

성신여자대학교
지역사회 공헌 프로그램

고교생 SW교육

쉽게 배우는

파이썬



성신여자대학교
SUNGSHIN WOMEN'S UNIVERSITY



University Innovation

대학혁신지원사업

CONTENTS



DAY1

1-1 파이썬 코딩으로 Spirograph 그리기

1-2 파이썬 코딩으로 영문학 작품에서
가장 긴 단어 찾기

DAY2

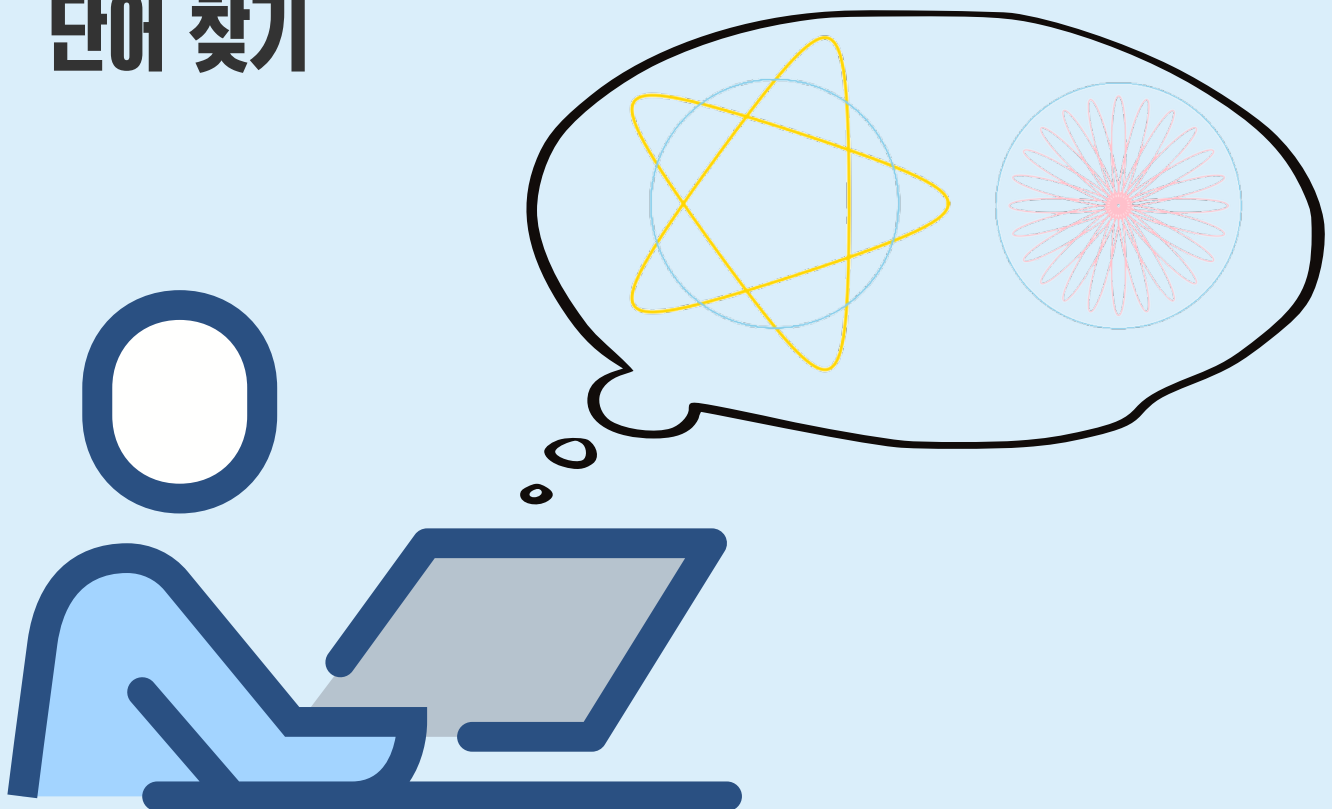
2-1 파이썬을 이용한 미분의 활용 시각화

2-2 파이썬을 이용한 Word Cloud 만들기

DAY1

1-1 파이썬 코딩으로 Spirograph 그리기

1-2 파이썬 코딩으로 영문학 작품에서 가장 긴 단어 찾기



Python 코딩으로 Spirograph 그리기

1. Spirograph 란?

반지름이 R (정수) 인 고정된 원 둘레를 따라 바깥쪽에서 반지름이 r (유리수) 인 위성 원이 돌 때, 위성 원의 중심으로부터 거리 d (유리수) 의 위치에 연필을 꽂아서 회전시켜 그린 도형

<https://nathanfriend.io/inspiral-web/>

2. 파이썬 (python) 코드 문법 소개

먼저, 파이썬(python)으로 간단하게 약수, 최대공약수, 최소공배수를 구해보고,

이어서 최소공배수를 활용하여 spirograph 를 그리는 코드를 작성해보자.

