

빅 데이터 분석을 위한 알고리즘 활용

빅 데이터분석을 위한 알고리즘 소개 및 Python을 활용한 실습



국 성희 (SUNGHEE KUK)

인하대학교 통계학 학사 / 석사 / 박사

2013.03. – 2015.01. 인하대학교 데이터 마이닝 연구소

- . 최적 용매 물질 분석을 위한 데이터 분석 및 처리 (인하대학교)
- . 지역별 알레르기 환자 특성 비교 프로젝트 (인하대 병원)
- . 수요예측을 위한 최적 시계열 모형 도출 알고리즘 개발 (인하대학교)
- . 한국 공항공사 데이터 분석 (사회 책임 윤리 경영 연구소)
- . 마케팅을 위한 고객 특성에 따른 군집화 알고리즘 생성 (SK Ubcare)

2015.01. – 2017.04. 통계 분석 전문 회사 (Begas)

- . 빅데이터 기반 추계모형 사전 컨설팅 (근로복지공단)
- . 분석 솔루션 개발 (베가스)
- . 제조 공정 질량분석 응용 시스템 개발을 위한 예측 모형 설계 및 개발 (중국 Baosteel)
- . 스마트 팩토리 솔루션 개발 (포스코 ICT)
- . 빅데이터 기반 범죄 분석 프로그램 개발 및 플랫폼 구축 (경찰청)

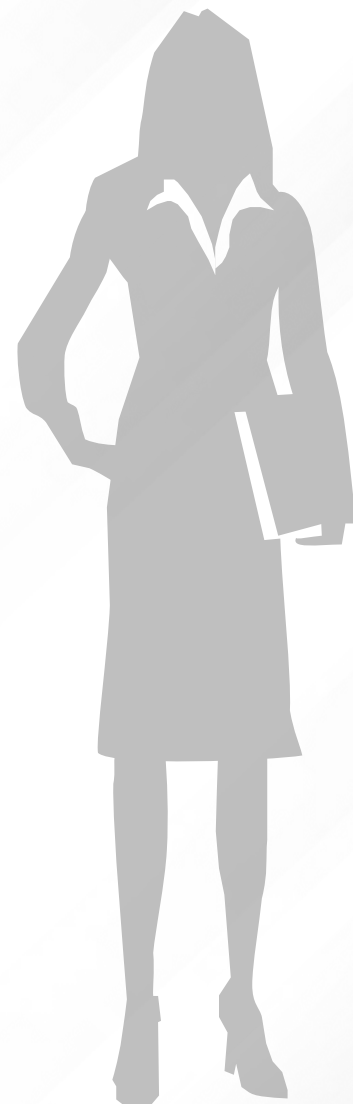
2017.06. – 2021.02. 인하대학교 고급 통계학 연구소

- . 가슴기살균제 건강피해 인정 및 판정기준 개선연구 (환경 독성 보건학회)

2021.01. – 2022.06. LG CNS

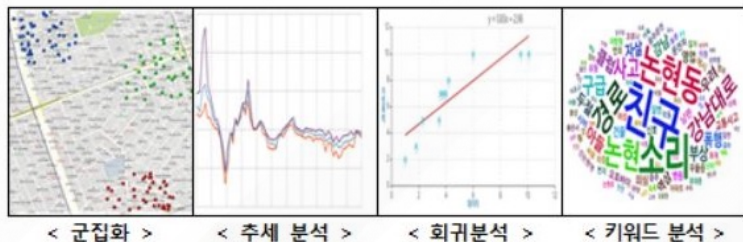
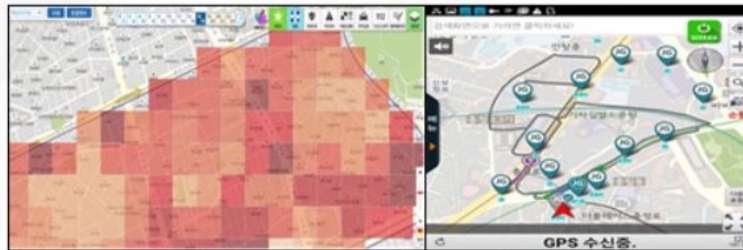
- . 분석 전문가를 위한 교육 프로그램 개발 (LG CNS)
- . 요금제 불만고객탐지 모델 개발 (LG U+)
- . 온라인 행동기반 기업고객 프로파일DB 구축 및 가입상담신청 예측모델개발(LG U+)
- . 영화 콘텐츠 평가 지표 고도화를 위한 지표 및 활용 방안 개발 (LG U+)
- . 불량 분석 솔루션 개발 (LG 에너지 솔루션)

2022.06. – 현재 프리랜서 분석가 & 인하대학교 비전임교원



데이터 분석 사례

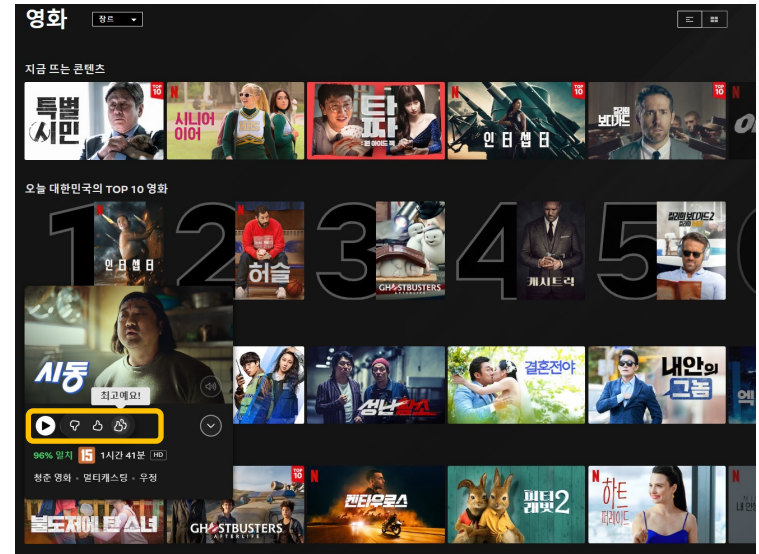
범죄 예측을 위한 데이터 분석



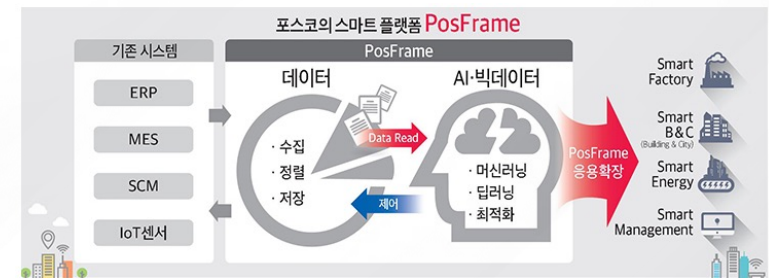
'범죄위험도 예측분석 시스템' 화면 [자료=경찰청]

