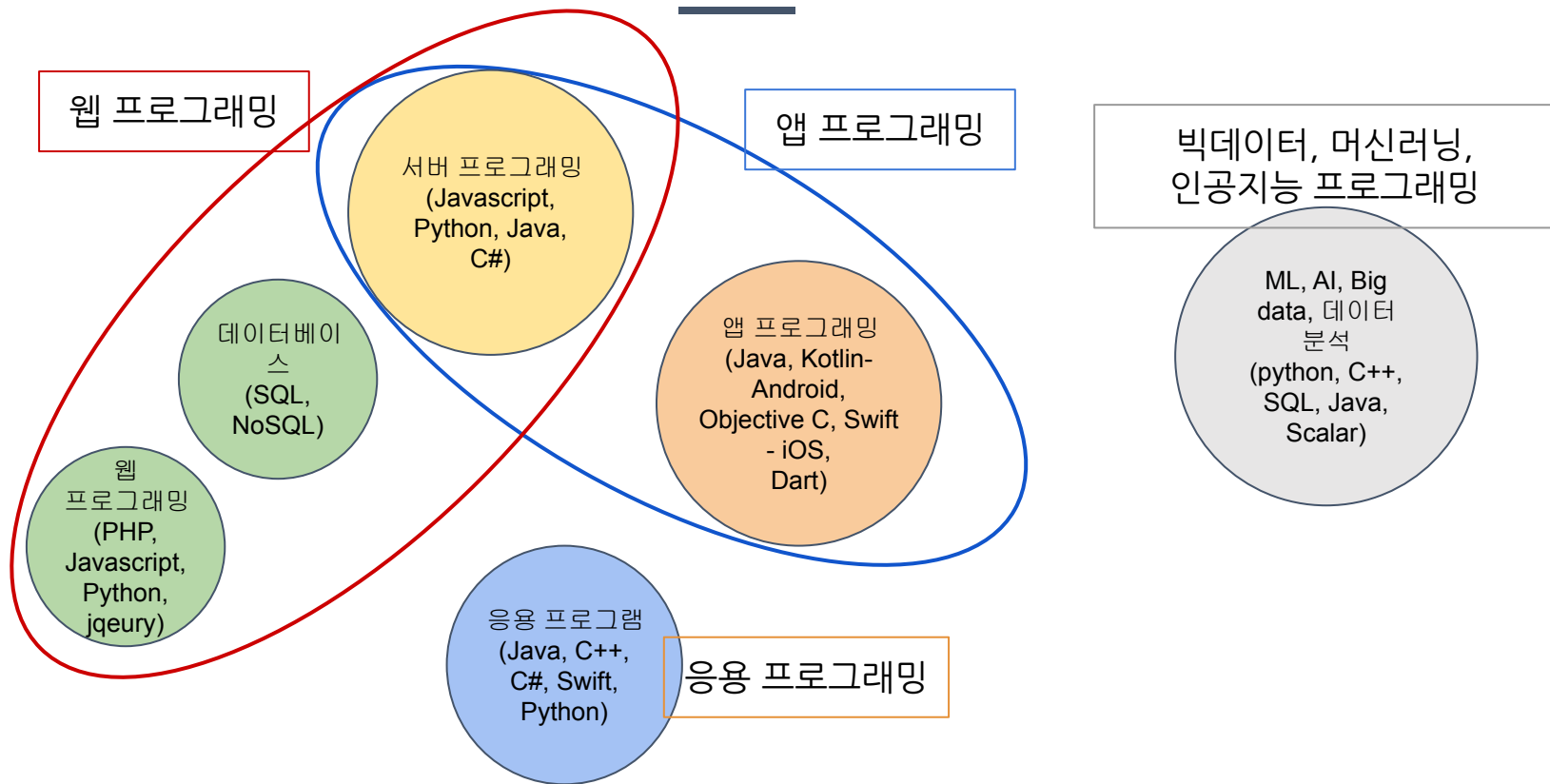
어셈블리어 (일부 기계어를 기호화)

파이썬과 같은 인간의 언어에 가까운 고급언어

이미지 출처

<https://terms.naver.com/entry.nhn?docId=2073348&cid=44414&categoryId=44414>

프로그래밍 분야와 적용 언어



파이썬이란?

귀도 반 로섬(Guido Van Rossum) 개발 - **심심해서**

BBC 방송 코미디 프로그램 (Monty Python's Flying Circus)

그리스 신화에 나오는 뱀 이름

1989년 개발시작 1990년 첫 버전 공개

대형 글로벌기업부터 스타트업까지 다양하게 활용
(구글, 유럽입자물리연구소, NASA, 핀인사이트....)



파이썬의 특징 (왜 파이썬인가?)

단순하고 간단하다

```
public class hello {  
    public static void main(string[] args) {  
        print("Hello World")  
        System.out.println("Hello World");  
    }  
}
```

파이썬의 특징 (왜 파이썬인가?)

방대한 라이브러리

웹개발, 게임개발, 데이터과학 등 범용언어

파이썬의 특징 (왜 파이썬인가?)

무료에 어느 운영체제든 사용가능

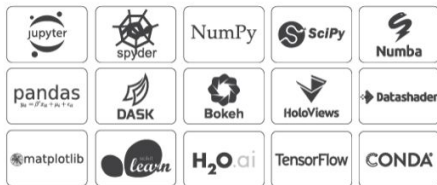
윈도우, 리눅스, 맥 모두 사용가능

2장. 파이썬 시작하기

- 파이썬기초 -

아나콘다?

1. 파이썬 내장
2. 과학계산을 위한 패키지 및 툴 포함

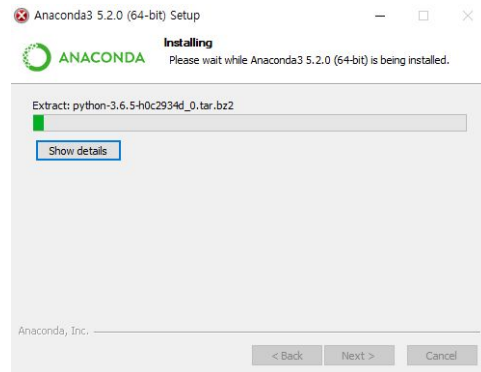
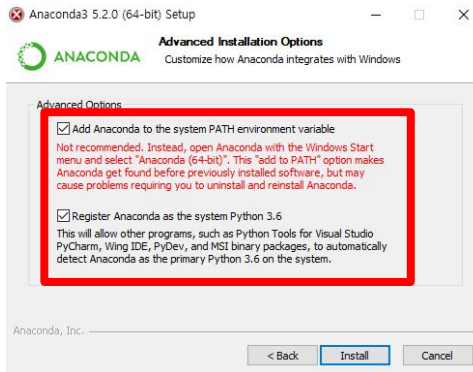
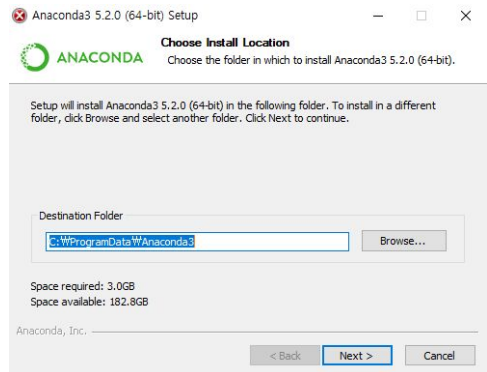
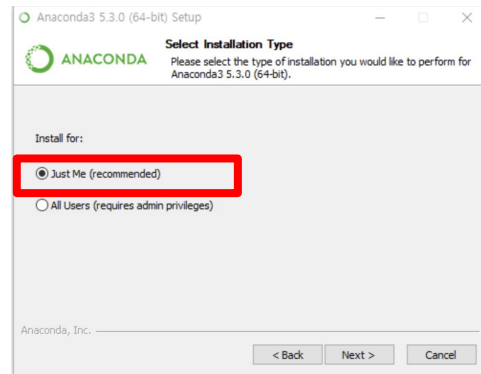
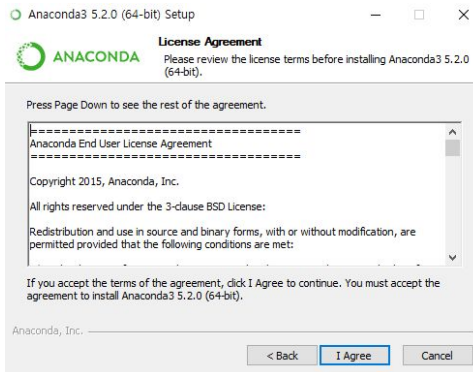


3. 가상환경 제공
(파이썬 다양한버전, 다양한 라이브러리 테스트)

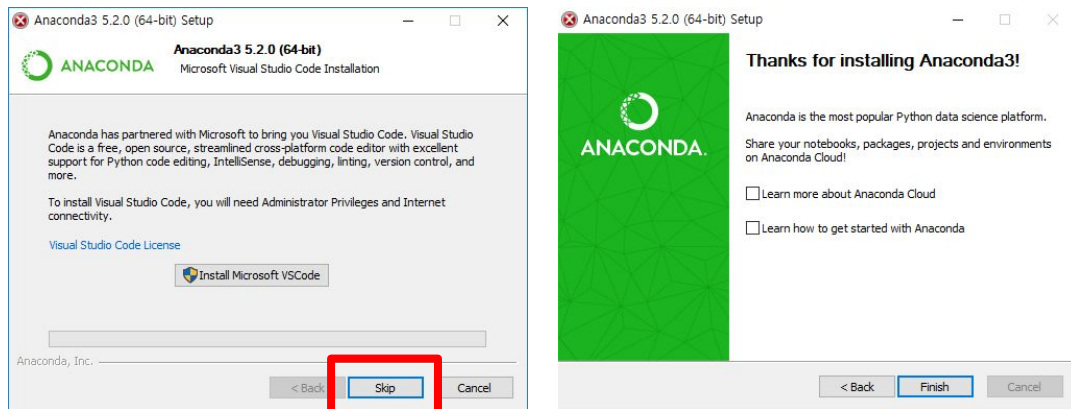
다운로드

<https://www.anaconda.com/distribution/>

아나콘다 설치

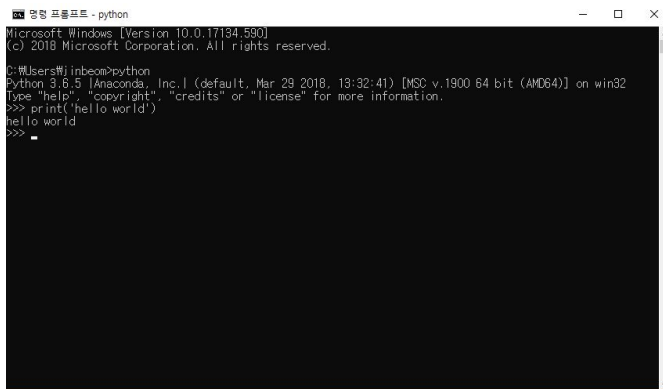
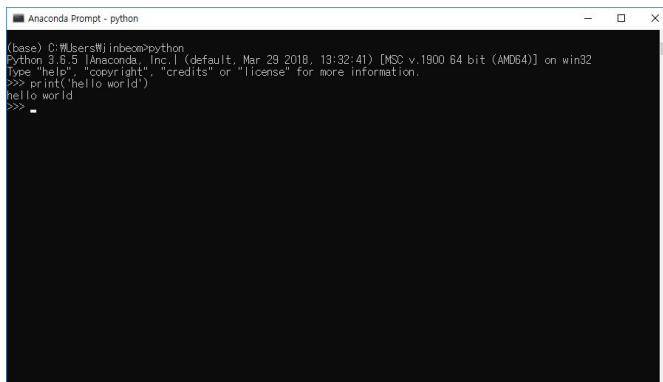
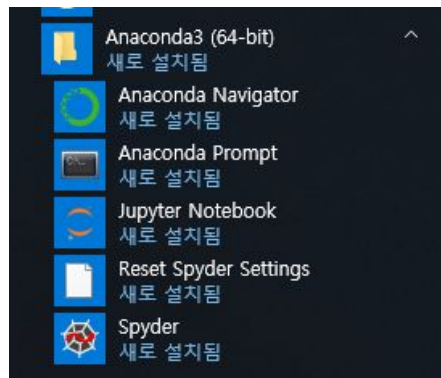