최대공약수와 최소공배수 구하기

```
In [ ]: n=12

i=1
while i <= n:
    if n % i == 0: print(i)
    i += 1
```

```
In [ ]: n=12

for i in range(1, n+1):      # range(a,b) 는 a 이상 b 미만
    if n % i == 0: print(i)
```

```
In [ ]: def GCD(m,n):
        i=min(m,n)
        while i > 0:
            if m % i == 0 and n % i == 0: return(i)
            i -= 1
        return 1
```

```
In [ ]: GCD(12,18)
```

```
In [ ]: def LCM(m,n):
        i=m
        j=n
        while True:
            if i==j: return i          # m, 2m, 3m, ... 과 n, 2n, 3n, ... 을 비교하여 최초로 일치하는 수가 최소공
            elif i<j: i += m
            else: j += n
```

```
In [ ]: print(LCM(12,54))
        print(LCM(12,54)/12)
        print(LCM(12,54)/54)
```

3. Epitrochoid 그리기

고정된 원 둘레를 따라 바깥쪽에서 위성 원이 돌 때 그려지는 spirograph (epitrochoid)

<https://en.wikipedia.org/wiki/Epitrochoid>

```
In [ ]: import matplotlib.pyplot as plt
        from numpy import pi, sin, cos, linspace

        def spiroE(t, R, r, d):
            return [(R+r)*cos(t)-d*cos(t*(R+r)/r), (R+r)*sin(t)-d*sin(t*(R+r)/r)]

        def circle(t, r):
```

```

    return [r*cos(t), r*sin(t)]

R = 1.0          # R 의 값은 정수로 부여
p = 5
q = 7
s = 4
r = p/q          # r, d 의 값은 유리수로 부여
d = s/q

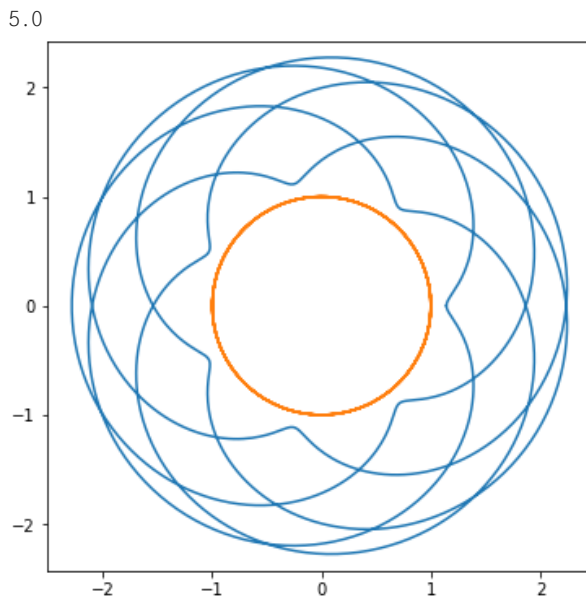
n = LCM(p,q)/q    # 위성원이 고정 원의 둘레를 몇 번 돌아야 닫힌 곡선이 완성되는가
print(n)

plt.figure(figsize=(6, 6))
plt.axis('equal')

t = linspace(0, n*2*pi, 1000)
plt.plot(spiroE(t, R, r, d)[0], spiroE(t, R, r, d)[1])
plt.plot(circle(t, R)[0], circle(t, R)[1])

plt.show()