* 자료 출처
<http://www.theviewers.co.kr/View.aspx?No=I520602>

넷플릭스 영화 추천 알고리즘 구축



제조업에서의 스마트 팩토리, 스마트 인더스트리



▲ 사진설명: 포스프레임(PosFrame)을 스마트인더스트리로 응용확장하는 과정

* 자료 출처
<https://newsroom.posco.com/kr/포스코-포스프레임posframe으로-스마트인더스트리-주도/>

데이터 분석 사례

영화 콘텐츠 평가지표 개발



인기도

- 영화를 얼마나 많이 시청하였는가
- 영화를 시청한 사람이 얼마나 많은가

몰입도

- 영화를 시청하는데 시청자들이 얼마나 몰입해서 시청하였는가

신선도

- 영화가 서비스된지 얼마가 지났는가
- 영화가 개봉된지 얼마가 지났는가

군집 지표

- 유사한 특성을 가진 영화들에 대한 군집을 생성하여 영화별 군집들에 대한 유사도는 어느 정도인가

통계량 개발

- Exact null distribution for the likelihood ratio test statistic

- LRT statistic

$$LRT = 2 \log \frac{\sup_{H_1} L(b, \tau^2)}{\sup_{H_0} L(b, \tau^2)} = \sup_{\tau \geq 0} \{-\log |V_\lambda| - Y^T P_\lambda^T V_\lambda^{-1} P_\lambda Y + \log |V_{\tau=0}| + Y^T P_{\tau=0}^T V_{\tau=0}^{-1} P_{\tau=0} Y\}$$

$V_{\tau=0}$ denotes V_λ under H_0 and $P_{\tau=0} = I_n - 1(1^T V_{\tau=0}^{-1} 1)^{-1} 1^T V_{\tau=0}^{-1}$

- RLRT statistic

$$RLRT = \sup_{\tau \geq 0} \{-\log |V_\lambda| - Y^T P_\lambda^T V_\lambda^{-1} P_\lambda Y - \log |1^T V_\lambda 1| + \log |V_{\tau=0}| + Y^T P_{\tau=0}^T V_{\tau=0}^{-1} P_{\tau=0} Y + \log |1^T V_{\tau=0} 1|\}$$

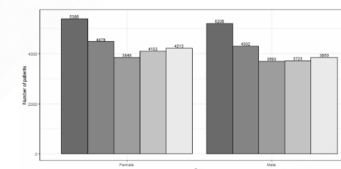
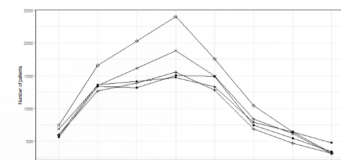
Let $Z = (Z_1, \dots, Z_n)^T$ where Z_i is generated from $N(0, \sigma_i^2)$.

$$LRT^* = \sup_{\tau \geq 0} \{-\log |V_\lambda| - Z^T P_\lambda^T V_\lambda^{-1} P_\lambda Z + \log |V_{\tau=0}| + Z^T P_{\tau=0}^T V_{\tau=0}^{-1} P_{\tau=0} Z\}$$

Suppose that we generate LRT_1^*, \dots, LRT_B^* for a large B . the p-value is computed as

$$\frac{1}{B} \sum_{j=1}^B (LRT \leq LRT_j^*)$$

의료분야 연구 분석



		Status ¹		Univariate ²		Multivariate ²	
		No metabolic diseases (%) ³	Metabolic diseases (%) ³	Odds ratio ² confidence interval ²	95% confidence interval ²	Odds ratio ² confidence interval ²	95% confidence interval ²
Zepidol ²	χ^2	615 (80.4) ³	149 (19.6) ³	0.814 ²	(0.6795, 0.9789) ²	1.2012 ²	(0.9805, 1.4823) ²
	χ^2	803 (78.0) ³	227 (22.0) ³	0.9500 ²	(0.8117, 1.1036) ²	1.1090 ²	(1.0086, 1.4170) ²
Olazapine ²	χ^2	5670 (79.5) ³	1429 (20.5) ³	0.8079 ²	(0.7566, 0.8627) ²	0.8962 ²	(0.8237, 0.9751) ²
	χ^2	7604 (75.8) ³	2422 (24.2) ³	1.1124 ²	(1.0510, 1.1774) ²	1.1284 ²	(1.0446, 1.2189) ²
Amisulpride ²	χ^2	2057 (79.5) ³	530 (20.5) ³	0.8561 ²	(0.7750, 0.9458) ²	1.0579 ²	(0.8436, 1.3361) ²
	χ^2	13227 (77.1) ³	3926 (22.9) ³	0.9981 ²	(0.9448, 1.0545) ²	0.9721 ²	(0.8921, 1.0591) ²
Aripiprazole ²	χ^2	4944 (82.5) ³	1050 (17.5) ³	0.6627 ²	(0.6161, 0.7123) ²	0.9693 ²	(0.8855, 1.0630) ²
	χ^2	5190 (81.2) ³	718 (18.8) ³	0.7518 ²	(0.6904, 0.8185) ²	1.0064 ²	(0.9085, 1.1147) ²

¹ In multivariate analysis, "Number of the types of antipsychotics use" was used as a continuous variable to avoid multicollinearity. ² Corresponding groups that did not use the respective antipsychotic drug was used as reference.

데이터 분석 프로세스

데이터 분석 프로세스에는 1. 데이터 수집 2. 데이터 탐색 3. 모델링 4. 모형평가 단계로 나뉩습니다.

각 단계에서 가장 많은 시간이 소요되는 단계는 “데이터 탐색”으로 단순히 분석 알고리즘을 적용하는 것만큼 매우 중요한 단계 입니다.