아나콘다 설치완료



IDE (Integrated Development Environment) : 효율적인 소프트웨어 개발을 위한 통합개발환경

아나콘다 설치완료

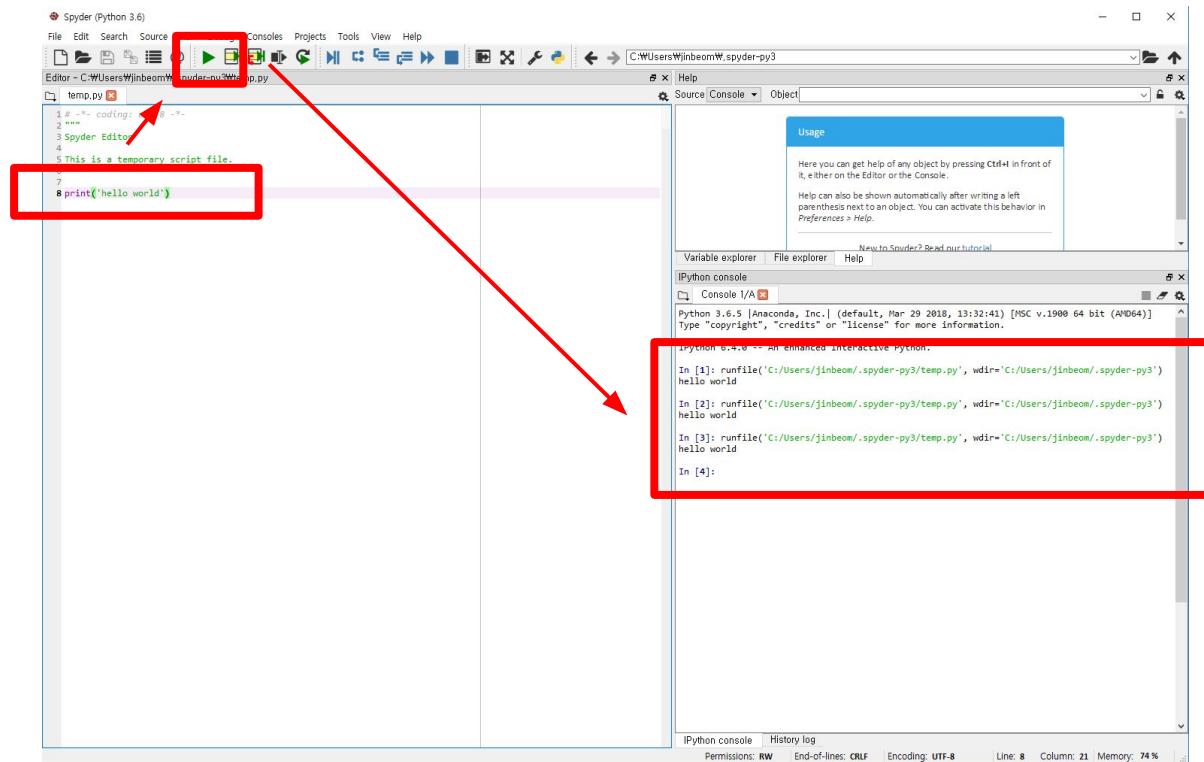


설치완료 테스트

python

>>> print('hello world')

Spyder 실행해보기



아나콘다 가상환경 만들기

가상환경생성

```
conda create -n study python=3.7
```

가상환경이름

파이썬버전

```

Anaconda Prompt - conda create -n study python=3.6

The following packages will be downloaded:

package | build | size
-----|-----|-----
sqlite-3.27.2 | he774522_0 | 941 kB
python-3.6.8 | h9f7ef89_7 | 20.3 MB
certifi-2019.3.9 | py36_0 | 156 kB
wheel-0.33.1 | py36_0 | 57 kB
pip-19.0.3 | py36_0 | 1.9 MB
vs2015_runtime-14.15.26706 | h3a45250_0 | 2.2 MB
vc-14.1 | h0510ff6_4 | 6 kB
setuptools-40.8.0 | py36_0 | 664 kB
Total: 28.2 MB

The following NEW packages will be INSTALLED:

certifi: 2019.3.9-py36_0
pip: 19.0.3-py36_0
python: 3.6.8-h9f7ef89_7
setuptools: 40.8.0-py36_0
sqlite: 3.27.2-he774522_0
vc: 14.1-h0510ff6_4
vs2015_runtime: 14.15.26706-h3a45250_0
wheel: 0.33.1-py36_0
winertstore: 0.2-py36h7fe50ca_0

Proceed ([y]/n)?

```

가상환경 리스트

```
conda info --envs
```

```
conda env list
```

```

(base) C:\Users\Wjinbeom>conda info --envs
# conda environments:
#
base * C:\ProgramData\Anaconda3
study C:\Users\Wjinbeom\AppData\Local\conda\conda\envs\study

```

아나콘다 가상환경 만들기

가상환경 활성화

conda activate study

```
(base) C:\Users\jinbeom>activate study  
(study) C:\Users\jinbeom>_
```

가상환경 비활성화

conda deactivate

```
(study) C:\Users\jinbeom>deactivate study  
deactivate does not accept arguments  
remainder_args: ('study',)
```

주피터 노트북 설치

pip install jupyter

```

Anaconda Prompt
(study) C:\Users\jinbeon\study>pip install jupyter
Collecting jupyter
  Downloading https://files.pythonhosted.org/packages/83/df/0f5dd132200728a86190997e1ea87cd76244e42d39ec5e88efd25b2abd7e/jupyter-1.0.0-py2.py3-none-any.whl
Collecting ipykernel (from jupyter)
  Downloading https://files.pythonhosted.org/packages/d8/b0/f0be5c5ab335196f5cce98e5b889a4fcf5bfe462eb0acc06cdf2e2caf65eb/ipykernel-5.1.0-py3-none-any.whl (113kB)
  100% |#####| 122kB 93kB/s
Collecting jupyter-console (from jupyter)
  Downloading https://files.pythonhosted.org/packages/cb/ee/6374ae8c21b7d0847f9c3722dcdfac986b8e54fa9ad9ea66e1eb6320d2b8/jupyter_console-6.0.0-py2.py3-none-any.whl
Collecting nbconvert (from jupyter)
  Downloading https://files.pythonhosted.org/packages/b8/39/1e67fea74dc9577cc49f9863fe3ec824e525d1304ab6027d95a94cd586f5/nbconvert-5.4.1-py2.py3-none-any.whl (407kB)
  100% |#####| 409kB 3.3MB/s
Collecting notebook (from jupyter)
  Downloading https://files.pythonhosted.org/packages/f6/36/89ebfffc3dd8c8dbd81c1ffb53e3d4233ee66b414c143959477cb07cc5f5/notebook-5.7.8-py2.py3-none-any.whl (9.0MB)
  100% |#####| 9.0MB 1.3MB/s
Collecting ipywidgets (from jupyter)
  Downloading https://files.pythonhosted.org/packages/30/9a/a008c7b1183fac9e5206d80a379b3c64eab535bd3d6cdc29a0b766fd82/ipywidgets-7.4.2-py2.py3-none-any.whl (111kB)
  100% |#####| 112kB 6.6MB/s
Collecting qtconsole (from jupyter)
  Downloading https://files.pythonhosted.org/packages/e0/7a/8aefbc0ed078dec7951ac9a06dcd1869243ecd7bcbce26fa47bf5e469a8f/qtconsole-4.4.3-py2.py3-none-any.whl (113kB)
  100% |#####| 122kB 6.5MB/s
Collecting jupyter-client (from ipykernel->jupyter)
  Downloading https://files.pythonhosted.org/packages/3b/c3/3043fe9ffdd140d09c9d091a056794ccdc427c
  
```

pip (Python Package Index)

파이썬의 여러 모듈, 라이브러리를 쉽게 설치 할 수 있도록 도와주는 라이브러리

참고사이트

<https://jupyter.org> - 주피터노트북 실행

<https://pip.pypa.io> - pip

주피터 노트북 실행

jupyter notebook

jupyter notebook list -> 현재 오픈된 노트북

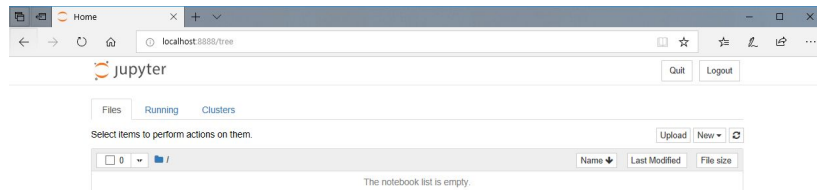
```

선택 Anaconda Prompt - jupyter notebook
Successfully built pandocfilters, prometheus-client, backcall, pyrsistent
Installing collected packages: tornado, six, python-dateutil, pyzmq, decorator, ipython-genutils,
traitlets, jupyter-core, jupyter-client, wcwidth, prompt-toolkit, backcall, pygments, parso, jedi,
colorama, pickleshare, ipython, ipykernel, jupyter-console, testpath, pyrsistent, attrs, jsons
chema, nbformat, mistune, defusedxml, webencodings, bleach, MarkupSafe, Jinja2, pandocfilters, en
trypoints, nbconvert, prometheus-client, pywinpty, terminado, Send2Trash, notebook, widgetsnbexte
nion, ipywidgets, qtconsole, jupyter
Successfully installed MarkupSafe-1.1.1 Send2Trash-1.5.0 attrs-19.1.0 backcall-0.1.0 bleach-3.1.0
colorama-0.4.1 decorator-4.4.0 defusedxml-0.5.0 entrypoints-0.3 ipykernel-5.1.0 ipython-7.4.0 ip
ython-genutils-0.2.0 ipywidgets-7.4.2 jedi-0.19.9 Jinja2-2.10.1 jsonschema-3.0.1 jupyter-1.0.0 jupy
ter-client-5.2.4 jupyter-console-6.0.0 jupyter-core-4.4.0 mistune-0.8.4 nbconvert-5.4.1 nbformat-
4.4.0 notebook-5.7.8 pandocfilters-1.4.2 parso-0.3.4 pickleshare-0.7.5 prometheus-client-0.6.0 pr
ompt-toolkit-2.0.9 pygments-2.3.1 pyrsistent-0.14.11 python-dateutil-2.8.0 pywinpty-0.5.5 pyzmq-1
8.0.1 qtconsole-4.4.3 six-1.12.0 terminado-0.8.2 testpath-0.4.2 tornado-6.0.2 traitlets-4.3.2 wcw
idth-0.1.7 webencodings-0.5.1 widgetsnbextension-3.4.2

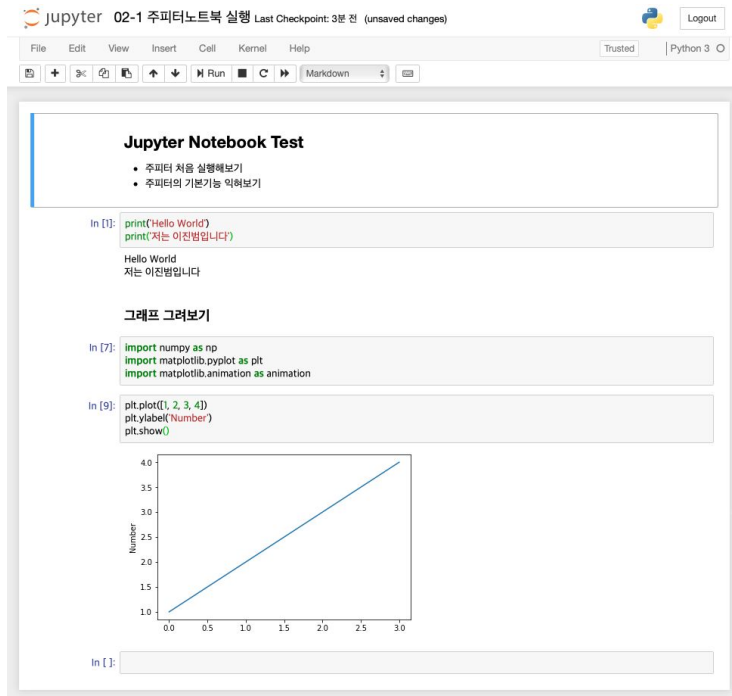
(study) C:\Users\jinbeom\study>jupyter notebook
[I 18:18:19.402 NotebookApp] Serving notebooks from local directory: C:\Users\jinbeom\study
[I 18:18:19.402 NotebookApp] The Jupyter Notebook is running at:
[I 18:18:19.402 NotebookApp] http://localhost:8888/?token=15683443b601c84ac8cf655007626a0c5fa1bdb3f06914e8
[I 18:18:19.402 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice t
o skip confirmation).
[C 18:18:19.402 NotebookApp]

To access the notebook, open this file in a browser:
file:///C:/Users/jinbeom/AppData/Roaming/jupyter/runtime/nbserver-404-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=15683443b601c84ac8cf655007626a0c5fa1bdb3f06914e8

```



주피터 노트북 실행



기본 단축키

Shift + Enter - 수행 후 아래셀 이동 (없으면 생성)

Alt + Enter - 수행 후 새로운 셀 추가

Ctrl + Enter - 수행

Ctrl + S - 노트북 저장

Enter - 입력모드로 전환

Esc - 입력모드에서 명령모드로 전환

M - 셀 타입을 마크다운으로 전환

Y - 셀 타입을 코드로 전환

Ctrl + / - 선택된 코드 주석처리

a 현재 셀 위쪽에 새로운 셀 추가

b 현재 셀 아래쪽에 새로운 셀 추가

dd 현재 셀 삭제

소스코드

https://www.edureka.co/blog/wp-content/uploads/2018/10/Jupyter_Notebook_CheatSheet_Edureka.pdf

02-1 주피터노트북 실행.ipynb

간단한 실습 - 사용자 입력, 출력

```
x = input("입력해주세요 : ")  
print(x)
```

소스코드

02-2 간단한 실습.ipynb

간단한 실습 - 주식

소스 흐름을 쉽게 파악할수 있도록 설명

한줄수식

“””

여러줄주식
여러줄주식

“””

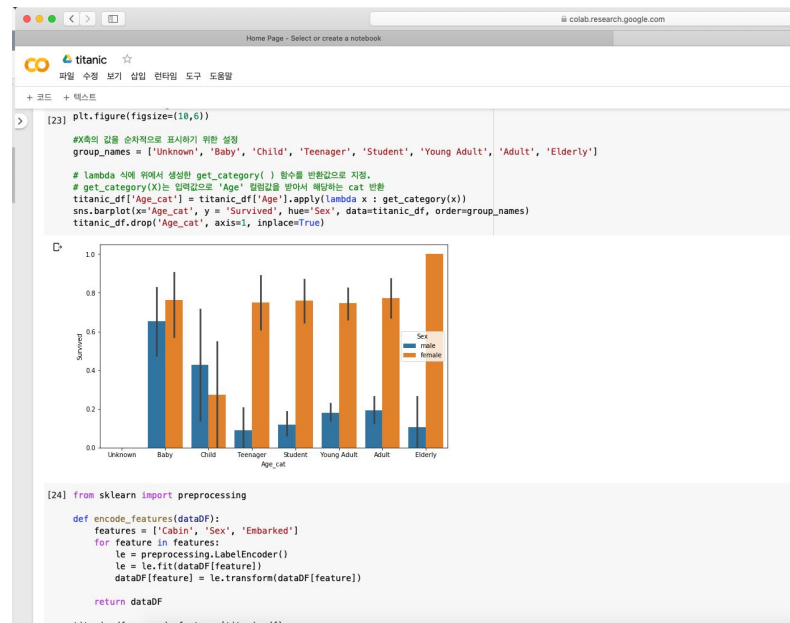
소스코드

02-2 간단한 실습.ipynb

Colab

Colaboratory는 설치가 필요 없으며 완전히 클라우드에서 실행되는 무료 Jupyter 노트 환경입니다.

