Python 기초

Lee Jhun Hyeok (wnsx0000@gmail.com)

March 6, 2024

목차

Part I

기본 문법

1. 서론

1.1. 프로그래밍 환경 구축

1.1.1. 파이썬 IDE

파이썬 개발환경. IDLE, 아나콘다, 주피터 노트북 등이 있음.

1.1.2. VSC

웹이나 마이크로소프트 앱스토어 등에서 파이썬을 다운받고, VSC에서 IntelliSense 패키지를 받아 환경설정 하여 사용할 수 있음.

1.2. 파이썬 코드 편집 모드

1.2.1. 쉘 모드

하나의 명령 단위로 명령을 실행하는 모드.

1.2.2. 코드 에디터 모드

여러 줄의 소스 코드를 입력한 후 한 번에 실행하는 모드

1.3. 기초 문법

1.3.1. ;의 유무

파이썬에서는 ;를 써도 되고 안 써도 됨.
두 개 이상의 명령을 하나의 줄에 작성할 때는 명령들을 구분하기 위해 사용할 수 있음.

1.3.2. 따옴표

파이썬에서는 큰따옴표와 작은따옴표를 구분하지 않음.
단, 여는 따옴표와 닫는 따옴표는 동일한 것이어야 함.

1.3.3. 주석 (설명문)

실행되지는 않지만 효율적인 표현을 위한 문법.
: 한 줄 주석. (c에서의 //와 방식은 동일)
“”” . . . “”” : 여러 줄 주석. 따옴표를 세 개 연달아 사용함. (c에서의 /* . . . */와 방식은 동일)

1.3.4. 자료형

파이썬에서는 기본적으로 자료형을 명시하지 않음. 자동으로 결정됨.

1.4. 모듈

1.4.1. 모듈

c언어의 #include와 유사한 문법. 라이브러리의 모듈을 가져와서 그 안에 있는 함수를 사용할 수 있음. 그 형식은 아래와 같음.

```
import <모듈>
import <모듈> as <string>
from <모듈> import <함수 이름>
```

1.4.2. 모듈 예시

time
time.sleep() : 인자에 작성한 수 만큼 프로그램을 잠깐 멈추는 함수. (단위는 sec임)

random
random.randint(a, b) : a부터 b까지의 정수 중 하나를 리턴함. (Random int)

winsound (windows sound)
winsound.Beep(a, b) : 소리 출력 함수. a는 소리의 높이이고, b는 출력되는 시간임. (시간의 단위는 밀리초임)

2. 기본 문법

2.1. 연산자

c에서와 동일한 원리로 산술 연산자와 배정 연산자를 줄여 복합 배정 연산자로쓸 수 있음.
c에서와는 달리 파이썬에서는 ++과 –연산자를 지원하지 않음.

2.1.1. 산술 연산자

+ : 더하기
- : 빼기
: 곱하기
/ : 실수 나누기 -> 실수 나누기에 정수만을 쓸 수도 있음.
// : 정수 나누기
% : 나머지 연산
* : 거듭제곱

2.1.2. 배정 연산자

= 을 사용함.
배정 연산자의 왼쪽에는 단 하나의 순수한 변수만 올 수 있음.

2.1.3. 비교 연산자 (관계 연산자)

c와 동일함.
단, 파이썬에서는 부등식을 더 편리하게 사용할 수 있음.
80 <= x < 90 과 같이 작성하면 수학에서처럼 취급할 수 있음.
즉, x가 80보다 크거나 같고 90보다 작으면 참임.
== : 두 값이 같으면 참.
!= : 두 값이 다르면 참.
>, < : 부등식이 참이면 참.
>=, <= : 부등식이 참이면 참.

2.1.4. 논리 연산자

and : 두 조건이 모두 잠이어야 참. -> c에서 &&
or : 두 조건 중 하나라도 참이면 참. -> c에서 ||
not : 조건이 거짓이면 참. -> c에서처럼 식의 왼쪽에 작성함.