01



데이터 수집

- 분석에 활용할
데이터 수집 및 추출

02



데이터 탐색

- 데이터 정제(전처리)
- 데이터 시각화
- 데이터 변환

03



모델링

- 분석 알고리즘 적용

04



모형 평가

- 알고리즘 성능 비교
- 알고리즘 성능 평가



CONTENTS

DATA Analysis

01

파이썬 프로그래밍 기초

파이썬에서 활용 가능한 자료의 형태와 기능들에 대해
학습함

02

데이터 수집

데이터 수집과정을 소개하고 파이썬을 이용하여 데이터를
수집하는 방법을 학습함

03

데이터 전처리 및 시각화

분석 데이터의 구조 및 형태와 파이썬을 활용하여 분석 데이터를
생성하기 위한 방법들을 학습함

04

AI 알고리즘 맛보기

회귀모형과, Tree기반의 앙상블 모형에 대해 간단히
학습하고 파이썬으로 생성함

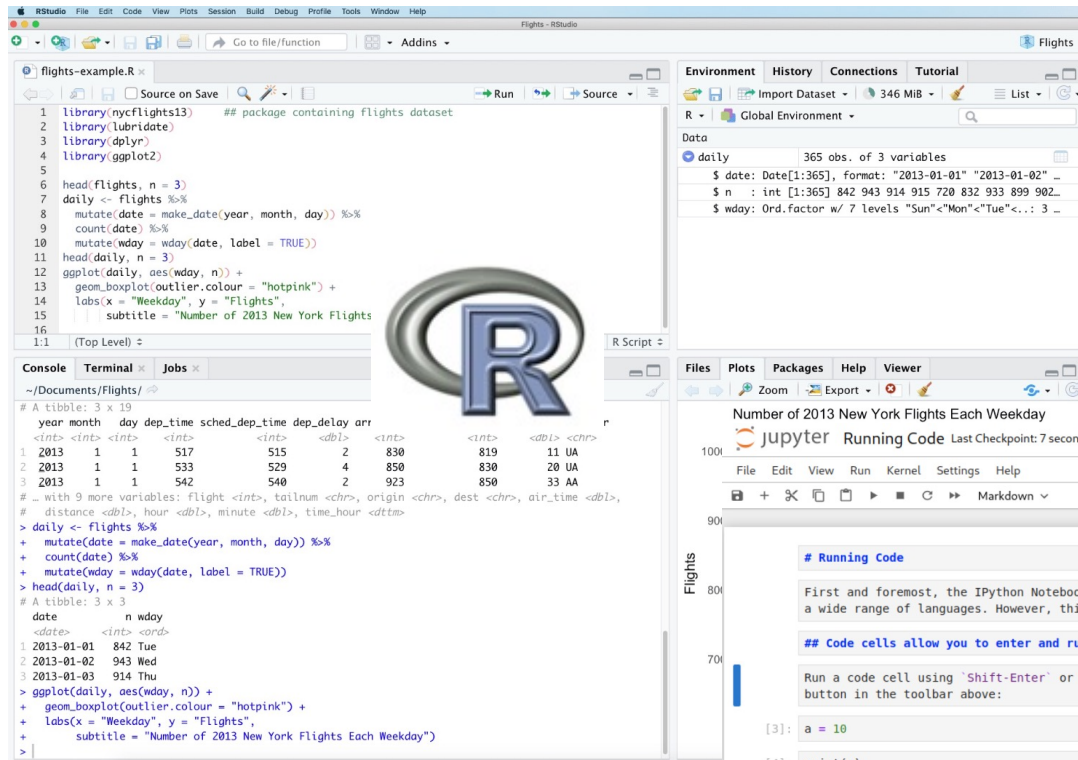
05

모형 평가 방법

다양한 AI 알고리즘들을 생성하고 평가하는 방법에 대해
학습함

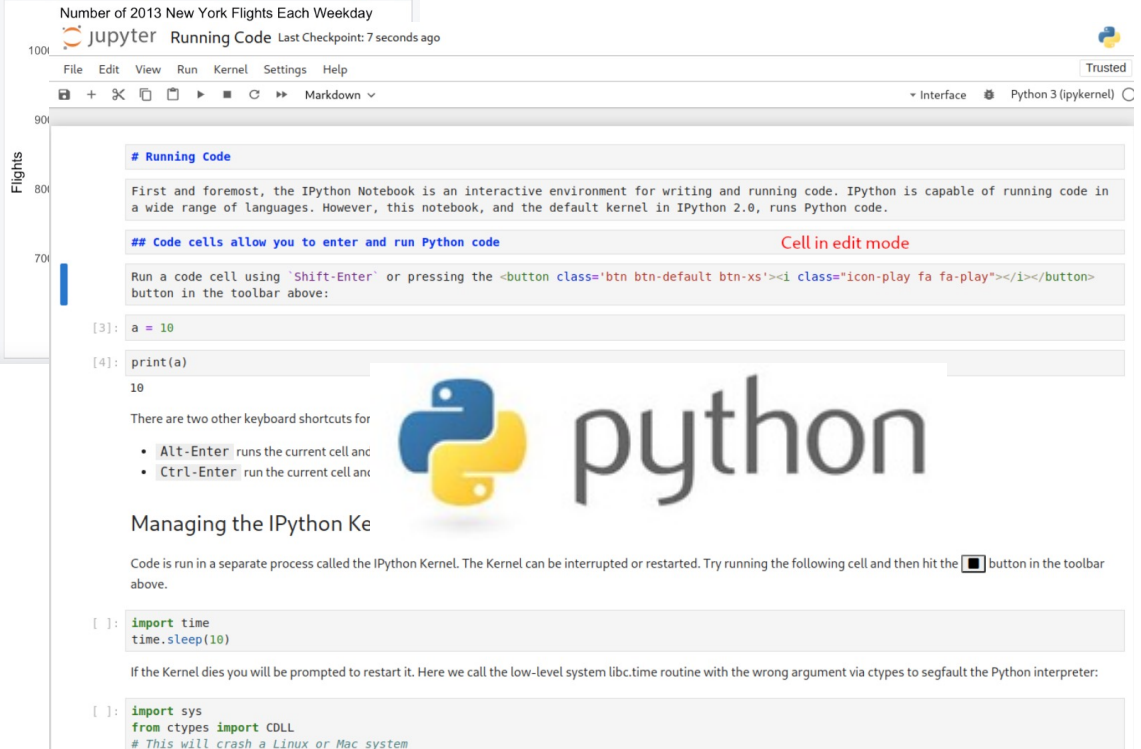
1.2 코랩(Colab) 사용하기

01. 파이썬 프로그래밍 기초



The screenshot shows the RStudio IDE with a script editor, console, and environment pane. The script editor contains R code for loading the nycflights13 dataset, creating a 'daily' dataset, and plotting the number of flights by weekday. The console shows the output of the code, including a tibble of flight data and a plot of flights by weekday.

```
1 library(nycflights13) ## package containing flights dataset
2 library(lubridate)
3 library(dplyr)
4 library(ggplot2)
5
6 head(flights, n = 3)
7 daily <- flights %>%
8   mutate(date = make_date(year, month, day)) %>%
9   count(date) %>%
10  mutate(wday = wday(date, label = TRUE))
11 head(daily, n = 3)
12 ggplot(daily, aes(wday, n)) +
13   geom_boxplot(outlier.colour = "hotpink") +
14   labs(x = "Weekday", y = "Flights",
15        subtitle = "Number of 2013 New York Flights
16
17 (Top Level)
18
19 Console
20 ~/Documents/Flights/
21 # A tibble: 3 x 19
22   year month   day dep_time sched_dep_time dep_delay arr
23   <int> <int> <int> <chr>      <dbt>      <int>      <dbt> <chr>
24 1 2013     1     1   517         515          2      830   11 UA
25 2 2013     1     1   533         529          4      850   20 UA
26 3 2013     1     1   542         540          2      850   33 AA
27 # _ with 9 more variables: flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
28 # distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
29 > daily <- flights %>%
30 +   mutate(date = make_date(year, month, day)) %>%
31 +   count(date) %>%
32 +   mutate(wday = wday(date, label = TRUE))