Colaboratory를 사용하면 브라우저를 통해 무료로 코드를 작성 및 실행하고, 분석을 저장 및 공유하며, 강력한 컴퓨팅 리소스를 이용할 수 있습니다.



<https://colab.research.google.com/notebooks>

Error 확인하기

- 에러가 발생한 위치 및 문제 원인 확인
- 구글링 (ex : *'int' object is not callable*)

```
[1]: data = 1  
     print(data)
```

1

```
[2]: print = 1
```

```
[3]: data = 1  
     print(data)
```

```
-----  
TypeError                                Traceback (most recent call last)  
<ipython-input-3-479867dba81c> in <module>  
      1 data = 1  
----> 2 print(data)  
  
TypeError: 'int' object is not callable
```

3장. 산술연산

- 파이썬기초 -

사칙연산

구분	예제	결과
덧셈	$1 + 2$	3
뺄셈	$1 - 2$	-1
곱셈	$5 * 2$	10
나눗셈	$5 / 2$	2.5
나눗셈 (몫)	$5 // 2$	2
나눗셈 (나머지)	$9 \% 2$	1
제곱	$2 ** 3$	8
괄호 (우선순위)	$(2 + 3) * 2$	10

소스코드

03-1 산술연산.ipynb

숫자의 자료형

자료의 형태를 확인하는 함수

type(값)

```
>> type(10)
```

결과 : int

자료형

int - 정수

float - 실수

bool - True or False

str - 문자

list - list

dic - dictionary

tuple - tuple

set - 집합

```
>> type(10.5)
```

결과 : float

소스코드

03-1 산술연산.ipynb

숫자형 변환

자료형 변환함수

int(값) - 정수로 변환

float(값) - 실수로 변환

```
>>> float(3) #정수를 실수로
```

결과 : 3.0

```
>>> int(3.5) #실수를 정수로
```

결과 : 3

```
>>> int(5 / 2) #형변환을 이용한 몫구하기
```

결과 : 2

소스코드

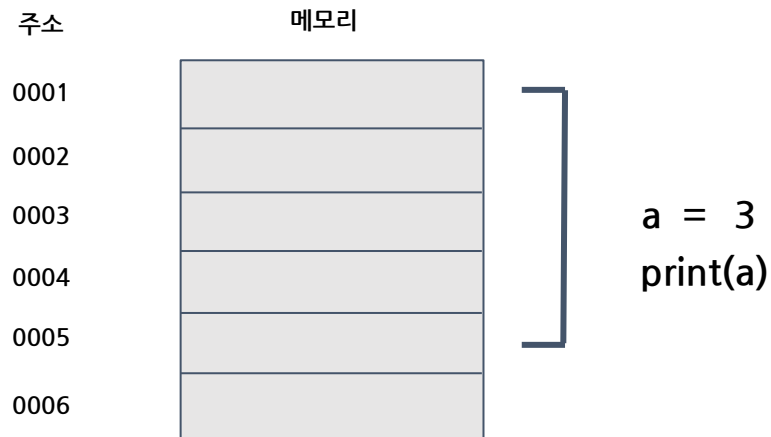
03-1 산술연산.ipynb

4장. 변수

- 파이썬기초 -

변수란?

하나의 데이터를 저장할 수 있는 메모리 공간, 변할수있는 수



프로그래밍이란 데이터(숫자, 문자열)를 기반으로 컴퓨터에 명령을 내리는 것인데 이 데이터를 저장하는 곳이 변수이다.

소스코드
04-1 변수.ipynb

변수사용방법

$$X = 10$$

변수명

값

$$X, Y, Z = 10, 20, 30$$

변수명

값

영문, 숫자 사용가능
 대소문자 구분
 숫자부터 시작할수 없음
 특수문자 사용불가 (+, -, *, /, \$ 등)
 파이썬의 키워드 사용불가

소스코드

04-1 변수.ipynb

변수로 계산하기

```
>>> x = 20
>>> y = 30
>>> c = x + y
>>> c
결과 : 50
```

```
>>> x = 10
>>> x + 10
>>> x
결과 : 10
```

```
>>> x = 10
>>> x = x + 10
>>> x
결과 : 20
```

```
>>> x = 10
>>> x += 10
>>> x
결과 : 20
```

비어있는 변수 (Null)

None

아무값도없는

```
>>> x = None
```

```
>>> print(x)
```

결과 : None

```
>>> type(x)
```

결과 : None Type

소스코드

04-1 변수.ipynb

실습01

사용자로부터 변수 x , 변수 y 를 입력받아
더하고 곱하는 프로그램을 작성하세요

실행 :

x 값을 입력해주세요 : 20

y 값을 입력해주세요 : 30

결과 :

50

소스코드

04-1 실습01.ipynb

5장. Boolean 과 비교, 논리연산자

- 파이썬기초 -

Boolean형

True

참

False

거짓

```
>>> True
```

결과 : True

```
>>> False
```

결과 : False

```
>>> Type(True)
```

결과 : bool

비교연산자

>

크다

>=

크거나같다

<

작다

<=

작거나같다

==

같다

!=

같지않다

연산결과는 Boolean

```
>>> print(3 > 1)
```

결과 : True

```
>>> print(10 != 10)
```

결과 : False

```
>>> Type(10==10)
```

결과 : bool

논리연산자

and

모두가 True 면 True
하나라도 False 면 False

```
>>> True and True
```

결과 : True

```
>>> True and False
```

결과 : False

```
>>> False and False
```

결과 : False

or

하나만 True 여도 True
모두가 False 면 False

```
>>> True or True
```

결과 : True

```
>>> True or False
```

결과 : True

```
>>> False or False
```

결과 : False

not

반대

```
>>> not True
```

결과 : False

```
>>> not False
```

결과 : True

비교 + 논리 연산자

$5 > 10$ and $20 == 20$

False

True

False

$5 == 5$ or $10 != 10$

True

False

True

실습01

사용자로 점수를 3개 입력받아
모든 점수가 65점보다 클 경우 True
점수가 하나라도 65점이 안넘을 경우 False
를 출력하세요

실행 :

첫번째 점수 :

20

두번째 점수 :

70

세번째 점수 :

80

결과 :

False

실행:

첫번째 점수 : 70

두번째 점수 : 80

세번째 점수 : 90

결과:

True

소스코드

05-1 실습01.ipynb

6장. 문자형

- 파이썬기초 -

문자형

```
first = "Hello, World"
```

```
multi = """Hello, World  
Hello, Python"""
```

소스코드

06-1 문자형.ipynb

문자열 연산

“과일” + “사과”
과일사과

“과일” * 3
과일과일과일

문자열 함수소개

len()

문자 길이

```
>>> word = "apple banana"
>>> len(word) #문자길이
12
```

upper()

대문자로 바꾸기

```
>>> word.upper()
1RANG2 BANANA
```

lower()

소문자 바꾸기

```
>>> word.lower()
1rang2 banana
```

replace(바꿀문자, 새문자)

문자열 바꾸기

```
>>> word.replace("apple", "orange")
orange banana
```

소스코드

06-1 문자형.ipynb

서식지정자 #1

“문자열 **%s** 문자열” % “추가문자”



```
>>> name = "tom"
```

```
>>> print("I am " + name + "!") # +사용
```

```
I am tom!
```

```
>>> print("I am %s!" % name) # 서식지정자사용
```

```
I am tom!
```

%(공백)s - 공백추가

```
>>> print("I am %10s!" % name)
```

```
I am   tom!
```

```
>>> print("I am %-10s!" % name)
```

```
I am tom  !
```

여러개의 추가문자

```
>>> f1, f2 = "apple", "banana"
```

```
>>> print("I like %s, %s!" % (f1, f2))
```

```
I like apple, banana
```

소스코드

06-1 문자형.ipynb

서식지정자 #2

“문자열 %s 문자열” % “추가문자”

```
>>> name = "tom"
```

```
>>> print("I am " + name + "!") # +사용
```

```
I am tom!
```

```
>>> print("I am %s!" % name) # 서식지정자사용
```

```
I am tom!
```

%s - 문자, %d - 정수, %f - 실수

```
>>> n1, n2 = 3, 3.2323
```

```
>>> print("n1 = %d, n2 = %f" % (n1, n2))
```

```
n1 = 3, n2 = 3.2323
```

%. (숫자)f - 소수점 이하 자리수

```
>>> print("n2 = %.2f" % n2)
```

```
n2 = 3.23
```

%(공백에 채울숫자)(공백)s - 숫자개수 맞추기

```
>>> print("n2 = %07.2f" % n2)
```

```
n2 = 0003.23
```

소스코드

06-1 문자형.ipynb

Format 함수

“문자열 {0}, {1} 문자열”.format(값, 값)

```
>>> “I like {0}, {1} !”.format(“apple”, “banana”)
```

```
I like apple, banana
```

```
>>> “Number {0} {2} {1}”.format(1,2,3)
```

```
Number 1 3 2
```

```
>>> “Number {0} {0} {1}”.format(1,2,3)
```

```
Number 1 1 2
```

```
>>> “Number {} {} {}”.format(1,2,3)
```

```
Number 1 2 3
```

변수사용

```
>>> f1, f2 = “apple”, “banana”
```

```
>>> “I like {0}, {1} !”.format(f1, f2)
```

```
I like apple, banana
```

{0:(숫자)<(숫자)} - 공백추가

```
>>> “Number {0:>4}!”.format(1)
```

```
Number    1!
```

```
>>> “Number {0:<4}!”.format(1)
```

```
Number 1   !
```

```
>>> “Number {0:0<4}!”.format(1)
```

```
Number 0001!
```

```
>>> “Number {0:0^5}!”.format(1)
```

```
Number 00100!
```

소스코드

06-1 문자형.ipynb

가치를 높이는 금융 인공지능 실무교육

Insight campus

실습01

(1) 문자를 입력받아 공백을 모두 제거하고 출력하세요

실행 :

아무 문자나 입력해주세요 : **안녕하세요 반갑습니다.**
안녕하세요반갑습니다.

(2) 이름과 점수3개를 입력받아 아래와 같이 출력하세요

실행 :

이름을 입력해 주세요 : **홍길동**
첫번째 점수를 입력해 주세요 : **80**
두번째 점수를 입력해 주세요 : **70**
세번째 점수를 입력해 주세요 : **60**
저의 이름은 홍길동 이고 총점은 210 입니다.

소스코드

06-1 실습01.ipynb

데이터 구조

- 리스트, 튜플, 딕셔너리, 세트 -

연관 있는 데이터를 효율적으로 관리하기 위한 구조

7장. 리스트와 튜플

- 파이썬기초 -

리스트와 튜플선언

시퀀스 자료형 - 연속된 여러값들을 한 변수에 저장

리스트 = [값, 값, 값 ...]