2.2. 변수

2.2.1. 변수

변수 선언을 따로 하지는 않고 대입할 때부터 명시함. c에서처럼 좌측에는 단 하나의 변수만 들어갈 수 있음.
그 형태는 아래와 같음.

```
<변수 이름> = <값>
```

2.2.2. 변수 이름 규칙

1. 숫자, 영어(대소문자 구분), 언더라인(_) 사용 가능
2. 영어 이외의 언어들도 사용 가능
3. 숫자로 시작 불가능
4. 공백 불가능
5. 이미 지정된 키워드는 사용 불가능
6. 언더라인(_)을 제외한 특수문자는 사용 불가능

2.3. 함수

2.3.1. 함수 정의

함수 이름은 변수 이름을 정하는 규칙과 같음.
파라미터가 없을 경우 그냥 비워 두면 됨.
반환값이 필요 없는 경우 return은 생략해도 됨.
정의 시 파라미터에 작성하는 변수에 값을 배정해주면 해당 값이 default값이 됨.
호출 시에 아무것도 작성하지 않으면 해당 값이 사용됨.
만약 함수 정의 시에 배정하지 않았는데 인자를 부족하게 작성한 경우, 오류 발생함.
호출 시에 인자를 파라미터 개수보다 적게 작성하면 인자의 값은 앞쪽 파라미터부터 들어감.
c에서처럼 파이썬에서도 호출 전에 함수의 존재를 알 수 있어야 하므로, 함수는 코드의 위쪽에 작성 해야 함.
작성 형식은 아래와 같음.

```
def <이름>(<파라미터>) :
명령문1
명령문2
return <결과값>
```

2.3.2. main 함수

파이썬에는 main 함수가 따로 없음. 그냥 빈 공간에다가 작성하면 main처럼 취급됨.

2.4. 조건문

2.4.1. if 조건문

조건이 참인 경우 명령을 실행하는 조건문.
오른쪽에 명시한 if의 형식을 지키면서 c에서와 동일하게 else와 else if를 사용할 수 있음. else는 그대로 else로, else if는 줄여서 elif로 표현함.
if 와 else, elif는 같은 레벨이어야 함. -> 들여쓰기 상에서 일치해야 의도대로 기능함.
아무 명령도 작성하지 않으면 오류 발생함. 대신 pass 키워드를 작성하면 됨.

```
if <조건> :
명령문1
명령문2
```

2.5. 반복문

2.5.1. while 반복문

명시한 조건이 참이면 루프를 돌아 명령들을 수행하는 반복문.
아무 명령도 작성하지 않으면 오류 발생함. 대신 pass 키워드를 작성하면 됨.

```
while <조건> :
    명령문1
    명령문2
```

2.5.2. for 반복문

list의 각 원소들이 순서대로 for 옆의 변수에 배정되며 루프가 도는 반복문.
변수에는 일반적인 변수를 작성함.
들여쓰기로 포함되는 명령들을 구분함. (tab키)
아무 명령도 작성하지 않으면 오류 발생함. 대신 pass 키워드를 작성하면 됨.

```
for <변수> in <list>:
    명령문1
    명령문2
```

2.6. 기본 함수들

2.6.1. print()

파이썬의 기본 출력 함수.
쉘 모드에서는 문자열, 숫자, 변수, 수식만 입력해도 print 함수를 생략한 것으로 취급되어 적용됨. 코드 에디터 모드에서는 print를 생략할 수 없음.

1. 사용 방법
괄호 안에 문자열, 숫자, 변수, 수식 등을 작성하여 출력할 수 있음.
문자열은 따옴표 안의 문자열이 출력됨.
숫자, 변수, 수식은 해당하는 값을 출력함.
여러 개의 인자를 쉼표로 구분하여 작성할 수 있음.
여러 줄에 걸쳐서 문자열을 작성하는 경우 따옴표를 세 개 사용함.

```
print(‘‘<문자열>’’)
print(<내용>)
```

2. Escape 문자
c에서와 동일한 기능을 함.
\n : 개행.
\t :탭.
\’ : 작은따옴표를 그대로 출력
\” : 큰따옴표를 그대로 출력
\\: 역슬래시를 그대도 출력
\b : 백스페이스. (역방향으로 한 글자 지움) -> IDLE 쉘 모드에서는 실행이 안 될 수도 있음.