33 > head(daily, n = 3)
34 # A tibble: 3 x 3
35   date       n wday
36   <date>   <int> <ord>
37 1 2013-01-01 842 Tue
38 2 2013-01-02 943 Wed
39 3 2013-01-03 914 Thu
40 > ggplot(daily, aes(wday, n)) +
41 +   geom_boxplot(outlier.colour = "hotpink") +
42 +   labs(x = "Weekday", y = "Flights",
43         subtitle = "Number of 2013 New York Flights Each Weekday")
44 >
```



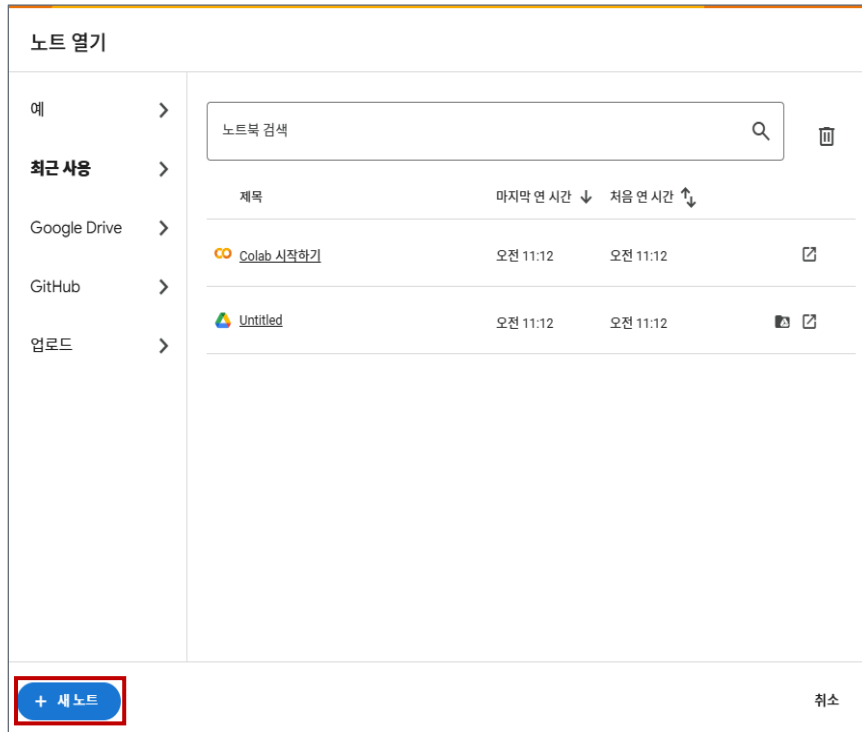
The screenshot shows the Jupyter Notebook interface with a code editor and output area. The code editor contains Python code for running a cell, and the output area shows the result of the code. The notebook also includes a section on managing the IPython kernel.

```
# Running Code
First and foremost, the IPython Notebook is an interactive environment for writing and running code. IPython is capable of running code in a wide range of languages. However, this notebook, and the default kernel in IPython 2.0, runs Python code.
## Code cells allow you to enter and run Python code
Run a code cell using 'Shift-Enter' or pressing the button in the toolbar above:
[3]: a = 10
[4]: print(a)
10
There are two other keyboard shortcuts for
• Alt-Enter runs the current cell and
• Ctrl-Enter run the current cell and
Managing the IPython Ke
Code is run in a separate process called the IPython Kernel. The Kernel can be interrupted or restarted. Try running the following cell and then hit the button in the toolbar above.
[ ]: import time
time.sleep(10)
If the Kernel dies you will be prompted to restart it. Here we call the low-level system libc.time routine with the wrong argument via ctypes to segfault the Python interpreter:
[ ]: import sys
from ctypes import CDLL
# This will crash a Linux or Mac system
```

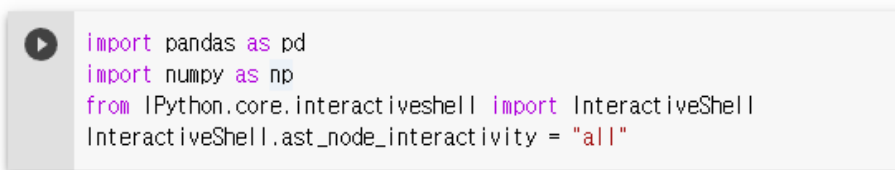

1.2 코랩(Colab) 사용하기

01. 파이썬 프로그래밍 기초

- 구글에서 코랩을 검색하거나 <https://colab.research.google.com>에 직접 접속합니다.
- 새 노트를 클릭합니다.



- 각 셀들을 실행시키고 싶은 경우 셀을 클릭 후, Ctrl+Enter 혹은 Shift+Enter를 누릅니다.



- 숫자형(Number)이란 숫자 형태로 이루어진 자료형으로, 이미 잘 알고 있는 것들입니다. 숫자형으로 사용되는 항목에는 정수, 실수, 8진수, 16진수, 복소수 형태가 있습니다.
- 그 중, 정수형과 실수형의 형태가 가장 많이 활용되는 형태라고 할 수 있습니다.