수정, 추가 가능

```
>>> a = [1, 2, 3, 4, 5, 6, 7]
>>> a = [1, "apple", 3.14, False]
>>> type(a)
<class 'list'>
>>> a = [] #빈값생성
>>> a = list()
```

튜플 = (값, 값, 값)

수정, 추가 불가능

```
>>> b = (1, 2, 3, 4, 5, 6, 7)
>>> b = (1, "apple", 3.14, False)
>>> type(b)
<class 'tuple'>
>>> b = () #빈값생성
>>> b = tuple()
```

소스코드

07-1 리스트와 튜플.ipynb

Range 함수

연속된 숫자를 생성하는 함수

range(끝)

```
>>> a = list(range(10))  
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
>>> a = list(range(1, 11))  
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
>>> a = tuple(range(10))  
(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
```

range(시작, 끝, 증가폭)

```
>>> a = list(range(-10, 10, 2))  
[-10, -8, -6, -4, -2, 0, 2, 4, 6, 8]
```

```
>>> a = list(range(5, 0, -1))  
[5, 4, 3, 2, 1]
```

소스코드

07-1 리스트와 튜플.ipynb

리스트, 튜플 형변환

list(튜플)

```
>>> a = (1, 2, 3, 4, 5)
>>> list(a)
[1, 2, 3, 4, 5]
```

tuple(리스트)

```
>>> a = [1, 2, 3, 4, 5]
>>> tuple(a)
(1, 2, 3, 4, 5)
```

소스코드

07-1 리스트와 튜플.ipynb

인덱스 접근 #1

기본접근 및 변경

```
>>> a = [1, 2, 3, 4, 5]
```

```
>>> a[2]
```

```
3
```

```
>>> a[-1]
```

```
5
```

```
>>> a[3] = 8
```

```
>>> a
```

```
[1, 2, 3, 8, 5]
```

슬라이스

```
>>> a = [1, 2, 3, 4, 5]
```

```
>>> a[0:4]
```

```
[1, 2, 3, 4]
```

```
>>> a[3:5]
```

```
[4, 5]
```

```
>>> a[1:-2]
```

```
[2, 3]
```

소스코드

07-1 리스트와 튜플.ipynb

인덱스 접근 #2

증가폭 변경

```
>>> a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> a[2: 8: 2]
[3, 5, 7]
```

끝 인덱스까지 가져오기

```
>>> a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> a[:3]
[1, 2, 3]
>>> a[3:]
[4, 5, 6, 7, 8, 9, 10]
```

슬라이스 요소할당

```
>>> a = [1, 2, 3, 4, 5]
>>> a[1:4] = ["a", "b", "c"]
>>> a
[1, a, b, c, 5]
>>> a[1:4] = ["d", "e"]
>>> a
[1, d, e, 5]
```

슬라이스 삭제

```
>>> a = [1, 2, 3, 4, 5]
>>> del a[1:4]
[1, 5]
```

소스코드

07-1 리스트와 튜플.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

리스트와 튜플 기능들 #1

특정값 있는지 확인 in

```
>>> a = [1, 2, 3, 4, 5]
>>> 1 in a
True
>>> 6 in a
False
```

연결하기 +

```
>>> a = [1, 2, 3, 4, 5]
>>> b = [6, 7, 8, 9]
>>> a + b
1, 2, 3, 4, 5, 6, 7, 8, 9
```

반복하기 *

```
>>> a = [1, 2, 3]
>>> a * 3
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

요소개수 구하기 len()

```
>>> a = [1, 2, 3]
>>> len(a)
3
```

소스코드

07-1 리스트와 튜플.ipynb

리스트 기능들 #1

요수 추가하기 `append(값)`, `extend(값)`

```
>>> a = [1, 2, 3, 4, 5]
>>> a.append(6)
[1, 2, 3, 4, 5, 6]
>>> a.extend([7, 8])
[1, 2, 3, 4, 5, 6, 7, 8]
```

리스트 요소삭제 `pop(인덱스)`

```
>>> a = [1, 2, 3]
>>> a.pop(0)
[2, 3]
>>> a.pop()
[2]
```

특정 인덱스에 요소추가 `insert(인덱스, 값)`

```
>>> a = [1, 2, 3]
>>> a.insert(2, 100)
[1, 2, 100, 3]
```

리스트 특정값을 찾아삭제 `remove(값)`

```
>>> a = [100, 200, 300]
>>> a.remove(200)
[100, 300]
```

소스코드

07-1 리스트와 튜플.ipynb

리스트 기능들 #2

특정값의 인덱스 구하기 index(값)

```
>>> a = [1, 2, 3, 4, 5]
>>> a.index(3)
2
```

특정값의 개수구하기 count(값)

```
>>> a = [1, 1, 2, 2, 2, 3, 3]
>>> a.count(2)
3
```

순서 뒤집기 reverse()

```
>>> a = [1, 2, 3]
>>> a.reverse()
[3, 2, 1]
```

정렬하기(오름차순) sort(), sort(reverse=False)

```
>>> a = [3, 2, 1, 4]
>>> a.sort()
[1, 2, 3, 4]
```

정렬하기(내림차순) sort(reverse=True)

```
>>> a = [3, 2, 1, 4]
>>> a.sort(reverse=True)
[4, 3, 2, 1]
```

소스코드

07-1 리스트와 튜플.ipynb

튜플 기능

두 변수 간 값 바꾸기

```
>>> a = 1  
>>> b = 2  
>>> temp = a  
>>> a = b  
>>> b = temp
```

튜플 사용하기

```
>>> a, b = b, a
```

다차원 리스트와 튜플

리스트 = [[값, 값], [값, 값] ...]

```
>>> a = [[1, 2], [3, 4], [5, 6]]
```

튜플 = ((값, 값), (값, 값) ...)

```
>>> a = ((1, 2), (3, 4), (5, 6))
```

접근시 리스트[세로인덱스][가로인덱스]

```
>>> a[0][1]
```

```
2
```

```
>>> a[2][0]
```

```
5
```

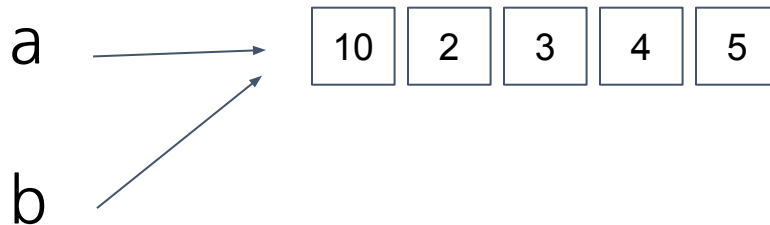
소스코드

07-1 리스트와 튜플.ipynb

리스트 할당과 복사

실제 값들의 복사가 일어나지 않음

```
>>> a = [1, 2, 3, 4, 5]
>>> b = a
>>> a is b
True
```



```
>>> b[0] = 10
>>> print(a)
[10, 2, 3, 4, 5]
```

소스코드

07-1 리스트와 튜플.ipynb

리스트 할당과 복사

copy() 함수로 실제 값들을 복사

```
>>> a = [1, 2, 3, 4, 5]
```

```
>>> b = a.copy()
```

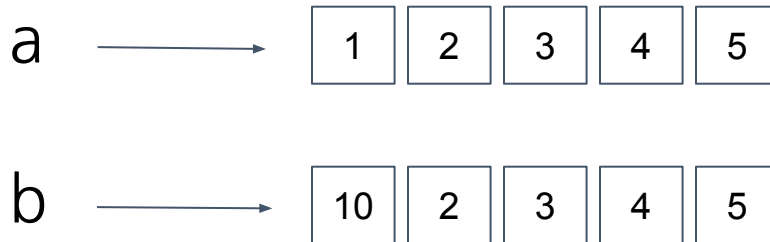
```
>>> a is b
```

```
False
```

```
>>> b[0] = 10
```

```
>>> print(a)
```

```
[1, 2, 3, 4, 5]
```



소스코드

07-1 리스트와 튜플.ipynb

다차원 복사

2차원 이상의 리스트는 `copy.deepcopy()` 를 이용 복사

```
>>> a = [[1, 2], [3, 4], [5, 6]]
>>> b = a.copy()
>>> b[0][0] = 10
>>> a
[[10, 2], [3, 4], [5, 6]]
```

```
>>> import copy
>>> a = [[1, 2], [3, 4], [5, 6]]
>>> b = copy.deepcopy(a) #깊은복사
>>> b[0][0] = 10
>>> print(a)
>>> print(b)
[[1, 2], [3, 4], [5, 6]]
[[10, 2], [3, 4], [5, 6]]
```

소스코드

07-1 리스트와 튜플.ipynb

실습01

range 함수를 이용하여 사용자가 입력한 수까지 2의 배수값을 넣은 리스트를 만들고 리스트의 맨 마지막에 사용자가 입력한 추가해주세요

실행 :

숫자를 입력해주세요 : 10

[2, 4, 6, 8, 10, 10]

숫자를 입력해주세요 : 20

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 20]

소스코드

07-1 실습01.ipynb

8장. 덕셔너리

- 파이썬기초 -

딕셔너리

딕셔너리 = {키1: 값1, 키2: 값2}

score = {"name": "Tom", "math": 80, "english": 70}

```
>>> score = {"name": "tom", "math": 80, "english": 70}
```

```
>>> score["name"]
```

```
tom
```

```
>>> score["name"] = "michale"
```

```
>>> score["name"]
```

```
michale
```

```
>>> type(score)
```

```
dic
```

소스코드

08-1 딕셔너리.ipynb

dict

딕셔너리 = dict(키1=값1, 키2=값2)

score = dict(name="Tom", math=80, english=70)

```
>>> score = dict(name="tom", math=80, english=70)
```

```
>>> score["name"]
```

```
tom
```

```
>>> score = dict() #비어있는 딕셔너리
```

```
>>> score = {} #비어있는 딕셔너리
```

소스코드

08-1 딕셔너리.ipynb

딕셔너리 기능 #1

키가 있는지 확인

```
>>> score = {"name": "tom", "math": 80, "english": 70}
>>> "math" in score
True
>>> "age" in score
False
```

키의개수

```
>>> len(score)
3
```

키-값 쌍 추가하기.setdefault(키, 값)

```
>>> score.setdefault("age", 20)
{"name": "tom", "math": 80, "english": 70, "age": 20}
```

키-값 수정하기 update({키: 값})

```
>>> score.update({"math": 90})
{"name": "tom", "math": 90, "english": 70, "age": 20}
```

소스코드

08-1 딕셔너리.ipynb

딕셔너리 기능 #2

키로 딕셔너리 항목삭제 pop(키,기본값)

```
>>> score = {"name": "tom", "math": 80, "english": 70}
>>> score.pop("name") #삭제된 키의 값 반환
tom
```

```
>>> score
{"math":80, "english":70}
```

```
>>> score.pop("age", 0)
0
```

모든 값 삭제 clear()

```
>>> score.clear()
>>> score
{}
```

모든 키, 값 가져오기

```
>>> score.keys()
dict_keys(["math", "english"])
>>> score.values()
dict_values([80, 70])
>>> score.items()
dict_items([("math",80), ("english",70)])
```

소스코드

08-1 딕셔너리.ipynb

딕셔너리 할당과 복사

딕셔너리 복사 copy()

```
>>> a = {"a": 0, "b": 1}
>>> b = a.copy() #리스트와 마찬가지로
```

중첩 딕셔너리의 경우 deepcopy()

```
>>> import copy
>>> a = {"a": {"c": 0, "d": 0}, "b": {"e": 0, "f": 0}}
>>> b = copy.deepcopy(a)
>>> print(b)
```

```
{"a": {"c": 0, "d": 0}, "b": {"e": 0, "f": 0}}
```

소스코드

08-1 딕셔너리.ipynb

실습01

(1) 이름, 나이, 연락처를 입력받아 딕셔너리를 만들어 출력해주세요

실행 :

이름을 입력해주세요 : 홍길동

나이를 입력해주세요 : 27

연락처를 입력해주세요 : 010-3023-1223

```
{'이름': '홍길동', '나이': '27', '연락처': '010-3023-1223'}
```

(2) 두사람의 이름, 나이, 연락처를 입력받아 각각 딕셔너리를 만들어 리스트에 넣어주세요

실행 :

이름을 입력해주세요 : 홍길동

나이를 입력해주세요 : 27

연락처를 입력해주세요 : 010-3023-1223

이름을 입력해주세요 : 이몽룡

나이를 입력해주세요 : 30

연락처를 입력해주세요 : 010-3030-4434

```
[{'이름': '홍길동', '나이': '27', '연락처': '010-3023-1223'}, {'이름': '이몽룡', '나이': '30', '연락처': '010-3030-4434'}]
```

소스코드

08-1 실습01.ipynb

9장. 세트

- 파이썬기초 -

세트

세트 = {값1, 값2, 값3, 값4}

animal = {"dog", "cat", "monkey", "horse"}

```
>>> animal = {"dog", "cat", "monkey", "horse"}
```

```
>>> type(score)
```

```
<class 'set'>
```

소스코드

09-1 세트.ipynb

세트의 기능

세트에 특정값 확인

```
>>> animal = {"dog", "cat", "monkey", "horse"}  
>>> "cat" in animal  
True
```

set을 사용하여 세트 만들기

```
>>> a = set("animal")  
>>> a  
{“a”, “n”, “i”, “m”, “a”, “l”}
```

```
>>> b = set(range(5))  
>>> b  
{0, 1, 2, 3, 4}
```


집합 연산 #1

합집합 |, set.union

```
>>> a = {1, 2, 3}
>>> b = {3, 4, 5}
>>> a | b
{1, 2, 3, 4, 5}
>>> set.union(a, b)
{1, 2, 3, 4, 5}
```

교집합 &, set.intersection

```
>>> a = {1, 2, 3}
>>> b = {3, 4, 5}
>>> a & b
{3}
>>> set.intersection(a, b)
{3}
```

집합 연산 #2

차집합 -, set.difference

```
>>> a = {1, 2, 3}
>>> b = {3, 4, 5}
>>> a - b
{1, 2}
>>> set.difference(a, b)
{1, 2}
```

대칭차집합 ^, set.symmetric_difference

```
>>> a = {1, 2, 3}
>>> b = {3, 4, 5}
>>> a ^ b
{1, 2, 4, 5}
>>> set.symmetric_difference(a, b)
{1, 2, 4, 5}
```

부분집합, 상위집합 확인

부분집합 \leq , `issubset`(다른세트)

```
>>> a = {1, 2, 3, 4}
>>> a <= {1, 2, 3, 4, 5}
True
>>> a.issubset({1, 2, 3, 4, 5})
True
>>> a <= {1, 2, 3}
False
>>> a.issubset({1, 2, 3})
False
```

상위집합 \geq , `issuperset`(다른세트)

```
>>> a = {1, 2, 3, 4}
>>> a >= {1, 2, 3}
True
>>> a.issuperset({1, 2, 3})
True
>>> a >= {1, 2, 3, 4, 5}
False
>>> a.issuperset({1, 2, 3, 4, 5})
False
```

겹치는 요소확인

isdisjoint(다른세트)

```
>>> a = {1, 2, 3, 4}
```

```
>>> a.isdisjoint({5, 6, 7, 8}) #겹치는 요소 없음
```

```
True
```

```
>>> a.isdisjoint({2, 3, 4, 5}) #2, 3, 4 겹침
```

```
False
```

소스코드

09-1 세트.ipynb

세트 조작하기

추가하기 add(요소)

```
>>> a = {1, 2, 3, 4}
>>> a.add(5)
>>> a
{1, 2, 3, 4, 5}
```

삭제하기 remove(요소), discard(요소)