3. 끝 문자 지정 (end 파라미터)
print의 마지막 문자를 지정할 수 있음.
end 파라미터는 맨 마지막 인자에 작성해 줘야 함.
“”과 같이 아무것도 작성하지 않으면 아무것도 안 들어감.
default값은 \n임. (print로 어떤 것을 출력하면 마지막에는 자동으로 개행됨)

4. 구분 문자 지정 (sep 파라미터)
쉼표 자리에 들어가는 문자를 지정할 수 있음.
sep 파라미터는 맨 마지막 인자에 작성해 줘야 함.
“”과 같이 아무것도 작성하지 않으면 아무것도 안 들어감.
default값은 공백 문자 하나임. (print에서 쉼표로 이으면 사이에 빈칸 하나가 자동으로 들어감)

5. 양식문자
c에서와는 달리, 파이썬에서는 양식문자를 모든 문자열에서 사용 가능함.
문자열의 바로 오른쪽에 양식문자에 들어갈 값을 명시함.
print 함수에서 유용함.
%d : 정수. 자릿수와 사용 방식을 결정하는 특정 형식 이 있음. (c와 동일)
%f : 실수. 자릿수와 사용 방식을 결정하는 특정 형식 이 있음. (c와 동일)
%g : 정수, 실수 자동결정
%s : 문자열
%c : 문자
%o : 8진 정수
%x : 16진 정수

하나의 양식문자 사용 : 양식문자에 들어갈 값을 % 로 구분하여 지정함.
2개 이상의 양식문자 사용 : 양식문자에 들어갈 값들을 %로 구분하여 지정함. 이때 값들은 괄호로 묶고, 쉼표로 구분함.

2.6.2. input()

파이썬의 기본 입력 함수.
문자, 문자열로 입력받음.
input()만 입력하면 키보드를 통해 입력 받고 값은 사용하지 않음.
입력 받은 값을 변수에 저장하려면 input()을 해당 변수에 배정해줘야 함.
괄호 안에 문자열, 숫자, 변수, 수식을 작성해서 프롬프트 기능을 사용할 수 있음.
이때 문자열은 따옴표로 묶어야 됨.
프롬프트를 사용해서 값을 입력 받고 변수에 저장해도, 입력 받은 값 만을 저장함. (프롬프트는 저장 안함)

2.6.3. 형변환 함수

1. int()
형을 정수로 변환하는 함수 아스키 값 등으로 변환하는 것이 아니라, 해당 텍스트와 동일한 형태의 정수로 변환함. (“3” -> 3)
x = int(input()) 등으로 사용함.

2. float()
형을 실수로 변환하는 함수
아스키 값 등으로 변환하는 것이 아니라, 해당 텍스트와 동일한 형태의 실수로 변환함. (“3.14” -> 3.14)
x = float(input()) 등으로 사용함.

3. 데이터 처리

3.1. list 자료형

3.1.1. list 자료형

수열(list)을 만드는 방법은 단순 명시, list(), range()로 세 가지가 있음. (수식까지 하면 4가지)
list는 출력 시 대괄호([])를 씌워서 출력됨.
list 안에 list가 원소로 들어갈 수도 있음.
list에는 문자열, 소수, 정수 등의 값이 들어갈 수 있음.
하나의 list에 다른 자료형의 값이 들어갈 수도 있음.

3.1.2. list 생성

1. 단순 명시
별다른 형식 없이 명시하는 것만으로도 list를 생성할 수 있음.
각괄호 안을 비워 두면 비어 있는 list 가 만들어짐.

2. list ()
수열 생성 함수.
list()의 파라미터로 아무것도 작성하지 않으면 비어 있는 list가 만들어짐.
(list() 만 작성)

3. range()
특정 패턴을 갖는 수열(리스트)을 만드는 함수.
각 인자에는 정수를 적음. -> 실수는 적을 수 없음.
인자의 순서는 c언어의 for와 유사함.
시작값부터 시작해서 종료값보다 작은 값까지 증가값만큼 커지는 수열을 생성함.
종료값은 포함되지 않음.
시작값을 생략하면 default로 0이 적용됨.
증가값을 생략하면 default로 1이 적용됨.
종료값은 생략할 수 없음.
이를 이용해서 range(<숫자>) 형식으로 자주 쓰임.
인자를 두 개만 작성하면 각각 앞부터 적용되어 시작값과 종료값으로 취급됨.