```



```

In [ ]:
import matplotlib.pyplot as plt
from numpy import pi, sin, cos, linspace

def spiroE(t, R, r, d):
    return [(R+r)*cos(t)-d*cos(t*(R+r)/r), (R+r)*sin(t)-d*sin(t*(R+r)/r)]

def circle(t, r):
    return [r*cos(t), r*sin(t)]

R = 3          # R 의 값은 정수로 부여
p = 2
q = 1
s = 2
r = p/q        # r, d 의 값은 유리수로 부여
d = s/q

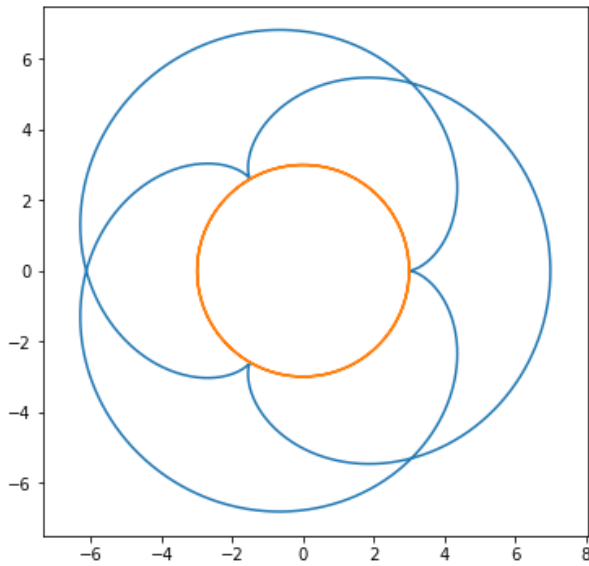
n = LCM(p,q)/q    # 위성원이 고정 원의 둘레를 몇 번 돌아야 닫힌 곡선이 완성되는가
print(n)

plt.figure(figsize=(6, 6))
plt.axis('equal')

t = linspace(0, n*2*pi, 1000)
plt.plot(spiroE(t, R, r, d)[0], spiroE(t, R, r, d)[1])
plt.plot(circle(t, R)[0], circle(t, R)[1])

plt.show()

```



4. Hypotrochoid 그리기

고정된 원 둘레를 따라 **안쪽**에서 위성 원이 돌 때 그려지는 spirograph (hypotrochoid)

<https://en.wikipedia.org/wiki/Hypotrochoid>

In []:

```
import matplotlib.pyplot as plt
from numpy import pi, sin, cos, linspace

def spiroH(t, R, r, d):
    return [(R-r)*cos(t)+d*cos(t*(R-r)/r), (R-r)*sin(t)-d*sin(t*(R-r)/r)]

def circle(t, r):
    return [r*cos(t), r*sin(t)]

R = 1.0          # R 의 값은 정수로 부여
p = 52
q = 96
s = 42
r = p/q          # r, d 의 값은 유리수로 부여
d = s/q

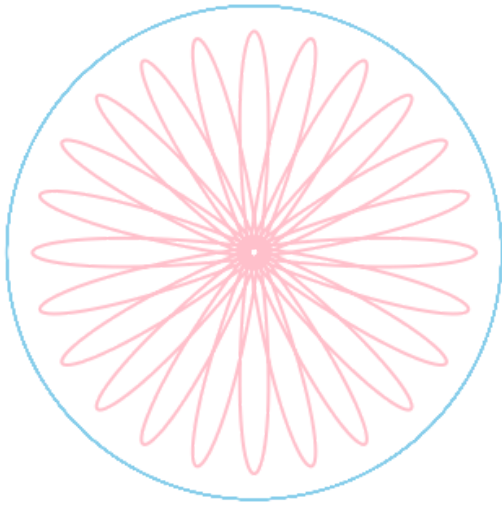
n = LCM(p,q)/q    # 위성원이 고정 원의 둘레를 몇 번 돌아야 닫힌 곡선이 완성되는가
print(n)

plt.figure(figsize=(6, 6))
plt.axis('equal')

t = linspace(0, n*2*pi, 1000)
plt.plot(spiroH(t, R, r, d)[0], spiroH(t, R, r, d)[1], 'pink', linewidth=2)
plt.plot(circle(t, R)[0], circle(t, R)[1], 'skyblue', linewidth=1)

plt.axis('off')
plt.show()
```

13.0



파이썬 그래프에 다양한 색을 사용할 수 있습니다. 색의 이름을 다음 사이트에서 참고하세요.