```
>>> a = {1, 2, 3, 4}
>>> a.remove(1)
>>> a
{2, 3, 4}
>>> a.discard(2)
>>> a
{3, 4}
```

소스코드
09-1 세트.ipynb

10장. 조건문 if

- 파이썬기초 -

if

조건문은 특정 조건일 때 코드를 실행하는 문법

if 조건식: 코드

들여쓰기 or 탭
공식 4칸

```
>>> x = 10
>>> if x == 10:
>>>     print("x 가 10 입니다")
```

소스코드

10-1 조건문 if.ipynb

if

if 문 조건 생략

```
>>> x = 10
>>> if x == 10:
    pass
```

if 문 들여쓰기

```
>>> x = 10
>>> if x == 10:
>>>     print("x 가 10 입니다")
>>>     print("x 가 12 가 아닙니다")
>>>         print("x 가 10 입니다") #error
>>> print("x가 10입니다") #error
>>> print("if문 밖") #if 문과는 상관없음
```

소스코드

10-1 조건문 if.ipynb

if

다양한 조건들

```
>>> if x > 3:
```

```
>>>     print("x 가 3보다 크다")
```

```
>>> if x > 2 and x < 10:
```

```
>>>     print("x가 2보다 크고 10보다 작다")
```

```
>>> if 0 < x < 20:
```

```
>>>     print("x는 0보다 크고 20보다 작다")
```

중첩 if 문

```
>>> if x >= 10:
```

```
>>>     if x <= 20:
```

```
>>>         print("10이상 20이하")
```

```
>>>     elif x <= 30:
```

```
>>>         print("20초과 30이하")
```

소스코드

10-1 조건문 if.ipynb

elif 와 else

if 조건문의 분기를 위한 문법

if 조건식: #조건1

코드 #조건1 True

elif 조건식: #조건2

코드 #조건1 False, 조건2 True

elif 조건식: #조건3

코드 #조건1 False, 조건2 False, 조건3 True

else:

코드 #모든 조건식이 False

```
>>> if x == "A":  
>>>     print("x 는 A")  
>>> elif x == "B":  
>>>     print("x 는 B")  
>>> elif x == "C":  
>>>     print("x 는 C")  
>>> else:  
>>>     print("x 는 A, B, C 가 아님")
```

소스코드

10-1 조건문 if.ipynb

실습01

사용자로 점수를 3개 입력받아
모든 점수가 65점보다 클 경우 합격 아닐경우 불합격을 출력하세요
단, 0~100 이 아닌 숫자가 입력된경우 잘못된 "잘못된 점수가
입력되었습니다" 를 출력하세요

실행 :

첫번째 점수를 입력해주세요 :

120

두번째 점수를 입력해주세요 : 90

세번째 점수를 입력해주세요 : 80

잘못된 점수가 입력되었습니다

첫번째 점수를 입력해주세요 : 80

두번째 점수를 입력해주세요 : 90

세번째 점수를 입력해주세요 : 75

합격

첫번째 점수를 입력해주세요 : 50

두번째 점수를 입력해주세요 : 60

세번째 점수를 입력해주세요 : 90

불합격

소스코드

10-1 실습01.ipynb

실습02

```
fruit = ['사과', '오렌지']
```

```
vegetable = ['당근', '호박']
```

위와 같은 리스트 두개를 만들고 유저로부터 카테고리과 상품명을 입력받아

카테고리가 과일일때는 fruit 리스트에 카테고리가 채소일때는 vegetable 리스트에 상품을 추가하고
리스트의 모든내용을 출력해주세요.

단, 카테고리명이 채소나 과일이 아닐경우 “ 존재하지 않는 카테고리입니다. “

이미 등록되어있는 경우 “이미등록된 과일입니다.” or “이미 등록등록된 채소입니다” 를 출력해주세요.

실행 :

등록할 카테고리를 선택해주세요 (과일, 채소) :채 등록할 카테고리를 선택해주세요 (과일, 채소) :야채

등록할 채소를 입력해주세요 :당근

등록할 야채를 입력해주세요 :당근

이미등록된 채소 입니다.

존재하지 않는 카테고리입니다.

등록할 카테고리를 선택해주세요 (과일, 채소) :과일

등록할 과일을 입력해주세요 :바나나

['사과', '오렌지', '바나나']

소스코드

10-1 실습02.ipynb

11장. 반복문 for

- 파이썬기초 -

for

어떠한 코드를 반복해야할때 사용
지정된 범위만큼 (주로 반복횟수가 정해져있을때 사용)

for 변수 in 값이 여러개인 자료형이나 변수: 코드

들여쓰기 or 탭

공식 4칸

```
>>> for i in range(0,10):  
>>>     print("현재값: ", i)
```

소스코드

11-1 반복문 for.ipynb

for

for문과 range()

```
>>> for i in range(0, 10)
```

```
>>>     print(i)
```

```
>>> for i in range(10, 0, -1)
```

```
>>>     print(i)
```

for문과 다양한 자료형들

```
>>> a = [1, 2, 3, 4, 5, 6, 7]
```

```
>>> for i in a:
```

```
>>>     print(i)
```

```
>>> for i in "Orange":
```

```
>>>     print(i, end=" ")
```

```
>>> a = {"name": "tom", "math": 80, "english": 70}
```

```
>>> for i in a:
```

```
>>>     print(i, end=" ")
```

```
>>>     print(a[i])
```

소스코드

11-1 반복문 for.ipynb

for

입력한 횟수만큼 반복하기

```
>>> count = int(input("반복할횟수?"))
>>> for i in range(count):
>>>     print('hello, world!', i)
```

enumerate 사용하여 index 접근

```
>>> a = [1, 2, 3, 4, 5, 6, 7]
>>> for idx, val in enumerate(a):
>>>     print(idx, val, sep=" ", "
```

for문 중첩

```
>>> for i in range(0, 5)
>>>     print("*****", i)
>>>     for j in range(0, 5)
>>>         print("j : ", j)
```

소스코드

11-1 반복문 for.ipynb

실습01

(1) $x = [3, 6, 9, 20, -7, 5]$ 의 값의 모든 요소에 10을 곱하여 저장한뒤 출력하세요

실행 :

[30, 60, 90, 200, -70, 50]

(2) $y = \{\text{"math": 70, "science": 80, "english": 20}\}$ 의 값의 모든 요소에 10을 더하여 저장한뒤 출력하세요

실행 :

{'math': 80, 'science': 90, 'english': 30}

(3) 숫자를 입력받고 입력받은 정수의 구구단을 출력하세요

실행 :

몇단을 출력하시겠습니까? 3

3 * 1 = 3

3 * 2 = 6

...

3 * 8 = 24

소스코드

11-1 실습01.ipynb

실습02

(1) words = ["school", "game", "piano", "science", "hotel", "mountain"] 중 글자수가 6글자 이상인 문자를 모아 새로운 리스트를 생성하세요

실행 :

```
['school', 'science', 'mountain']
```

(2) 구구단을 1단부터 9단까지 출력하세요

실행 :

```
1 * 1 = 1
```

```
1 * 2 = 2
```

```
1 * 3 = 3
```

```
....
```

```
9 * 9 = 81
```

소스코드

11-1 실습02.ipynb

실습03

(1) [3, 6, 9, 20, -7, 5] 리스트를 sort 와같은 함수를 사용하지말고 for문을 활용하여 오름차순으로 정렬해주세요.

실행 :

[-7, 3, 5, 6, 9, 20]

(2) 1-100 까지 숫자중 3과 5의 공배수일경우 “3과 5의 공배수”

나머지 숫자중 3의배수일경우 “3의배수” 나머지 숫자중 5의배수일경우 “5의배수”

모두 해당되지 않을경우 그냥숫자 를 출력하세요

실행 :

1

2

3의배수

...

14

3과 5의 공배수

16

...

12장. 반복문 while

- 파이썬기초 -

while

어떠한 코드를 반복해야할때 사용
조건에 따라 반복 (주로 반복횟수가 정해져있지 않을때)

while 조건식: 코드

들여쓰기 or 탭
공식 4칸

```
>>> i = 0
>>> while i < 10:
>>>     print("현재값 : ", i)
>>>     i += 1
```

소스코드

12-1 반복문 while.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

while 응용

입력한 횟수만큼 반복하기

```
>>> count = int(input("반복횟수?"))
>>> i = 0
>>> while i < count:
>>>     print("입력한 횟수만큼 반복")
>>>     i += 1
```

입력조건이 맞을때까지 반복하기

```
>>> i = 0
>>> while i != 5:
>>>     i = int(input("5를 입력하면 반복이 중단됩니다."))
>>> print("종단!")
```

소스코드

12-1 반복문 while.ipynb

실습01

사용자로부터 숫자를 계속 입력받다가
0을 입력하면 합계를 출력해주세요

실행 :

값을 입력해주세요 : 30

값을 입력해주세요 : 20

값을 입력해주세요 : 50

값을 입력해주세요 : 40

값을 입력해주세요 : 30

값을 입력해주세요 : 0

합계는 ? 170

소스코드

12-1 실습01.ipynb

실습02 (가위바위보 게임만들기)

random 모듈 사용방법 - 랜덤값호출

```
>>> import random #random 모듈을 가져온다
>>> random.random()
0.00202302032
>>> random.randint(1,3)
3
```

가위(1), 바위(2), 보(3) 을 입력해주세요 : 3

유저 : 보, 컴퓨터 : 보

가위(1), 바위(2), 보(3) 을 입력해주세요 : 2

유저 : 바위, 컴퓨터 : 보

가위(1), 바위(2), 보(3) 을 입력해주세요 : 1

유저 : 가위, 컴퓨터 : 보

가위(1), 바위(2), 보(3) 을 입력해주세요 : 4

게임종료 (전체:3 ,승리:1)

**1~3 을 입력하면 게임진행 이외의 숫자를 입력하면
게임종료**

소스코드

12-1 실습02.ipynb

15장. break, continue

- 파이썬기초 -

break, continue

for, while 에서 제어흐름을 벗어나기 위해사용

break

for, while 을 완전히 중단

continue

이번 반복만 중단하고
처음으로 돌아가 다음반복 수행

소스코드

15-1 break, continue.ipynb

break, continue

for 문에서의 예제

break

```
>>> for i in range(5):  
>>>     if(i == 3):  
>>>         break  
>>>     print(i, end=" ")
```

결과

012

continue

```
>>> for i in range(5):  
>>>     if(i == 3):  
>>>         continue  
>>>     print(i, end=" ")
```

결과

0124

소스코드

15-1 break, continue.ipynb

break, continue

while 문에서의 예제

break

```
>>> i = 0
>>> while i < 30:
>>>     if i == 20 :
>>>         break
>>>     print(i, end=" ")
>>>     i += 1
```

결과

012.....19

continue

```
>>> i = 0
>>> while i < 30:
>>>     i += 1
>>>     if i % 2 == 0:
>>>         continue
>>>     print(i, end=" ")
```

결과

1 3 5 7 29

소스코드

15-1 break, continue.ipynb

실습01

(1) 아래 예시코드의 while 문을 완성하여 사용자가 입력한 숫자만큼 출력해주세요. (break 사용)

```
user = int(input("숫자를 입력하세요:"))
cnt = 0
while True:
    (여기 코드를 완성해 주세요)
```

실행 :
 숫자를 입력하세요: 20
 0
 1
 ...
 19
 20

(2) 아래 예시코드의 for 문을 완성하여 사용하여 사용자가 입력한 숫자까지의 짝수를 출력하기. (continue 사용)

```
user = int(input("숫자를 입력하세요:"))
for i in range(user+1):
    (여기 코드를 완성해 주세요)
    print(i)
```

실행 :
 숫자를 입력하세요: 20
 0
 2

 14
 16
 18
 20

소스코드

15-1 실습01.ipynb

13장. 리스트 응용

- 파이썬기초 -

리스트의 가장큰수 가장작은수 구하기

max, min

```
>>> a = [32, 45, 2, 5, 76]
```

```
>>> small = a[0]
```

```
>>> for i in a:
```

```
>>>     if i < small:
```

```
>>>         small = i
```

```
>>> large = a[0]
```

```
>>> for i in a:
```

```
>>>     if i > large:
```

```
>>>         large = i
```

```
>>> a = [32, 45, 2, 5, 76]
```

```
>>> a.sort()
```

```
>>> a[0]
```

```
>>> a.sort(reverse=True)
```

```
>>> a[0]
```

```
>>> a = [32, 45, 2, 5, 76]
```

```
>>> min(a)
```

```
>>> max(a)
```

소스코드

13-1 리스트 응용.ipynb

합계구하기

sum

```
>>> a = [32, 45, 2, 5, 76]
```

```
>>> b = 0
```

```
>>> for i in a:
```

```
>>>     b += i
```

```
>>> a = [32, 45, 2, 5, 76]
```

```
>>> sum(a)
```

소스코드

13-1 리스트 응용.ipynb

split, join 함수

split 함수

문자를 리스트로

```
>>> fruit = "사과,배,옥수수,당근"  
>>> fruit_list = fruit.split(",")  
>>> print(fruit_list)
```

`['사과', '배', '옥수수', '당근']`

join 함수

리스트를 문자로

```
>>> fruit_list = ['사과', '배', '옥수수', '당근']  
>>> fruit = "".join(fruit_list)  
>>> print(fruit)
```

`사과배옥수수당근`

```
>>> fruit = " ".join(fruit_list)  
>>> print(fruit)
```

`사과 배 옥수수 당근`

```
>>> fruit = ", ".join(fruit_list)  
>>> print(fruit)
```

`사과,배,옥수수,당근`

소스코드

13-1 리스트 응용.ipynb

리스트 컴프리헨션(list comprehension)

list[식 for 변수 in 리스트]