3.1.3. 인덱스

인덱스를 사용하여 변수에 배정한 list의 여러 원소 중 하나를 특정하여 사용하거나 배정 수 있음. c의 배열처럼 인덱스는 0부터 시작함.

```
<변수이름>[<인덱스>]
```

3.1.4. list에 데이터 추가하기

1. append()
list의 맨 뒤에 항목을 추가하는 함수.
변수 이름에 list를 배정한 변수의 이름을 작성함.
값에 추가할 값을 작성함.

```
<변수이름>.append(<값>)
```

2. insert()
list 중간에 항목을 추가하는 함수.
작성한 값이 list의 중간에 끼워지게 됨.
명시한 인덱스 위치에 값이 들어가고, 기존에 저장되어 있던 값들은 해당 위치부터 한 칸 씩 뒤로 밀림.
위치로 인덱스를 명시함.
slicing과 동일한 인덱스 적용 방식을 가짐.

```
<변수이름>.insert(<위치> <값>)
```

3. extend()
두 개의 list을 연결하는 함수.
해당 변수의 값을 확장하는 것.
변수1에 저장된 list의 뒤에 변수2에 저장된 list가 연결되어 변수1에 저장됨.

```
<변수1>.extend(<변수2>)
```

3.1.5. list의 데이터 지우기

1. <변수>.remove(<값>)
해당 변수에 저장되어 있는 list에서 인자에 작성한 값을 찾아서 제거하는 함수.
해당 값이 2개 이상인 경우, 앞의 값만 제거됨.

2. <변수>.pop(<인덱스 값>)
해당 변수에 저장되어 있는 list에서 인자에 작성한 인덱스 값에 해당하는 위치의 값을 제거하는 함수.

3.2. 데이터 분석

Pandas를 사용한 데이터 분석.

3.2.1. Pandas

데이터 분석을 위한 라이브러리.
1. 데이터를 프레임화(일반화)하는 기능을 가짐.
2. 대용량 데이터를 빠르게 자동 분석하고 처리할 수 있게 함.
아나콘다를 설치하면 함께 자동 설치됨.

```
import pandas as pd
```

3.2.2. 데이터 준비하기

1. 데이터 관련 함수들
pd.read_excel('<파일 경로> ')
엑셀 파일을 불러와서 DatatFrame 자료형으로 데이터를 리턴하는 함수.
csv 파일을 불러와서 데이터를 리턴하는 함수.
맨 뒤 인자에 header=none을 작성. 데이터의 첫 줄에 칼럼명(헤더)이 존재하지 않음을 의미.
맨 뒤 인자에 delimiter='<값>'을 작성. 해당 값으로 데이터들을 구분함.

```
pd. read\_csv('<파일 경로>')
```

2. 파일 경로의 사용
경로는 해당 파일을 우클릭해 '속성' 메뉴로 알 수 있음.
속성에는 해당 파일이 존재하는 폴더까지만 표시되기 때문에 '/<파일>.<확장자>'를 추가로 작성해줘야 함.
파이썬에서 사용할 때는 \기호를 /로 바꿔 줘야 함.
앞에 있는 c:도 그대로 붙여 줘야 함.

3.2.3. 데이터 출력하기

seaborn 등의 데이터도 이 함수들로 출력이 가능함.

<변수>.info() : 해당 변수에 저장된 데이터에 대한 상세정보를 출력하는 함수.
<변수>.describe() : 해당 변수에 저장된 데이터 중 분석이 가능한 데이터 의 통계를 출력하는 함수.
<변수>.head() : 데이터의 앞쪽 레코드들만 출력하는 함수. 기본 5개, 매개변수에 정수 입력하면 해당 개수만큼 출력함.
<변수>.tail() : 데이터의 뒤쪽 레코드들만 출력하는 함수. 기본 5개, 매개변수에 정수 입력하면 해당 개수만큼 출력함.

3.3. 데이터 시각화

seaborn, matplotlib를를 활용한 데이터 시각화.

3.3.1. seaborn

사용 가능한 함수들을 정리함.

1. sns.relplot (data=<변수명>, x='<칼럼명>', y='<칼럼명>');
작성한 변수명에 저장된 데이터에서, 두 칼럼 사이의 관계를 여러 요소들을 고려하여 scatter(기본) 차트를 생성하는 함수.
데이터에 해당 칼럼이 존재해야 함.
; 를 작성하여 출력을 완료함. (plt.show()와 동일한 기능.)
함수는 내부적으로 matplotlib을 사용하기 때문에, matplotlib을 import하여 plt.show()를 사용하는 것이 일반적임.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
1. hue='<칼럼명>' : 해당 칼럼을 기준으로 색을 나눔.
2. style='<칼럼명>' : 해당 칼럼을 기준으로 marker를 임의의 모양으로 구분함.