<https://www.tutorialexample.com/list-of-matplotlib-common-used-colors-matplotlib-tutorial/>

In []:

```
import matplotlib.pyplot as plt
from numpy import pi, sin, cos, linspace

def spiroH(t, R, r, d):
    return [(R-r)*cos(t)+d*cos(t*(R-r)/r), (R-r)*sin(t)-d*sin(t*(R-r)/r)]

def circle(t, r):
    return [r*cos(t), r*sin(t)]

R = 3          # R 의 값은 정수로 부여
p = 2
q = 1
s = 2
r = p/q        # r, d 의 값은 유리수로 부여
d = s/q

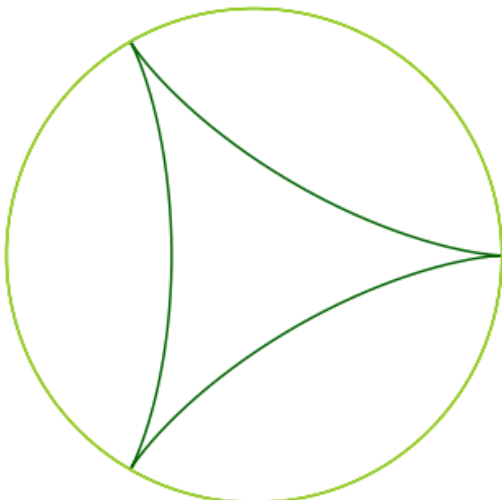
n = LCM(p,q)/q  # 위성원이 고정 원의 둘레를 몇 번 돌아야 닫힌 곡선이 완성되는가
print(n)

plt.figure(figsize=(6, 6))
plt.axis('equal')

t = linspace(0, n*2*pi, 1000)
plt.plot(spiroH(t, R, r, d)[0], spiroH(t, R, r, d)[1], 'darkgreen')
plt.plot(circle(t, R)[0], circle(t, R)[1], 'yellowgreen')

plt.axis('off')
plt.show()
```

2.0



```

In [ ]: import matplotlib.pyplot as plt
        from numpy import pi, sin, cos, linspace

        def spiroH(t, R, r, d):
            return [(R-r)*cos(t)+d*cos(t*(R-r)/r), (R-r)*sin(t)-d*sin(t*(R-r)/r)]

        def circle(t, r):
            return [r*cos(t), r*sin(t)]

        R = 5          # R 의 값은 정수로 부여
        p = 3
        q = 1
        s = 5
        r = p/q         # r, d 의 값은 유리수로 부여
        d = s/q

        n = LCM(p,q)/q  # 위성원이 고정 원의 둘레를 몇 번 돌아야 닫힌 곡선이 완성되는가
        print(n)

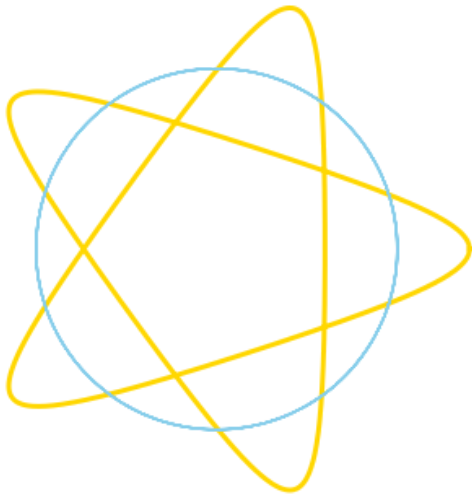
        plt.figure(figsize=(6, 6))
        plt.axis('equal')

        t = linspace(0, n*2*pi, 1000)
        plt.plot(spiroH(t, R, r, d)[0], spiroH(t, R, r, d)[1], 'gold', linewidth=3)
        plt.plot(circle(t, R)[0], circle(t, R)[1], 'skyblue')

        plt.axis('off')
        plt.show()

```

3.0



고교생 소프트웨어 - 쉽게 배우는 파이썬

01

Python 코딩으로 영문학 작품에서 가장 긴 단어 찾기

윤태진 교수
성신여자대학교 영어영문학과



강의안내



01

파이썬 소개

02

The longest English word?

03

Python Building Blocks

04

Alice's Adventures in Wonderland에서 가장 긴 단어는?