```
>>> a = [i for i in range(10)]
```


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

```
>>> a = [i + 5 for i in range(10)]
```

```
[5, 6, 7, 8, 9, 10, 11, 12, 13, 14]
```

```
>>> a = [i * 3 for i in range(10)]
```

```
[0, 3, 6, 9, 12, 15, 18, 21, 24, 27]
```


c = [i + 5 for i in range(10)]

소스코드

13-1 리스트 응용.ipynb

리스트 컴프리헨션(list comprehension)

list[식 for 변수 in 리스트 if 조건]

```
>>> a = [i for i in range(10) if i % 2 == 0]
```

```
[0, 2, 4, 6, 8]
```

```
>>> a = [i for i in range(10) if i % 2 == 1]
```

```
[1, 3, 5, 7, 9]
```

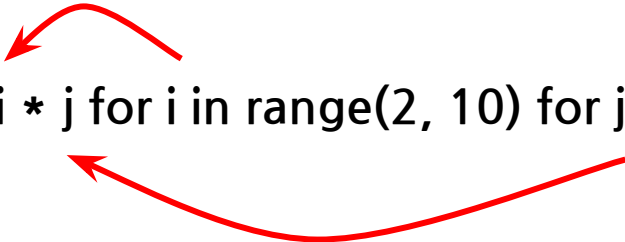

c = [i for i in range(10) if i % 2 == 0]

소스코드

13-1 리스트 응용.ipynb

리스트 컴프리헨션(list comprehension)

list[식 for 변수 in 리스트 for 변수 in 리스트]



```
c = [i * j for i in range(2, 10) for j in range(1, 10)]
```

```
>>> a = [i * j for i in range(2, 10) for j in range(1, 10)]
```

```
>>> a
```

```
[2, 4, 6, 8, 10 ..... 81]
```

소스코드

13-1 리스트 응용.ipynb

리스트 컴프리헨션(list comprehension)

리스트를 딕셔너리로 변경

```
>>> keys = ["name", "age", "address"]
>>> users = ["tom", 20, "incheon"]
>>> dicdic = { keys[i] : users[i] for i in range(0,3) }
>>> dicdic
{'name': 'tom', 'age': 20, 'address': 'incheon'}
```

소스코드

13-1 리스트 응용.ipynb

ZIP

zip(리스트1, 리스트2)



```
>>> keys = ["name", "age", "address"]
>>> users = ["tom", 20, "incheon"]
>>> dic = dict(zip(keys, users))
>>> print(dic)
{'name': 'tom', 'age': 20, 'address': 'incheon'}
>>> lis = list(zip(keys, users))
>>> print(lis)
[('name', 'tom'), ('age', 20), ('address', 'incheon')]
```

소스코드

13-1 리스트 응용.ipynb

실습01

(1) word = ["school", "game", "piano", "science", "hotel", "mountain"] 중 글자수가 6글자 이상인 문자를 모아 새로운 리스트를 생성하세요

(리스트 컴프리헨션을 사용해주세요)

실행 :

```
['school', 'science', 'mountain']
```

(2) word = ["school", "game", "piano", "science", "hotel", "mountain"] 리스트의 글자수가 들어가는 새로운 리스트를 생성하세요

(리스트 컴프리헨션을 사용해주세요)

실행 :

```
[6, 4, 5, 7, 5, 8]
```

소스코드

13-1 실습01.ipynb

2차원리스트(다차원리스트)

2차원 리스트 선언

```
>>> a = [[10, 20], [30, 40], [50, 60]]
>>> a = [[10, 20],
          [30, 40],
          [50, 60]]
>>> a[0][0]
10
>>> a[0][1]
20
>>> a[1][1]
40
```

2차원 리스트 값추가

```
>>> a = [[10, 20], [30, 40], [50, 60]]
>>> a[0].append(10)
[[10,20,10],[30,40],[50,60]]
>>> a[1].append(20)
[[10,20,10],[30,40,20],[50,60]]
>>> a[2].extend([1,2])
[[10,20,10],[30,40,20],[50,60,1,2]]
```

소스코드

13-1 리스트 응용.ipynb

2차원 리스트 값 출력 (다차원리스트)

for 문 사용

```
>>> a = [[10, 20], [30, 40], [50, 60]]
>>> for x, y in a:
>>>     print(x, y)
10 20
30 40
50 60
```

이중 for 문 사용

```
>>> a = [[10, 20], [30, 40], [50, 60]]
>>> for i in a:
>>>     for j in i:
>>>         print(j, end=' ')
>>> print()
10 20
30 40
50 60
```

소스코드

13-1 리스트 응용.ipynb

2차원 리스트 값 접근 (다차원리스트)

for 와 range 사용

```
>>> a = [[10, 20], [30, 40], [50, 60]]
>>> for i in range(len(a)):
>>>     for j in range(len(a[i])):
>>>         print(a[i][j], end=" ")
>>>     print()
10 20
30 40
50 60
```

for 와 enumerate 사용

```
>>> a = [[10, 20], [30, 40], [50, 60]]
>>> for idx, val in enumerate(a):
>>>     for idx2, val2 in enumerate(val):
>>>         print(idx, idx2, val2)
0 0 10
0 1 20
1 0 30
1 1 40
2 0 50
2 1 60
```

소스코드

13-1 리스트 응용.ipynb

2차원 리스트 만들기

```
>>> a = []
>>> for i in range(3):
>>>     temp = []
>>>     for j in range(2):
>>>         temp.append(0)
>>>     a.append(temp)
>>> print(a)
[[0, 0],[0, 0],[0, 0]]
```

소스코드

13-1 리스트 응용.ipynb

실습02

아래 두 리스트를 곱해 새로운 리스트 C 를 만드세요

a = [[10, 20], [30, 40], [50, 60]]

b = [[2, 3], [4, 5], [6, 7]]

실행 :

[[20, 60], [120, 200], [300, 420]]

실습03

아래의 학습코드를 가지고 a 리스트가 [[1,2],[3,4],[5,6]] 와 같이 만들어지도록 수정하세요

```
>>> a = []  
>>> for i in range(3):  
>>>     temp = []  
>>>     for j in range(2):  
>>>         temp.append(0)  
>>>     a.append(temp)  
>>> print(a)
```

[[1,2],[3,4],[5,6]]

소스코드

13-1 실습03.ipynb

14장. 별 출력(조건, 반복문 익숙해지기)

- 파이썬기초 -

별출력하기

사각형별

* * * * *

* * * * *

* * * * *

* * * * *

* * * * *

계단식별

*

* *

* * *

* * * * *

* * * * *

대각선별

*

*

*

*

*

계단식(역순)

* * * * *

* * * *

* * *

* *

*

계단식(역순)

* * * * *

* * * *

* * *

* *

*

소스코드

14-1 별출력.ipynb

16장. 파일 입출력

- 파이썬기초 -

파일 입출력

파일쓰기

```
file = open("file.txt", "w") # 파일을 쓰기모드(w)  
file.write("First File") # 문자열 저장  
file.close() # 파일객체닫기
```

파일읽기

```
file = open("file.txt", "r") # 파일을 읽기모드(r)  
text = file.read() # 파일에서 내용 읽기  
print(text)  
file.close() # 파일객체닫기
```

소스코드

16-1 파일 입출력.ipynb

with

자동으로 파일객체 닫기

```
with open("file.txt", "r") as file:  
    text = file.read()  
    print(text)
```

소스코드

16-1 파일 입출력.ipynb

pickle 모듈을 이용한 리스트 파일에 저장하기

```
import pickle
```

```
text = ["First File", "Second Line"]
```

```
with open("data.pkl", "wb") as file: #data 파일을 바이너리 쓰기모드로 열기  
    pickle.dump(text, file)
```

소스코드

16-1 파일 입출력.ipynb

pickle 모듈을 이용한 리스트 파일에서 불러오기

```
import pickle
```

```
with open("data.pkl", "rb") as file: #data 파일을 바이너리 읽기모드로 열기  
    data = pickle.load(file)  
    print(data)
```

```
["First File", "Second Line"]
```

소스코드

16-1 파일 입출력.ipynb

pickle 모듈을 이용한 다양한자료형 저장하기

```
import pickle
```

```
name = "tom"
```

```
age = 24
```

```
address = "서울시 마포구"
```

```
scores = {"python": 90, "deeplearning": 95, "database": 85}
```

```
with open("data2.pkl", "wb") as file:
```

```
    pickle.dump(name, file)
```

```
    pickle.dump(age, file)
```

```
    pickle.dump(address, file)
```

```
    pickle.dump(scores, file)
```

소스코드

16-1 파일 입출력.ipynb

pickle 모듈을 이용한 다양한자료형 불러오기

```
import pickle
```

```
with open("data2.pkl", "rb") as file:
```

```
    name2 = pickle.load(file)
```

```
    age2 = pickle.load(file)
```

```
    address2 = pickle.load(file)
```

```
    scores2 = pickle.load(file)
```

소스코드

16-1 파일 입출력.ipynb

실습01

가위바위보 게임 업그레이드

이전에 만든 가위바위보 게임을 총 게임횟수와 승리횟수를 게임을 다시 실행해도 유지되도록 수정하세요

소스코드

16-1 실습01.ipynb

실습02

성적관리 프로그램 개발

메뉴를 선택해주세요 1 - 입력, 2 - 조회, 3 - 삭제, 0 - 종료):

1

이름 : 이진범

수학 : 30

과학 : 40

영어 : 50

메뉴를 선택해주세요 1 - 입력, 2 - 조회, 3 - 삭제, 0 - 종료):

2

[0] 이름 : 이민호, 수학 : 40, 과학 : 50, 영어 : 30

[1] 이름 : 이진범, 수학 : 30, 과학 : 40, 영어 : 50

메뉴를 선택해주세요 1 - 입력, 2 - 조회, 3 - 삭제, 0 - 종료):

3

[0] 이름 : 이민호, 수학 : 40, 과학 : 50, 영어 : 30

[1] 이름 : 이진범, 수학 : 30, 과학 : 40, 영어 : 50

삭제할 번호를 입력해주세요 : 0

삭제가 완료되었습니다.

메뉴를 선택해주세요 1 - 입력, 2 - 조회, 3 - 삭제, 0 - 종료):

0

종료되었습니다

다음과 같이 동작하는 프로그램을 개발하세요

**프로그램 종료후 다시 실행할때 이전에 입력됐던 값을
파일에 저장해놔다 불러와주세요**

소스코드

16-1 실습02.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

17장 함수

- 파이썬기초 -

함수

특정한 기능을 반복해서 사용해야할때



함수이름 인자, 매개변수

```
def func(a, b):  
    print("함수입니다.")  
    return a + b
```

코드블럭

반환값

```
func_test = func(1, 2)  
print(func_test)
```

소스코드

17-1 함수.ipynb

이미지 출처

<https://terms.naver.com/entry.nhn?docId=2039077&cid=47308&categoryId=47308>

가치를 높이는 금융 인공지능 실무교육

Insight campus

함수

인자와 반환값이 없는 함수

```
>>> def func():  
>>>     print("Hello, Function")  
>>> func()
```

인자는 있으나 반환값이 없는 함수

```
>>> def func(name):  
>>>     print("Hello, " + name)  
>>> func("function")
```

인자와 반환값이 있는 함수

```
>>> def func(a, b):  
>>>     return a + b  
>>> y = func(1, 2)
```

값을 여러개 반환

```
>>> def func(a, b):  
>>>     return a + b, a - b  
>>> y = func(1, 2)
```

함수

반환값(return)은 함수의 종료를 위해 사용 되기도 한다

```
>>>def id_check(id)
>>> if id == "admin":
>>>     print("invalid id: admin")
>>>     return print
>>> print("valid id: " , id)
```

여러 인자를 반환(튜플 사용)

<pre>>>> def func(a): >>> b = a + 1 >>> return a, b >>> print(func(10))</pre>	<pre>>>> def func(a): >>> b = a + 1 >>> return a, b >>> a, b = func(10)</pre>
---	---

변수의 유효범위

```
a = 5 # 전역변수
```

```
def func1():  
    a = 1 # func1 에서만 사용  
    print(a)
```

```
func1() # 1 출력  
print(a) # 5 출력
```

```
a = 5 # 전역변수
```

```
def func2():  
    print(a) # 전역변수 사용  
  
func2() # 5 출력
```

```
a = 5 # 전역변수
```

```
def func3():  
    global a # 전역변수 사용  
    a = 1 # 전역변수 변경  
    print(a)
```

```
func3() # 1 출력  
print(a) # 1 출력
```

실습01

리스트와 찾고싶은 값을 모두 입력하면
입력한 값의 인덱스(위치) 를 리턴해주는 함수만들기

실행 :

```
lis = [1, 2, 3, 1, 4, 2, 1]
```

```
allindex(lis, 1)
```

```
[0, 3, 6]
```

소스코드

17-1 실습01.ipynb

함수응용 #1

인자값에 리스트 사용 (언패킹)

```
>>> def func(a, b, c):
>>>     print(a, b, c)
>>> x = [1, 2, 3]
>>> func(*x)
123
```

가변인수

```
>>> def func(*args):
>>>     for arg in args:
>>>         print(arg)
>>> func(1)
>>> func(1, 2, 3, 4, 5)
```

가변인수와 고정인수 같이사용

```
>>> def func(a, *args):
>>>     print(a, end="")
>>>     for arg in args:
>>>         print(arg, end="")
>>> func(100, 1, 2, 3, 4, 5)
100 1 2 3 4 5
```

함수응용 #2

키워드 인수 & 딕셔너리 언패킹