숫자형

■ 숫자형 자료

항목	파이썬 사용 예
정수 (Integer)	123, -345, 0
실수 (Floating-point)	123.45 -1234.5, 3.4e10
8진수 (Octal)	0o34, 0o25
16진수 (Hexadecimal)	0x2A, 0xFF
복소수 (Complex number)	1+2j, -3j

파이썬 실습

■ 정수형 / 실수형

▼ 정수형

```
[ ] a = 123
a
```

```
123
```

```
[ ] a = -345
a
```

```
-345
```

```
[ ] a = 0
a
```

```
0
```

▼ 실수형

```
[ ] a = 123.45
a
```

```
123.45
```

```
[ ] a = -1234.5
a
```

```
-1234.5
```

```
a = 3.4e-10
a
```

```
3.4e-10
```

■ 8진수와 16진수

```
a = 0o177
a
```

```
127
```

```
[ ] a = 0x8ff
a
```

```
2303
```

```
[ ] b = 0xABC
b
```

```
2748
```

- 숫자형(Number)이란 숫자 형태로 이루어진 자료형으로, 이미 잘 알고 있는 것들입니다. 숫자형으로 사용되는 항목에는 정수, 실수, 8진수, 16진수, 복소수 형태가 있습니다.
- 그 중, 정수형과 실수형의 형태가 가장 많이 활용되는 형태라고 할 수 있습니다.

숫자형

■ 숫자형 자료

항목	파이썬 사용 예
정수 (Integer)	123, -345, 0
실수 (Floating-point)	123.45 -1234.5, 3.4e10
8진수 (Octal)	0o34, 0o25
16진수 (Hexadecimal)	0x2A, 0xFF
복소수 (Complex number)	1+2j, -3j

파이썬 실습

■ 복소수

```
[ ] a = 1+2j
a
```

↻ (1+2j) 복소수는 괄호로 표기됩니다.

```
[ ] b = 3-4j
b
```

↻ (3-4j)

```
[ ] a.real      복소수의 실수 부분을 리턴합니다.
```

↻ 1.0

```
[ ] a.imag      복소수의 허수 부분을 리턴합니다.
```

↻ 2.0

```
[ ] a.conjugate()      복소수의 켤레복소수를 리턴합니다.
```

↻ (1-2j)

```
[ ] abs(a)      복소수의 절댓값을 리턴합니다.
```

↻ 2.23606797749979

- 숫자형(Number)을 활용하기 위해 기본적인 사칙 연산 뿐만 아니라 다양한 형태의 연산자를 사용할 수 있습니다.

사칙 연산

```
[1] a = 3  
    b = 4  
    a+b
```

```
↔ 7
```

```
[2] a*b
```

```
↔ 12
```

```
[3] a/b
```

```
↔ 0.75
```

x의 y제곱

```
▶ a = 3  
   b = 4  
   a**b
```

```
↔ 81
```

나눗셈 관련

```
▶ 7 % 3
```

```
↔ 1
```

```
[6] 7/4
```

```
↔ 1.75
```

```
[7] 7//4
```

```
↔ 1
```

- 문자열(String)이란 문자, 단어 등으로 구성된 문자들의 집합을 의미합니다.
- 문자열의 덧셈과 곱셈 연산으로부터 문자열을 수정할 수 있으며, 문자열 내에 특정 단어나 글자를 선택할 수 있습니다.

문자열 입력 방법

[8] "Hello World"	양쪽에 " 사용
↔ 'Hello World'	
[9] 'Python is fun'	양쪽에 ' 사용
↔ "Python is fun"	
[10] """Life is too short, You need python"""	양쪽에 """사용
↔ 'Life is too short, You need python'	
[11] '''Life is too short, You need python'''	양쪽에 ''' 사용
↔ "Life is too short, You need python"	

사용 방법 구분

[12] "Python's favorite food is perl"	문장 안에 ' 가 있는 경우
↔ 'Python's favorite food is perl'	
[14] ""Python is very easy." he says."	문장 안에 " 가 있는 경우
↔ "Python is very easy." he says."	
▶ multiline = ''' Life is too short You need python ... print(multiline)	여러 줄의 문자열을 생성하는 경우 (원래는 \n 를 사용해야 함)
↔ Life is too short You need python	

문자열 더해서 연결하기 (Concatenation)

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

'Python is fun!'
```

문자열 곱하기

```
a = "python"
a*2

'pythonpython'
```

문자열 인덱싱 및 슬라이싱

L	i	f	e		i	s		t	o	o		s	h	o	r	t	,		Y	o	u		n	e	e	d		P	y	t	h	o	n
0										1										2										3			
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3

```
a = "Life is too short, You need python"
print(a[0])
print(a[-2])

L
o
```

```
print(a[0:4])
print(a[19:])
print(a[19:-7])

Life
You need python
You need
```


- 파이썬에는 사용할 수 있는 다양한 자료형들이 존재합니다.
- 다양한 파이썬의 자료형들을 학습하면, 데이터 분석을 위한 코드작성 시 유용하게 활용할 수 있습니다.

항목	파이썬 사용 예	특징
리스트 (List)	리스트 명 = [요소1, 요소2, 요소3, ...]	여러 개의 숫자와 문자모음을 저장
튜플 (tuple)	튜플 명 = (요소1, 요소2, 요소3, ...)	리스트와 거의 같으나 값을 바꿀 수 없음
딕셔너리(dictionary)	딕셔너리명 = {Key1:Value1, Key2:Value2, ..}	대응 관계를 나타낼 수 있음
집합	집합명 = set(리스트, 문자 ... 등등),	자료의 순서가 없고, 중복을 허용 안함

1.3 파이썬 자료의 형태(리스트)

01. 파이썬 프로그래밍 기초

- 숫자와 문자열을 모음으로 저장하는 경우 “리스트 자료형”을 사용할 수 있습니다.

리스트 생성하기

```
[1] a = []  
b = [1, 2, 3]  
c = ['Life', 'is', 'too', 'short']  
d = [1, 2, 'Life', 'is']  
e = [1, 2, ['Life', 'is']]  
print(a)  
print(b)  
print(c)  
print(d)  
print(e)
```

```
☞ []  
[1, 2, 3]  
['Life', 'is', 'too', 'short']  
[1, 2, 'Life', 'is']  
[1, 2, ['Life', 'is']]
```

리스트 인덱싱과 슬라이싱