01

Python 소개

Guido van Rossum 귀도 반 로섬



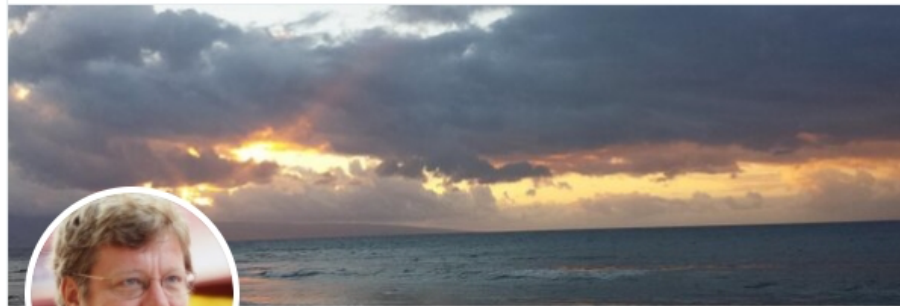
89

크리스마스 연휴

99

DARPA

Computer Programming for Everybody



팔로우

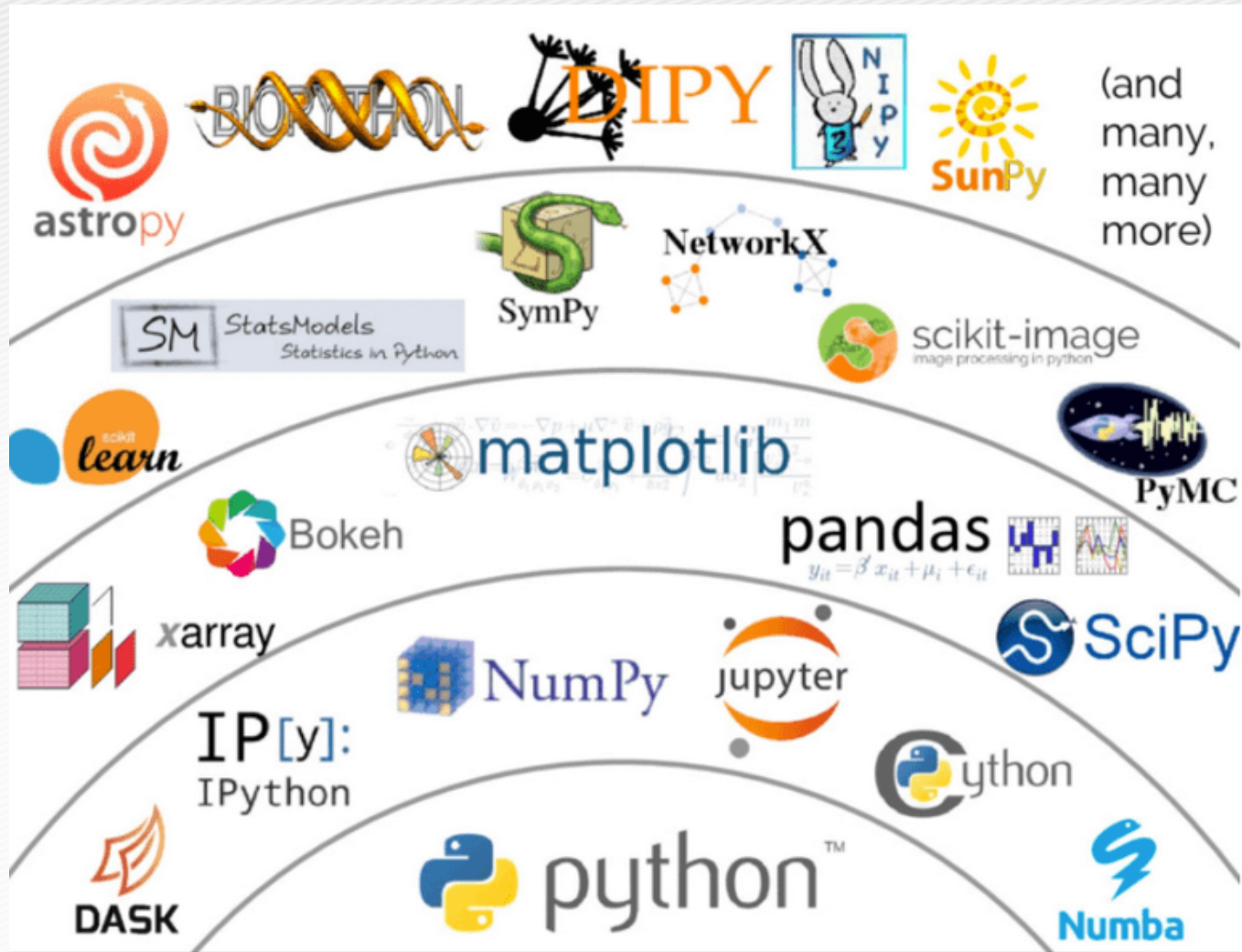
Guido van Rossum ✓

@gvanrossum

Python's BDFL-emeritus, Distinguished Engineer at Microsoft, Computer History Fellow. Opinions are my own. He/him.