```
>>> def func(email, name):  
>>>     print("이메일 : ", email)  
>>>     print("이름 : ", name)  
  
>>> func(email = "aa@aa.com", name = "tom")  
>>> x = {"email": "aa@aa.com", "name": "tom"}  
>>> func(**x)
```

가변 키워드인수

```
>>> def func(**kwargs):  
>>>     print("이메일 : ", kwargs["email"])  
>>>     print("이름 : ", kwargs["name"])  
  
>>> func(email = "aa@aa.com", name="tom")
```


함수응용 #3

매개변수 초기값

```
>>> def func(email, name, age=20):
```

```
>>>     print("이메일 : ", email)
```

```
>>>     print("이름 : ", name)
```

```
>>>     print("나이 : ", age)
```

```
>>> func(email = "aa.aa.com", name = "tom")
```

```
>>> func(email = "aa.aa.com", name = "tom", age=18)
```

실습02

가변인수와 고정인수를 사용해 모든값을 더하거나 곱하는 함수 작성

실행 :

```
calc("+", 1, 2, 3, 4, 5)
```

15

```
calc("*", 1, 2, 3, 4, 5)
```

120

소스코드

17-1 실습02.ipynb

재귀함수

함수안에서 자기자신을 호출하는 방식

```
def test():  
    print('재귀함수!')  
    test()
```

RecursionError: maximum recursion depth exceeded while calling a Python object

재귀함수! 가 계속 출력되다가 재귀의 깊이가 일정길이를 초과하면 오류

```
test()
```

재귀함수

재귀함수 종료조건

```
def test(end):  
    if end == 0:  
        return  
    print('재귀함수!')  
    end -= 1  
    test(end)
```

test(5)



test(4)

test(3)

test(2)

test(1)

test(0)

재귀함수

재귀호출로 팩토리얼 구하기

```
def factorial(n):
    if n == 1:
        return 1
```

```
    return n * factorial(n-1)
```

5 * factorial(4)
4 * factorial(3)
3 * factorial(2)
2 * factorial(1)
factorial(1) return 1

factorial(5)

5 * factorial(4)
4 * factorial(3)
3 * factorial(2)
2 * factorial(1)
factorial(1) return 1

↑
1

5 * factorial(4)
4 * factorial(3)
3 * factorial(2)
2 * factorial(1)

↑
2

5 * factorial(4)
4 * factorial(3)
3 * factorial(2)

↑
6

5 * factorial(4)
4 * factorial(3)

↑
24

5 * factorial(4) **120**

실습03

1. 양의정수 n 을 인자로받아, 1부터 n 까지 합을구하는 재귀함수 구현

실행 :
f_sum(5)

15

2. 재귀함수를 이용 숫자를 입력받아 가장높은 자리수부터 출력하세요

실행 :
f_number(1234)

1
2
3
4

람다표현식 (lambda expression)

람다표현식은 익명함수를 만드는 방법
함수를 간단하게 작성할 수 있어 다른함수의 인수로 넣을때 주로 사용

lambda 매개변수들: 식

```
def plus_ten(x):  
    return x + 10
```

```
>> plus_ten(1)
```

11

```
lambda x: x+10
```

```
plus_ten = lambda x: x + 10
```

```
>> plus_ten(1)
```

11

람다표현식 (lambda expression)

람다표현식 바로 호출하기

```
>> (lambda x: x + 10)(1)
```

11

람다표현식 내에 변수는 사용할수 없지만 바깥에 있는 변수는 사용가능

```
>> y = 20
```

```
>> (lambda x: x+y)(1)
```

21

람다표현식 (lambda expression)

map - 반복되는 자료형의 값들을 함수를 이용 값을가공

일반함수사용

```
a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
def f(x):
    if x % 2 == 0:
        return 0
    else:
        return x
```

```
>> list(map(f, a))
```

```
[1, 0, 3, 0, 5, 0, 7, 0, 9, 0]
```

람다표현식사용

```
a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
>> list(map(lambda x: 0 if x % 2 == 0 else x, a))
[1, 0, 3, 0, 5, 0, 7, 0, 9, 0]
```

소스코드

17-1 함수.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

람다표현식 (lambda expression)

람다표현식에서는 elif 를 사용할수 없다.

lambda 매개변수들: **결과1** if **조건식1** else **결과2** if **조건식2** else **결과3**

```
>> a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
>> list(map(lambda x: 0 if x % 2 == 0 else 1 if x % 3 == 0 else 2, a))
```

```
[2, 0, 1, 0, 2, 0, 2, 0, 1, 0]
```

람다표현식 (lambda expression)

filter - 반복되는 자료형의 값들을 함수를 이용 참인것만 걸러낸다

filter

```
>> a = [2, 6, 4, 3, 6, 8, 3, 9, 6]
>> list(filter(lambda x: x > 2 and x < 8, a))
[6, 4, 3, 6, 3, 6]
```

실습04

1. `numbers = [12, 32, 55, 12, 32, 4, 86, 50]` 리스트에서 60보다 크면 합격 50~60점까지는 대기, 50보다 작으면 불합격이 들어간 리스트를 만드세요.
(람다와 `map` 이용)
2. 파일명이 들어가있는 다음 리스트에서 `files = ["memo.txt", "1.jpg", "32.png", "23.jpg", "223.jpg"]` 리스트에서 확장자가 `jpg` 파일만 골라내 리스트를 만드세요.
(람다와 `filter` 이용)
`find("문자열")` 함수사용 : 해당 문자열이 있을 경우 문자열의 인덱스, 없을 경우 -1 을 반환하는 함수

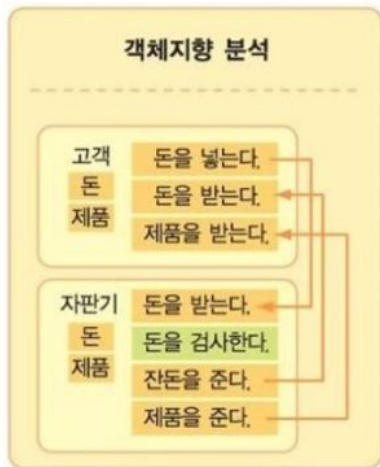
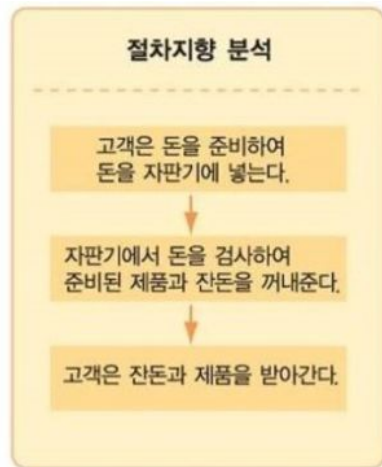
소스코드

17-1 실습04.ipynb

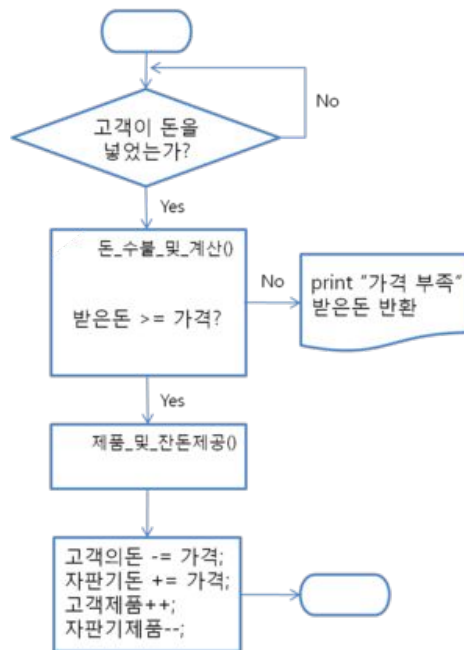
18장 객체와 클래스

- 파이썬기초 -

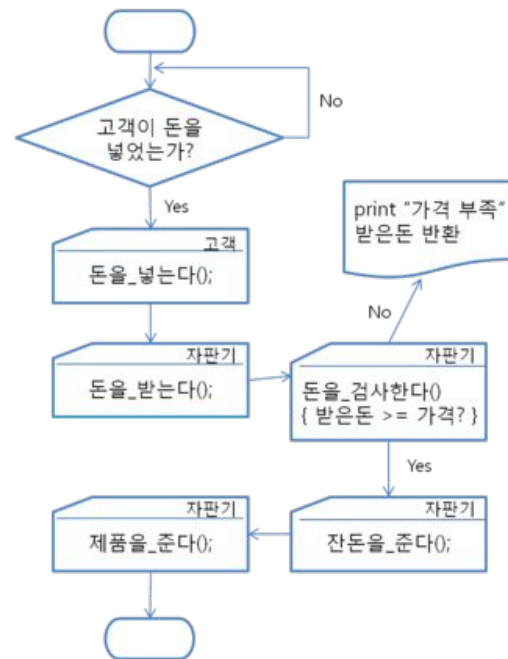
절차 지향 vs 객체 지향



절차지향 방식

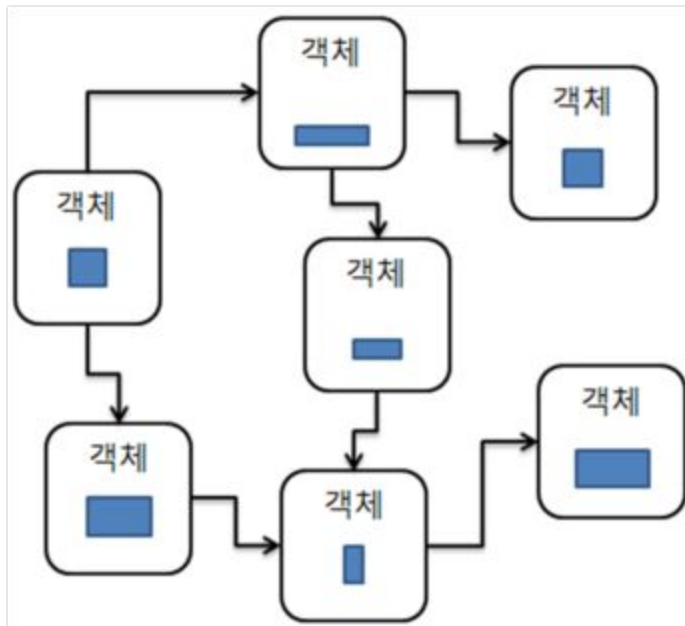
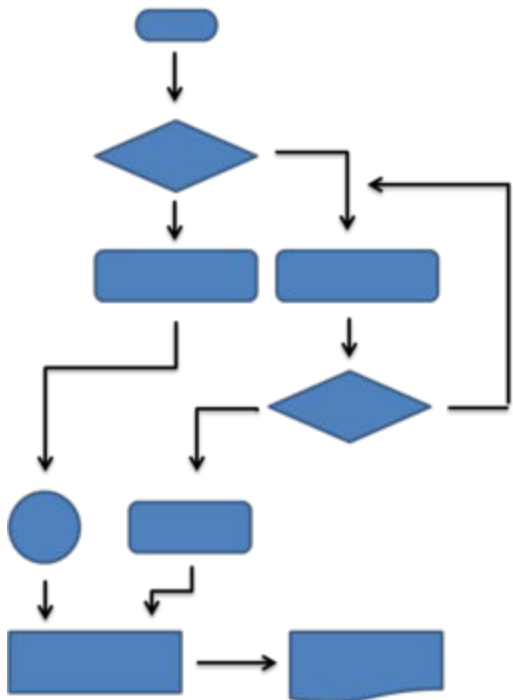


객체지향 방식



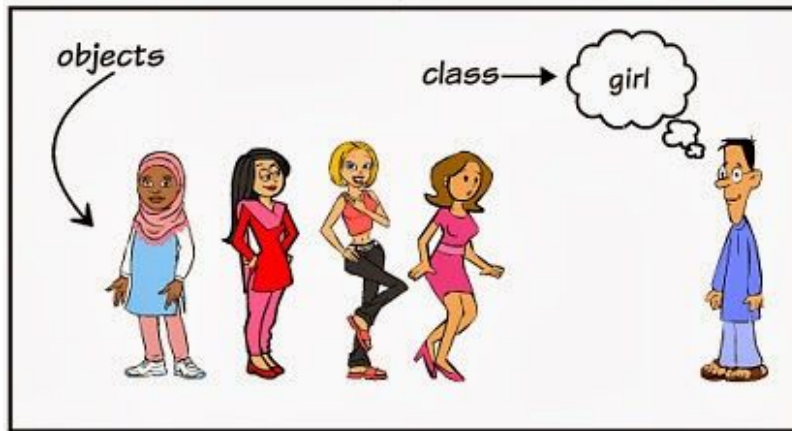
<https://gbsb.tistory.com/3>

절차 지향 vs 객체 지향



<https://gbsb.tistory.com/3>

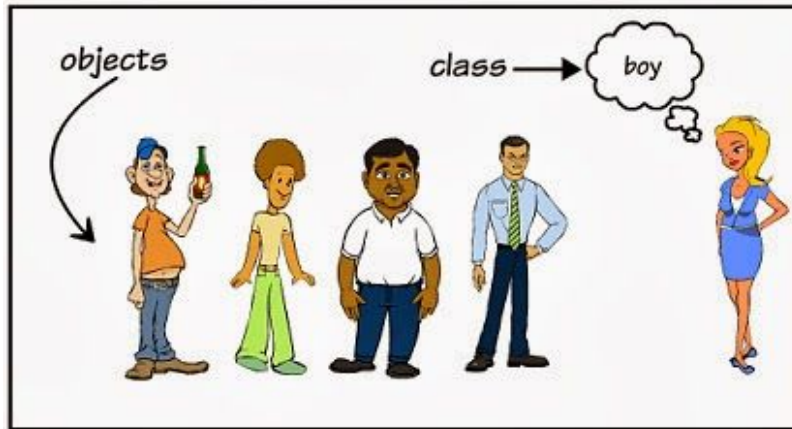
객체와 클래스



객체 : 행동을 하는 실제

클래스 : 객체의 정의 (속성과 기능)

클래스는 정하기 나름 : girl, boy -> human



객체와 클래스

객체 : 속성(변수)과 행동(함수)으로 구성된 대상

클래스 : 객체를 만들기 위한 도구(문법)

클래스이름