3. size='<칼럼명>' : 해당 칼럼을 기준으로 marker를 임의의 크기로 구분함.
4. kind='<차트 종류>' : 지정한 종류로 차트를 생성함. 지정하지 않는다면 scatter 차트가 생성됨. line을 작성하면 line 차트를 생성함. 차트 종류는 따옴표로 씌워서 지정해야 함. (문자열임.)

2. sns.lineplot (data=<변수명>, x='<칼럼명>', y='<칼럼명>');
작성한 변수명에 저장된 데이터에서, 두 칼럼 사이의 관계를 line 차트로 생성하는 함수.

3. displot (<변수명>.<칼럼명>)
칼럼 하나의 분포를 차트로 생성하는 함수.
.<칼럼명> 표기법은 약식이므로, 띄어쓰기 등이 있을 때는 ['<칼럼명>'] 형식으로 작성해야 함.
query() 함수 등을 사용해 특정 값 근처만 출력할 수도 있음.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
1. bins=<정수> -> 생성할 막대의 개수를 지정함.
; 를 작성하여 출력을 완료함. (plt.show()와 동일한 기능.)
함수는 내부적으로 matplotlib을 사용하기 때문에, matplotlib을 import하여 plt.show()를 사용하는 것이 일반적임.

4. sns.countplot(<변수명>.<칼럼명>)
범주형 데이터를 막대 차트로 생성하는 함수.
한 번에 하나의 칼럼만을 넣을 수 있음. barplot() 함수를 사용하면 두 개의 칼럼을 사용할 수 있음.
; 를 작성하여 출력을 완료함. (plt.show()와 동일한 기능.)
함수는 내부적으로 matplotlib을 사용하기 때문에, matplotlib을 import하여 plt.show()를 사용하는 것이 일반적임.

5. sns.barplot(data=<변수명>, x='<칼럼명>', y='<칼럼명>')
해당 변수에 저장된 범주형 데이터 하나와 수치형 데이터 하나로 막대 차트를 생성하는 함수. x에는 범주형 데이터를, y에는 수치형 데이터를 지정해야 함.
; 를 작성하여 출력을 완료함. (plt.show()와 동일한 기능.)
함수는 내부적으로 matplotlib을 사용하기 때문에, matplotlib을 import하여 plt.show()를 사용하는 것이 일반적임.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
1. orient='<형태>' : 막대 차트를 수평 또는 수직으로 생성하도록 지정함. h는 수평, v는 수직(기본)임. 수평으로 지정할 경우, x, y 위치가 바뀌기 때문에 반대로 지정해 줘야 함.

6. sns.boxplot(data=<변수명>, x='<칼럼명>')
해당 변수에 저장된 수치형 데이터 하나의 분포를 박스형 차트로 생성하는 함수.
x축만을 지정하면 하나의 박스를 생성함.
x, y축을 지정하면 여러 개의 박스를 생성함.
박스 가운데 세로선은 중위값(2사분위)을 나타냄,
박스의 왼쪽, 오른쪽 경계선은 각각 25%(1사분위), 75%(3사분위)를 나타냄.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
1. orient='<형태>' : 막대 차트를 수평 또는 수직으로 생성하도록 지정함. h는 수평(기본), v는 수직임.
2. hue='<칼럼명>' : 해당 칼럼을 기준으로 색을 나눔.

7. <변수명>.query('<조건>')
해당 변수에 저장된 데이터 중 조건에 맞는 값을 가진 데이터들만 모아서 리턴하는 함수.
조건은 "<칼럼명> < 10" 이런 식으로 지정할 수 있음.

8. sns.pairplot(<변수명>)
작성한 변수명에 저장된 데이터에 있는 모든 칼럼을 사용하여 여러 개의 차트로 표시하는 함수.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
1. hue='<칼럼명>' : 해당 칼럼을 기준으로 색을 나눔.

9. sns.set_style('<배경>')
차트의 배경 스타일을 지정하는 함수.
배경 종류
white : 흰색 배경.
dark : 검은색 배경.
grid : 색깔 뒤에 작성하여 grid를 추가할 수 있음. (ex. whitegrid, darkgrid)

3.3.2. matplotlib

사용 가능한 함수들을 정리함.