```
[5] e = [1, 2, ['Life', 'is']]  
print(e[0])  
print(e[2])
```

```
☞ 1  
['Life', 'is']
```

```
[8] print(e[-1][0])  
print(e[-1][1])
```

```
☞ Life  
is
```

```
[9] a = [1, 2, 3, 4, 5]  
a[0:2]
```

```
☞ [1, 2]
```

```
[10] a = "12345"  
a[0:2]
```

```
☞ '12'
```

리스트 연산자

```
▶ a = [1, 2, 3]  
b = [4, 5, 6]  
print(a + b)  
print(a * 3)
```

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

```
[13] a[2] = 4  
print(a)  
  
a[1:2] = ['a', 'b', 'c']  
print(a)
```

```
☞ [1, 2, 4]  
[1, 'a', 'b', 'c', 4]
```

```
[14] a[1:3] = []  
a
```

```
☞ [1, 'c', 4]
```

```
[16] a = [1, 2, 3]  
a.append(4)  
print(a)  
a = [1, 4, 3, 2]  
a.sort()  
print(a)  
a.reverse()  
print(a)
```

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

1.3 파이썬 자료의 형태(튜플)

01. 파이썬 프로그래밍 기초

- 튜플은 리스트와 거의 비슷하지만 입력 방법과, “수정이 불가능하다”는 점이 크게 다른 점 입니다.

튜플 생성하기

```
[18] t1 = ()
      t2 = (1,)
      t3 = (1, 2, 3)
      t4 = 1, 2, 3
      t5 = ('a', 'b', ('ab', 'cd'))
      print(t1)
      print(t2)
      print(t3)
      print(t4)
      print(t5)
```

```
↩ ()
  (1,)
  (1, 2, 3)
  (1, 2, 3)
  ('a', 'b', ('ab', 'cd'))
```

튜플 요소값 변경 오류

```
[19] t1 = (1, 2, 'a', 'b')
      t1[0] = 'c'

↩ -----
      TypeError                                 Traceback (most recent call last)
      <ipython-input-19-1978018605> in <cell line: 0>()
          1 t1 = (1, 2, 'a', 'b')
      ----> 2 t1[0] = 'c'

      TypeError: 'tuple' object does not support item assignment
```

다음 단계: 오류 설명

```
[21] del t1[0]

↩ -----
      TypeError                                 Traceback (most recent call last)
      <ipython-input-21-2661958664> in <cell line: 0>()
      ----> 1 del t1[0]

      TypeError: 'tuple' object doesn't support item deletion
```

튜플의 인덱싱, 슬라이싱 연산

```
[24] print(t1[3])
      print(t1[1:])
      t2 = (3, 4)
      print(t1 + t2)
      print(t2 * 3)

↩ b
  (2, 'a', 'b')
  (1, 2, 'a', 'b', 3, 4)
  (3, 4, 3, 4, 3, 4)
```

1.3 파이썬 자료의 형태(딕셔너리)

01. 파이썬 프로그래밍 기초

- 딕셔너리 자료형은 “이름”=“홍길동”, “생일”=“06월 01일”등의 대응관계를 나타낼 수 있는 자료형 입니다.

딕셔너리 입력방법

Key	Value
name	pey
phone	01011112222
birth	0601
height	150

```
[28] dic = {'name':'pey', 'phone':'01011112222', 'birth':'0601'}  
      dic['height']=150  
      print(dic)  
{'name': 'pey', 'phone': '01011112222', 'birth': '0601', 'height': '150'}
```

딕셔너리 사용방법

```
[34] print(dic['name'])  
pey  
[35] del dic['height']  
      print(dic)  
{'name': 'pey', 'phone': '01011112222', 'birth': '0601'}
```

딕셔너리 관련 함수

```
dic.keys()  
dict_keys(['name', 'phone', 'birth'])  
  
dic.values()  
dict_values(['pey', '01011112222', '0601'])  
  
dic.items()  
dict_items([('name', 'pey'), ('phone', '01011112222')])  
  
dic.clear()  
print(dic)  
{}
```

1.3 파이썬 자료의 형태(집합)

01. 파이썬 프로그래밍 기초

- 집합은 파이썬2.3부터 지원되기 시작한 자료형으로 집합에 관련된 것들을 쉽게 처리하기 위해 만들어진 자료형 입니다.

집합 입력방법

```
s1 = set([1,2,3])
s1

{1, 2, 3}

s2 = set("Hello")
s2

{'H', 'e', 'l', 'o'}
```

- 중복 허용 x
- 순서가 없음

집합 자료형 활용 방법

```
s1 = set([1,2,3,4,5,6])
s2 = set([4,5,6,7,8,9])

print(s1 & s2)
print(s1.intersection(s2))

{4, 5, 6}
{4, 5, 6}

print(s1 | s2)
print(s1.union(s2))

{1, 2, 3, 4, 5, 6, 7, 8, 9}
{1, 2, 3, 4, 5, 6, 7, 8, 9}

print(s1 - s2)
print(s1.difference(s2))