```
class Dog:
    def __init__(self, name, color):
        self.hungry = 0 # 인스턴스 속성
        self.name = name
        self.color = color
    def eat(self): # 인스턴스 메서드
        self.hungry -= 10
        print("밥먹음 ", self.hungry)
    def walk(self):
        self.hungry += 10
        print("산책 ", self.hungry)
```

생성자함수

```
choco = Dog("choco", "black") # 객체생성
jjong = Dog("jjong", "white")

choco.eat()
choco.eat()
choco.walk()
print(choco.hungry)
print(jjong.hungry)
```

소스코드

18-1 객체와 클래스.ipynb

가치를 높이는 금융 인공지능 실무교육

Insight campus

비공개 속성 (private attribute)

class Dog:

```
def __init__(self, name, color):
```

```
    self.name = name
```

```
    self.color = color
```

```
    self.__hungry = 0 # 비공개 속성
```

```
def eat(self):
```

```
    if self.__hungry <= 0:
```

```
        print("배가너무 불러요!")
```

```
    else:
```

```
        self.__hungry -= 10
```

```
        print("밥먹음 ", self.__hungry)
```

```
def walk(self):
```

```
    self.__hungry += 10
```

```
    print("산책 ", self.__hungry)
```

```
def condition(self):
```

```
    print("%s 배고픔 : %d", (self.name, self.__hungry))
```

외부에서 속성에 접근하지 못하게 차단
속성명 앞에 __ 추가

```
mery = Dog("mery", "black")
```

```
mery.eat()
```

```
mery.walk()
```

```
mery.walk()
```

```
mery.contition()
```

```
mery.__hungry += 100 # 오류
```

소스코드

18-1 객체와 클래스.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

클래스 속성 (class attribute)

모든 객체(인스턴스)가 공유하는 속성
객체없이 클래스명으로 접근가능

```
class Dog:
    dog_count = 0 # 클래스 속성

    def __init__(self, name, color):
        self.name = name # 인스턴스 속성
        self.color = color
        Dog.dog_count += 1 # 클래스 속성 접근

    def dogCount(self):
        print("총 강아지는 :", Dog.dog_count)
```

```
hello = Dog("hello", "black")
hello.dogCount()
happy = Dog("happy", "black")
happy.dogCount()
```

소스코드

18-1 객체와 클래스.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

정적 메서드 (static method, class method)

객체없이 클래스명으로 접근가능
인스턴스 속성, 인스턴스 메서드 접근불가

```
class Calc:
```

```
    @staticmethod
```

```
    def add(a, b):  
        print(a + b)
```

```
class Calc:
```

```
    @classmethod
```

```
    def add(cls, a, b):  
        print(a + b)
```

```
ex)
```

```
Calc.add(10, 20)
```

```
30
```

```
Calc.add(30, 40)
```

```
70
```

실습01

Car 클래스를 만드세요

- 객체 생성시 차이름, 배기량, 생산년도 입력받고 인스턴스 속성으로 만들어주세요
- 차이름, 배기량, 생산년도는 직접 변경하지 못합니다
- 차이름을 확인하는 함수와 변경하는 함수를 만드세요
- 배기량에 따라 1000CC 보다 작으면 소형
1000CC 이상 2000CC 이하 중형
2000CC 보다크면 대형을 출력하는 인스턴스함수를 만드세요
- 객체 생성시마다 등록된 차량 갯수를 기록하는 클래스속성을 만들어주세요
- 총 등록된 차량개수를 출력하는 클래스 함수를 만드세요

상속

공통되는 내용은 부모 클래스로 만들고
클래스(자식)를 만들때 공통되는 내용을 부모클래스로부터 상속

```
class 부모클래스:
```

```
    코드
```

```
class 자식클래스(부모클래스명):
```

```
    코드
```

소스코드

18-1 객체와 클래스.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

상속

부모클래스

```
class Animal:
    def __init__(self):
        self.hungry = 0
    def eat(self):
        self.hungry -= 10
        print("밥먹음 ", self.hungry)
    def walk(self):
        self.hungry += 10
        print("산책 ", self.hungry)
```

자식클래스

```
class Dog(Animal):
    def __init__(self):
        super().__init__()
    def sound(self):
        print("멍멍")

class Cat(Animal):
    def __init__(self):
        super().__init__()
    def sound(self):
        print("야옹")
```

```
>>> dog = Dog()
>>> dog.sound()
멍멍
>>> dog.walk()
산책 10
>>> dog.walk()
산책 20
>>> cat = Cat()
>>> cat.sound()
야옹
>>> cat.walk()
산책 10
```

소스코드

18-1 객체와 클래스.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

오버라이딩

부모의 기능을 물려받고 일부기능수정

부모클래스

```
class Animal:
    def __init__(self):
        self.hungry = 0
    def eat(self):
        self.hungry -= 10
        print("밥먹음 ", self.hungry)
    def walk(self):
        self.hungry += 10
        print("산책 ", self.hungry)
```

자식클래스

```
class Dog(Animal):
    def __init__(self):
        super().__init__()
    def sound(self):
        print("멍멍")
    def eat(self):
        super().eat()
        print("왈왈")
```

```
>>> dog = Dog()
```

```
>>> dog.eat()
```

밥먹음 -10

왈왈

소스코드

18-1 객체와 클래스.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

추상클래스

상속받는 클래스의 메서드 구현을 강제

부모클래스

```
from abc import *
```

```
class Animal(metaclass=ABCMeta):
```

```
    def __init__(self):
```

```
        self.hungry = 0
```

```
    @abstractmethod
```

```
    def sound(self):
```

```
        pass
```

```
    self.hungry -= 10
```

```
    print("밥먹음 ", self.hungry)
```

```
    def walk(self):
```

```
        self.hungry += 10
```

```
        print("산책 ", self.hungry)
```

자식클래스

```
class Dog(Animal):
```

```
    def __init__(self):
```

```
        super().__init__()
```

```
    def sound(self):
```

```
        print("멍멍")
```

```
    def eat(self):
```

```
        super().eat()
```

```
        print("왈왈")
```

소스코드

18-1 객체와 클래스.ipynb

가치를 높이는 금융 인공지능 실무교육

Insightcampus

static method, class method 차이점2

```
class Animal:
```

```
    type = "동물"
```

```
    @staticmethod
```

```
    def getType1():
        return Animal.type
```

```
    @classmethod
```

```
    def getType2(cls):
        return cls.type
```

```
    def __init__(self):
        self.hungry = 0
```

```
class Dog(Animal):
```

```
    type = "강아지"
```

```
    def __init__(self):
        super().__init__()
    def sound(self):
        print("멍멍")
```

```
ex)
```

```
>>> Dog.getType1()
```

```
동물
```

```
>>> Dog.getType2()
```

```
강아지
```

소스코드

18-1 객체와 클래스.ipynb

가치를 높이는 금융 인공지능 실무교육

Insight campus

실습02

1. Character 클래스를 만들어 주세요

Character 클래스는 health 속성을 추가해주세요 (생성시 200)
Character 클래스는 move() 메서드를 추가하고 메서드 사용시 health 가 -10 이 됩니다.
Character 클래스는 rest() 메서드를 추가하고 메서드 사용시 health 가 + 10 됩니다.
현재 Health를 알수있는 checkHealth() 메서드를 추가해주세요

2. Knight 와 Healer 클래스 만들어 주세요

Knight 와 Healer 클래스는 Charcter 클래스를 상속합니다.
Knight 클래스는 move() 사용시 Health 가 -5 더 소모
Knight 클래스는 attack() 추가하고 실행시 공격합니다를 출력해주세요
Healer 클래스는 mana속성을 추가해주세요 (생성시 100)
Healer 클래스는 heal(character) 메서드를 추가하고 메서드는 character 들을 매개변수로 받습니다.
Healer 클래스는 heal(character) 메소드 실행시 mana가 -10되고 전달받은 character 객체의 rest() 메소드를 실행합니다.
Healer 클래스는 현재 마나속성을 확인할수있는 checkMana() 메서드를 추가해주세요

소스코드

18-1 실습02.ipynb

19장 예외처리

- 파이썬기초 -

예외처리

프로그램을 실행하다 발생하는 오류

```
>>> print(10 / 0)
```

ZeroDivisionError: integer division or modulo by zero

try:

```
    print(10 / 0) # 실행할 코드
```

except:

```
    print("예외오류발생") # 예외발생시 코드
```

특정예외만 처리

```
x = [1, 2, 3]
```

```
try:
```

```
    print(10 / 0)
```

```
    x[5]
```

```
except ZeroDivisionError as e:
```

```
    print("숫자를 0으로 나눌수 없음", e)
```

```
except IndexError as e:
```

```
    print("잘못된 인덱스", e)
```

예외처리 else와 finally

```
try:
    print(10 / 0)
except:
    print("예외오류발생")
else:
    print("오류발생하지 않음")
finally:
    print("무조건 실행")
```

else : 오류가 발생하지 않을때만 동작
finally : 오류발생여부 상관없이 무조건
동작

예외 발생시키기 raise

```
try:
    raise Exception("예외강제발생")
except Exception as e:
    print("예외발생", e)
```

```
class MyError(Exception):
    def __init__(self):
        super().__init__("나의오류")
```

```
try:
    raise MyError
except Exception as e:
    print("예외발생", e)
```


실습01

아래의 코드를 수정하여 data3.p 파일이 없을 경우
사용자에게 name, address, email 을 입력받아 data3.p 파일을 생성하게 코드를 수정하세요

```
import pickle

with open("data3.p","rb") as file:
    name = pickle.load(file)
    address = pickle.load(file)
    email = pickle.load(file)
    print(name, address, email)
```

소스코드

19-1 실습01.ipynb

실습02

매개변수로 전달받은 숫자의 구구단을 출력해주는 함수를 만드는데
매개변수로 1~9 이외의 숫자를 전달하면 `NotNumberException` 을 발생시켜 주세요

```
>> gugudan (2)
```

```
2 X 1 = 2  
2 X 2 = 4  
  
...  
2 X 9 = 18
```

```
>> gugudan(10)
```

```
NotNumber Traceback (most recent call last)  
...  
NotNumber: 잘못된 숫자입니다.
```

21장 모듈과 패키지

- 파이썬기초 -

모듈

변수, 함수, 클래스 등을 모아놓은 스크립트 파일

모듈 (**calc.py**)

```
name = "calc"
```

```
def add(a, b):
    return a + b
```

```
def sub(a, b):
    return a - b
```

import

```
>>> import calc
```

```
>>> prtin(calc.add(5,6))
```

```
11
```

```
>>> prtin(calc.sub(5,6))
```

```
-1
```

from import

```
>>> from calc import add, sub
```

```
>>> prtin(add(5,6))
```

```
11
```

```
>>> prtin(sub(5,6))
```

```
-1
```

소스코드

calc.py

main.py

시작점 확인 `__name__`

```
name = "calc"
```

파이썬은 어떤 모듈에서든 실행가능

```
def add(a, b):  
    return a + b
```

해당 모듈이 시작점일경우 `__name__` 의 값은 `"__main__"`

```
def sub(a, b):  
    return a - b
```

시작점이 아닐경우 `__name__` 는 해당 모듈의 **모듈명(파일명)**

```
if __name__ == '__main__':  
    print("시작점")
```

소스코드

calc.py

main.py

패키지

여러가지 모듈을 모아놓은것

모듈 (pkcalc/calc.py)

```
def add(a, b):  
    return a + b
```

```
def sub(a, b):  
    return a - b
```

import

```
>>> import pkcalc.calc as calc
```

```
>>> print(calc.name)
```

```
calc
```

```
>>> print(calc.add(5,6))
```

```
11
```

from import

```
>>> from pkcalc.calc import name, add
```

```
>>> print(name)
```

```
calc
```

```
>>> print(add(5,6))
```

```
11
```

소스코드

pkcalc/calc.py

pkcalc/__init__.py

pk_main.py

가치를 높이는 금융 인공지능 실무교육

Insightcampus

패키지명으로 import (__init__.py)

모듈 (pkcalc/calc.py)

```
def add(a, b):  
    return a + b
```

```
def sub(a, b):  
    return a - b
```

pkcalc/__init__.py

```
from .calc import add, sub
```

소스코드

pkcalc/calc.py
pkcalc/__init__.py
pk_main.py

가치를 높이는 금융 인공지능 실무교육

Insightcampus

모듈, 패키지 설치 및 모듈과 패키지를 찾는경로

pip :preferred installer program

pip install 패키지 명 (<https://pypi.org/>)

```
import sys
```

```
print(sys.path)
```

site-packages 폴더에 pip로 설치한 패키지가 들어감

실습01

아래와 같이 동작하도록 pklist 패키지를 만드세요

```
import pklist

pklist.list_max([1,2,3],[5,6,7],[4,3,2,5,6])
>>> 7
pklist.list_min([1,2,3],[5,6,7],[4,3,2,5,6])
>>> 1
pklist.list_avg([1,2,3],[5,6,7],[4,3,2,5,6])
>>> 4.0
```

소스코드

21-1 실습01.ipynb