1. plt.plot(<데이터>, <데이터>)
차트 생성 함수.
기본적으로 선형 그래프가 생성됨.
매개변수에는 데이터를 작성함. (list나 pandas를 사용해서 생성한 데이터 묶음 등.)
데이터 묶음을 하나만 제공하면 y축 값으로만 들어가고 x축 값은 자동 설정됨.
데이터 묶음을 두 개 제공하면 첫 번째 인자의 데이터가 x값으로, 두 번째 인자의 데이터가 y값으로 들어감.
기본적으로 하나의 plt.plot() 함수에서는 하나의 데이터 묶음만 처리가 가능함.
여러가지 값들을 개별적으로 그래프로 그리려면 여러 개의 plot() 함수를 사용해야 함. 개별적인 차트가 생성되고, 출력은 한꺼번에 됨.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
이것들은 맨 오른쪽 매개변수에 작성함.
1. marker='<marker>' -> 차트의 선형 그래프의 나타난 값에 도형을 추가하는 옵션.
2. markersize=<size> -> 차트 marker의 크기를 설정함.
3. color='<색>' -> 차트의 색깔을 설정함.
4. label='<범례>' -> 차트의 범례를 설정함. (각 그래프가 어떤 값을 나타내는 것인지 이름 설정.) plt.legend() 함수로 출력해줘야 함.
5. linestyle='<형태>' -> 선의 스타일을 설정함. 따옴표 안에 아무것도 작성하지 않으면 선이 표시되지 않음.
6. linewidth=<넓이> -> 선의 두께를 설정함.
color, marker, linestyle은 축약형 표기 사용 가능. '<color><marker><linestyle>'을 인자로 작성함. (ex. plt.plot(data, 'ro–'))

2. plt.bar(<데이터>, <데이터>)
두 데이터로 막대그래프를 생성하는 함수.
plt.plot()과 plt.bar()를 동시에 사용하여 선과 막대를 한꺼번에 출력할 수도 있음.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
1. alpha=<정수> -> 투명도를 지정함. 1이 가장 진한 값.

3. plt.pie(<데이터>, labels=<데이터>)
두 데이터로 pie 차트를 생성하는 함수.
첫 번째 인자로 들어가는 데이터는 값이고, 두 번째 인자 labels에 들어가는 데이터는 해당 값의 이름임.

4. plt.scatter(<데이터1>, <데이터2>)
scatter 차트 생성 함수.
scatter 차트는 점을 찍어서 나타내는 그래프임.
x축의 데이터와 y축의 데이터를 분리해서 제공해야 함.
x축의 데이터(numpy)는 첫 번째 인자에, y축의 데이터(numpy)는 두 번째 인자에 작성함.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
1. c=<데이터> : 데이터(numpy)로 정수 수열을 넣으면 수에 해당되는 색이 각각의 점에 부여됨.
2. s=<데이터> : 데이터(numpy)로 정수 수열을 넣으면 수에 해당되는 크기가 각각의 점에 부여됨.

5. plt.legend()
plt.plot()에서 작성한 label을 출력하는 함수.
이 함수를 수행해야 label이 출력됨.
plt.plot()에서 label을 지정하지 않으면 이 함수를 사용할 수 없음.
인자로 아무것도 작성하지 않으면 자동으로 위치를 선정해 출력함.
loc='<문구>'를 인자로 작성하면 해당하는 위치에 출력함.
문구 종류는 오른쪽 그림에 나와 있음.
plt.plot() 함수에서 사용 가능.

6. plt.xlabel('<x축>') / plt.ylabel('<y 축>')
plot의 x, y축 이름을 작성하는 함수.
plt.plot() 함수에서 사용 가능.

7. plt.show()
차트 출력 함수.

실행 전, 명령 묶음의 제일 마지막에 작성해야 함.
이 함수 대신 ;를 써도 되기는 함.

8. plt.grid()
차트에 격자무늬 출력 함수.
매개변수로는 아무것도 작성하지 않아도 됨. (true 값으로 반영됨.)
plt.plot() 함수에서 사용 가능.