{1, 2, 3}
{1, 2, 3}
```

교집합

합집합

차집합

집합 관련 함수들

```
s1 = set([1,2,3])
s1.add(4)
s1

{1, 2, 3, 4}

s1.update([4,5,6])
s1

{1, 2, 3, 4, 5, 6}

s1.remove(2)
s1

{1, 3, 4, 5, 6}
```


연습문제

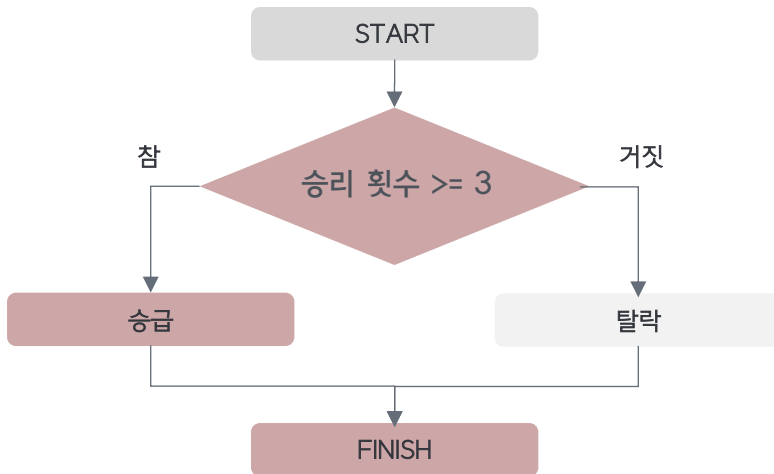
- Q1** 홍길동 씨의 주민등록번호는 881120-1068234이다. 홍길동씨의 주민등록번호를 연월일(YYYYMMDD)부분과 그 귀의 숫자 부분으로 나누어 출력하여라
- Q2** ['Life', 'is', 'too', 'short']라는 리스트를 Life is too short라는 문자열로 만들어 출력하여라.
- Q3** (1,2,3)이라는 튜플에 4라는 값을 추가하여 (1,2,3,4)처럼 만들어 출력하여라.
- Q4** "A"=90, "B"=80, "C"=70로 대응되는 딕셔너리를 생성하고, "B"에 해당되는 값을 추출하고 삭제하여라.

- 알고리즘을 개발하기 위해 '조건문'은 없어서는 안되는 부분입니다.
- 조건문은 사람이 어떤 상황에 놓여져 있을 때, 판단하는 것처럼 컴퓨터에게 특정 조건이 주어졌을 때 행동해야 하는 방법을 인식시켜주는 부분이라고 생각할 수 있습니다.

if문 (if / else)

■ 기본 개념

승급전에서 세판 이상 이기면 승급하고,
세 판 미만으로 이기면 떨어진다.



조건 연산자

■ 조건 연산자 요약

비교 연산자	내용
$x < y$	x가 y보다 작다
$x > y$	x가 y보다 크다
$x == y$	x가 y와 같다
$x != y$	x가 y와 같지 않다
$x >= y$	x가 y보다 크거나 같다
$x <= y$	x가 y보다 작거나 같다
x조건 or y조건	x조건과 y조건 둘 중에 하나만 참이면 참
x조건 and y조건	x조건과 y조건 모두 참이어야 참
not x조건	x조건이 거짓이면 참
x in 리스트/튜플/문자열	리스트/튜플/문자열 안에 x가 있으면 참
x not in 리스트/튜플/문자열	리스트/튜플/문자열 안에 x가 있지 않으면 참

- 알고리즘을 개발하기 위해 '조건문'은 없어서는 안되는 부분입니다.
- 조건문은 사람이 어떤 상황에 놓여져 있을 때, 판단하는 것처럼 컴퓨터에게 특정 조건이 주어졌을 때 행동해야 하는 방법을 인식시켜주는 부분이라고 생각할 수 있습니다.

if문 (if / else)

■ 기본 구조

if 조건문  콜론 입력!

수행할 문장 1

수행할 문장 2

동일한 들여쓰기로
한 블록을 생성

수행할 문장 3

→ 들여쓰기 위치 주의!

else : 콜론 입력!

수행할 문장 1

수행할 문장 2

동일한 들여쓰기로
한 블록을 생성

Python 코드

```
[14] # 전체 5판 중 게임에서 이긴 총 수
win = 3

if win >=3 :
    print("축하합니다. 승급하셨습니다.")
else :
    print("안타깝게도 승급에 실패하셨습니다.")
```

축하합니다. 승급하셨습니다.

```
[15] # 전체 5판 중 게임에서 이긴 총 수
win = 2

if win >=3 :
    print("축하합니다. 승급하셨습니다.")
else :
    print("안타깝게도 승급에 실패하셨습니다.")
```

안타깝게도 승급에 실패하셨습니다.

- 알고리즘을 개발하기위해 '조건문'은 없어서는 안되는 부분입니다.
- 조건문은 사람이 어떤 상황에 놓여져 있을 때, 판단하는 것처럼 컴퓨터에게 특정 조건이 주어졌을 때 행동해야 하는 방법을 인식시켜주는 부분이라고 생각할 수 있습니다.

elif 추가 문

■ 기본 구조

if 조건문A :

수행할 문장 1

수행할 문장 2

elif 조건문B :

수행할 문장 1

수행할 문장 2

이전 조건문인
조건문A가 거짓이고
조건문B이 참일 때, 수행

else :

수행할 문장 1

수행할 문장 2

Python 코드

변수 age가
0이상 9이하 이면 '영유아'/10이상 19이하 이면 '10대',
20이상 29이하 이면 '20대'/30이상 39이하 이면 '30대'
40이상 49이하 이면 '40대'/아무것도 해당하지 않으면 '기타'가

출력하는 조건문을 생성해라.

해답

- 동일한 동작을 여러 번 반복해야 하는 경우, 코드를 한 줄씩 입력하는 것은 매우 비효율적인 일이며, 많은 '행' 데이터를 확인하거나 처리하기 위해서는 반복문을 반드시 활용할 수 있어야 합니다.
- 반복문에는 상황에 따라 while문 for문과 같은 문법을 사용합니다.

while문

■ 기본 구조

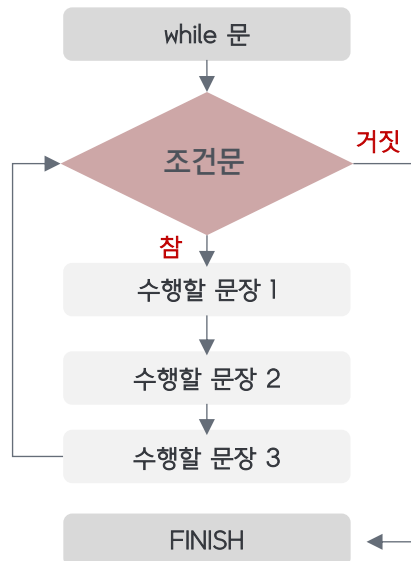
- while문은 조건문이 참인 '동안' 그 안에 속해 있는 문장들을 반복해서 수행하는 문법임

while 조건문 :

수행할 문장 1

수행할 문장 2

수행할 문장 3



break / continue

■ break

- 반복문 내부에서 강제로 빠져나오고 싶을 때 사용하는 문법임

■ continue

- continue문에서 제시하는 특정 조건을 만족하는 경우, 그 하위 문장을 실행하지 않고 반복문의 시작지점으로 위치를 이동시킴

while 조건문 :

if 조건문 :

break / continue

수행할 문장 1

- 동일한 동작을 여러 번 반복해야 하는 경우, 코드를 한 줄씩 입력하는 것은 매우 비효율적인 일이며, 많은 '행' 데이터를 확인하거나 처리하기 위해서는 반복문을 반드시 활용할 수 있어야 합니다.
- 반복문에는 상황에 따라 while문 for문과 같은 문법을 사용합니다.

Python 코드 (기본, continue)

```
playGame = 0

while playGame <= 3 :
    print("게임을 %d 판 했습니다." %playGame)

    if playGame == 3:
        print("마지막 게임이었습니다.")

    playGame = playGame + 1
```

게임을 0 판 했습니다.
게임을 1 판 했습니다.
게임을 2 판 했습니다.
게임을 3 판 했습니다.
마지막 게임이었습니다.

```
# continue
a = 0
while a < 6 :
    a = a + 1
    if a % 2 == 0: continue
    print(a)
```

1
3
5

Python 코드 (break)

```
# break
item = 4
money = 30000
myitem = 0

while money > 1300 :
    print("아이템을 구매합니다.")
    item = item - 1
    myitem = myitem + 1
    money = money - 500
    print("나의 아이템은 " + str(myitem)+ "개, 남은 돈은 " +str(money)+"원입니다.")
    print("상점에 남은 아이템은 "+str(item)+"개 입니다.\n")
    if not item:
        print("아이템이 전부 소진되었습니다. 상점 문을 닫습니다.")
        break
```

아이템을 구매합니다.
나의 아이템은1개, 남은 돈은 29500원입니다.
상점에 남은 아이템은 3개 입니다.

아이템을 구매합니다.
나의 아이템은2개, 남은 돈은 29000원입니다.
상점에 남은 아이템은 2개 입니다.

아이템을 구매합니다.
나의 아이템은3개, 남은 돈은 28500원입니다.
상점에 남은 아이템은 1개 입니다.

아이템을 구매합니다.
나의 아이템은4개, 남은 돈은 28000원입니다.
상점에 남은 아이템은 0개 입니다.

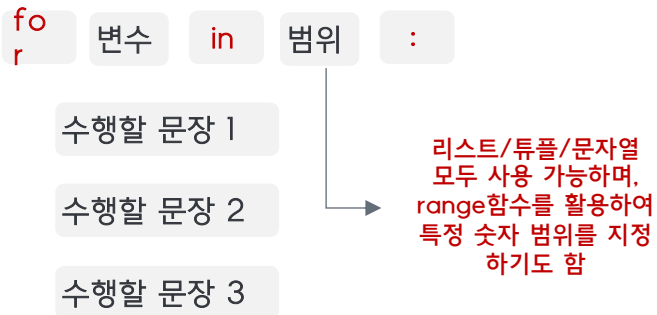
아이템이 전부 소진되었습니다. 상점 문을 닫습니다.

- 동일한 동작을 여러 번 반복해야 하는 경우, 코드를 한 줄씩 입력하는 것은 매우 비효율적인 일이며, 많은 '행' 데이터를 확인하거나 처리하기 위해서는 반복문을 반드시 활용할 수 있어야 합니다.
- 반복문에는 상황에 따라 while문 for문과 같은 문법을 사용합니다.

for문

■ 기본 구조

- for문은 가장 기본적이고 대표적인 반복문으로 범위를 지정하여 범위에 해당하는 내용을 하나씩 순차적으로 대입하여 수행할 문장을 반복하는 문법임



Python 코드

```
# 기본 구조
test_list = ['one', 'two', 'three']
for i in test_list :
    print(i)
```

```
one
two
three
```

```
# 튜플 리스트 예제
a = [(1,2), (3,4), (5,6)]
for (f, l) in a :
    print(f+l)
```

```
3
7
11
```

```
# range 함수를 활용한 for문
# 1부터 10까지 더하기
sum = 0
for i in range(1, 11) :
    sum = sum + i
print(sum)
```

```
55
```

- 동일한 동작을 여러 번 반복해야 하는 경우, 코드를 한 줄씩 입력하는 것은 매우 비효율적인 일이며, 많은 ‘행’ 데이터를 확인하거나 처리하기 위해서는 반복문을 반드시 활용할 수 있어야 합니다.
- 반복문에는 상황에 따라 while문 for문과 같은 문법을 사용합니다.

반복문과 조건문 활용 예시

■ 기본 구조

```
sample3 = pd.read_csv("sample3.csv")  
sample3.head()
```

	id	course_nm	status
0	1234	computer application	allowed
1	1235	computer application	allowed
2	1236	statistic	allowed
3	1237	statistic	allowed
4	1238	statistic	allowed

수강생이 30명 이상인 경우, "Large Room",
20명 이상인 경우, "Medium Room",
10명 이상인 경우 "Small Room"으로 배정하고,
수강인원이 10명 미만이어서 status가 "not allowed"인 경우,
"not assigned"으로 배정하는 변수를 생성해보자

Python 코드

해답

- 현업 혹은 실제로 데이터 분석을 하는 경우 특정 로직이 여러 군데에서 활용되는 경우가 많습니다.
- 이러한 경우 동일한 코드를 여러 번 반복해서 사용하는 것보다 나만의 함수로 정의하여 사용하는 것이 훨씬 효율적인 방법입니다.

함수

■ 기본 구조

- 반복적으로 사용되는 부분을 여러 군데에서 활용할 수 있도록 묶어서 정의하여 활용함

`def` 함수명(입력 인수) :

수행할 문장 1

수행할 문장 2

`return` 상황에 따라 생략 가능

Python 코드

```
def mySum(a, b) :  
    print(a + b)  
  
mySum(1,2)
```

3

```
def mySum(a, b) :  
    return a + b  
  
mySum(1, 2)
```

3

```
def mySum_fn(method, *args) :  
    if method == 'sum' :  
        result = 0  
        for i in args :  
            result = result + i  
    elif method == 'mul' :  
        result = 1  
        for i in args :  
            result = result * i  
    return result
```

```
mySum_fn('sum', 1,2,3,4)
```

10

연습문제

Q5 다음 코드의 결과값은 무엇인가?

```
a = "Life is too short, you need python"

if "wife" in a: print("wife")
elif "python" in a and "you" not in a: print("python")
elif "shirt" not in a: print("shirt")
elif "need" in a: print("need")
else: print("none")
```

Q6 While문을 이용하여 아래와 같이 별(*)을 표시하는 프로그램을 작성하여라.

```
*
**
***
****
*****
```


연습문제

Q7

A학급에 총 10명의 학생이 있다. 이 학생들의 중간고사 점수는 다음과 같다. for문을 이용하여 A학급의 평균 점수를 구하여라.

70, 60, 55, 75, 95, 90, 80, 80, 85, 100

```
A = [70, 60, 55, 75, 95, 90, 80, 80, 85, 100]
```

```
total =   
for  in  :  
    total +=   
    average =   
    print()
```