9. plt.title('<제목>')
차트 제목 출력 함수.
한글 제목을 작성하면 오류 발생함.
1. plt.rc('font', family='Gothic')
2. plt.rc('font', family='NanumGothic')
3. plt.rc('font', family='Malgun Gothic')
이때 세가지 중 하나의 명령을 사용해야 한글 제목을 사용할 수 있음.
plt.plot() 함수에서 사용 가능.

10. plt.savefig('<파일명>')
주피터 노트북의(주피터 노트북 사용 시.) 기본 폴더에 출력한 차트를 저장하는 함수.
단순히 마우스 오른쪽 키로 저장하면 사진 품질이 떨어짐.

11. plt.xlim(<시작>, <끝>) / plt.ylim(<시작>, <끝>)
ㄷ plot의 x, y값의 출력 범위를 정하는 함수.
plt.plot() 함수에서 사용 가능.

12. plt.xticks(<데이터>, <데이터>)
데이터로 제공한 list 값 만을 x축의 눈금으로 출력하는 함수.
plt.plot(), plt.bar() 함수에서 사용 가능. 근데 왜 데이터를 2개 주지?

13. plt.figure(figsize=(<숫자1>, <숫자2>))
차트 출력 시 크기를 지정하는 함수.
숫자1에는 가로, 숫자2에는 세로를 지정함.
seaborn, matplotlib 모두에 사용이 가능함.
차트 생성 코드 앞쪽에 작성해야 함.
해당 위치를 확보해 두고, 차트를 그 안에 그림.

14. plt.annotate(<문자열>, xy=(<x좌표>, <y좌표>), xytext=(<x좌표>, <y좌표>))
차트에 화살표를 그려서 문자열을 표시하는 함수.
문자열에는 출력할 문자열을 지정함. (직접 작성하는 것이면 따옴표 씌워서.)
xy에는 화살표가 가리킬 좌표를 지정함.
xytext에는 문자열이 출력될 좌표를 지정함.
한글 문자열을 출력하면 오류 발생함.
1. plt.rc('font', family='Gothic')
2. plt.rc('font', family='NanumGothic')
3. plt.rc('font', family='Malgun Gothic')
이때 세가지 중 하나의 명령을 사용해야 한글 문자열을 사용할 수 있음.
고급 옵션들. (맨 뒤 인자에 작성하는 것들.)
1. arrowprops=('color:'<색깔>') : 색깔 등 화살표의 속성 지정. 색깔로 r 등을 지정할 수 있음.

15. plt.imshow(<데이터>)
인자의 데이터로 이미지 차트를 생성하는 함수.
plt.plot() 함수와 동일하게, 출력 시에는 plt.show() 함수를 호출하는 것이 기본임.
word cloud를 생성한 후 출력할 때 사용함.

16. plt.axis('off')
차트의 축을 설정하는 함수.
인자로 'off'를 지정하여 축을 지울 수 있음. (wordcloud에서 사용.)

3.4. 데이터 연산

numpy를 이용한 데이터 연산.

3.4.1. numpy

과학적 계산을 위한 외부 라이브러리.

```
import numpy as np
```

3.4.2. Array 생성

1. np.arange(<시작값>, <종료값>, <증가값>)
numpy의 수열 생성 함수.
range() 함수와 사용 방법은 동일함.
특징1. range() 함수의 인자에는 소수를 작성할 수 없지만, np.arange() 함수에서는 가능함.
특징2. np.arrange() 함수로 생성한 수열을 저장한 변수를 사칙 연산하면 각 원소에 개별적으로 사칙 연산한 것으로 취급됨.

2. np.array(<list>)
인자에 작성한 list로 numpy array를 형성하는 함수.

3. np.linspace(<시작값>, <종료값>, <구분값>)
시작값부터 종료값까지의 값을 구분값만큼의 개수로 구분하여 수열을 생성하는 함수.
값을 구분값만큼의 개수로 등분하는 것임.
시작값과 종료값을 모두 포함하여 수열을 생성함.

3.4.3. 랜덤 값 생성

1. np.random.randint(<숫자1>, <숫자1>, <개수>)
숫자1과 숫자2 사이의 정수 중에서 작성한 개수만큼 무작위 값을 추출하여 수열을 생성하는 함수.

2. np.random.normal(<숫자>, <표준편차>, <개수>)
해당 숫자를 기준으로 표준편차 이내에 있는 값 중에서 작성한 개수만큼 무작위 값을 추출하여 수열을 생성하는 함수.