📍 San Francisco Bay Area 🔗 python.org/~guido/ 📅 가입일: 2008년 8월

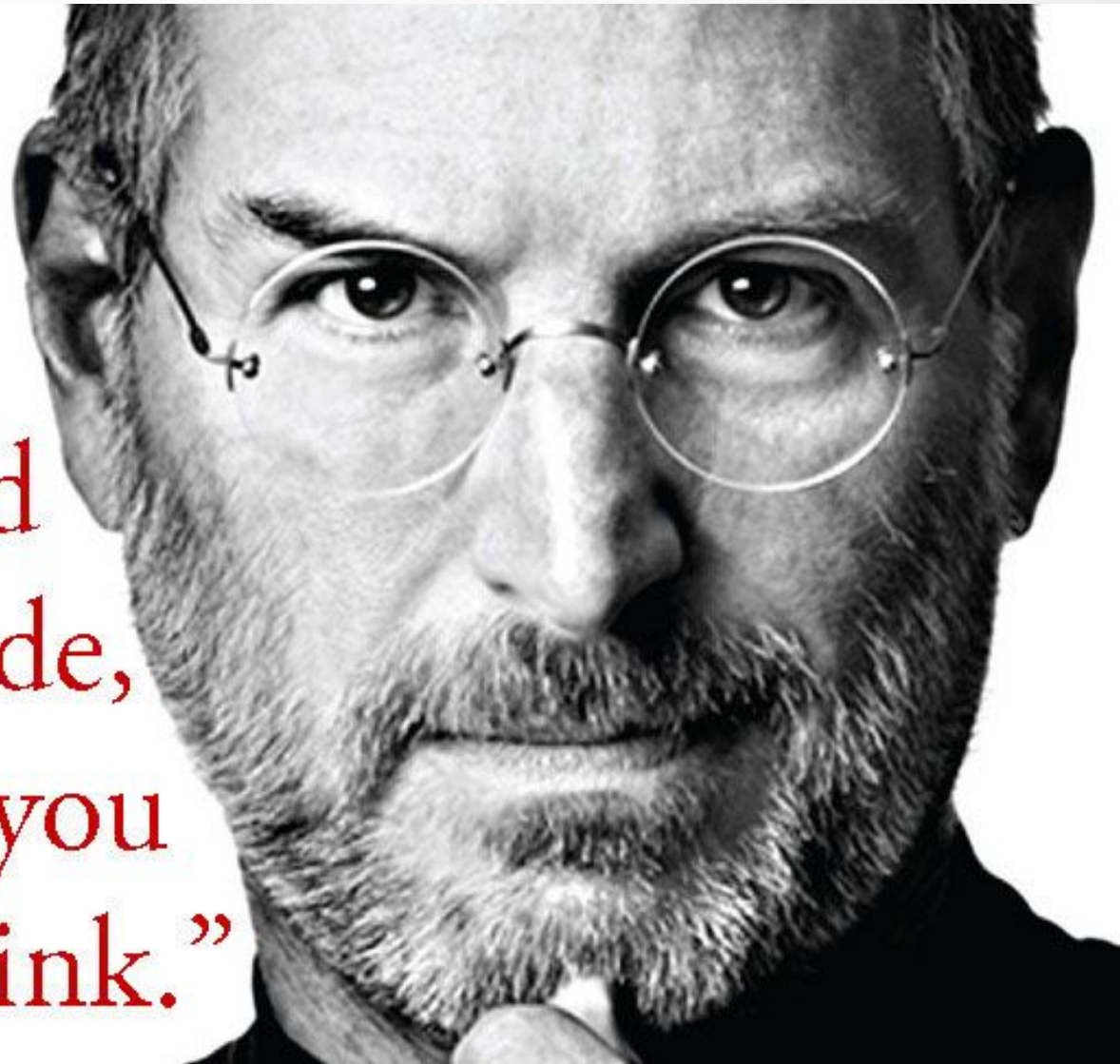
515 팔로우 중 19.9만 팔로워



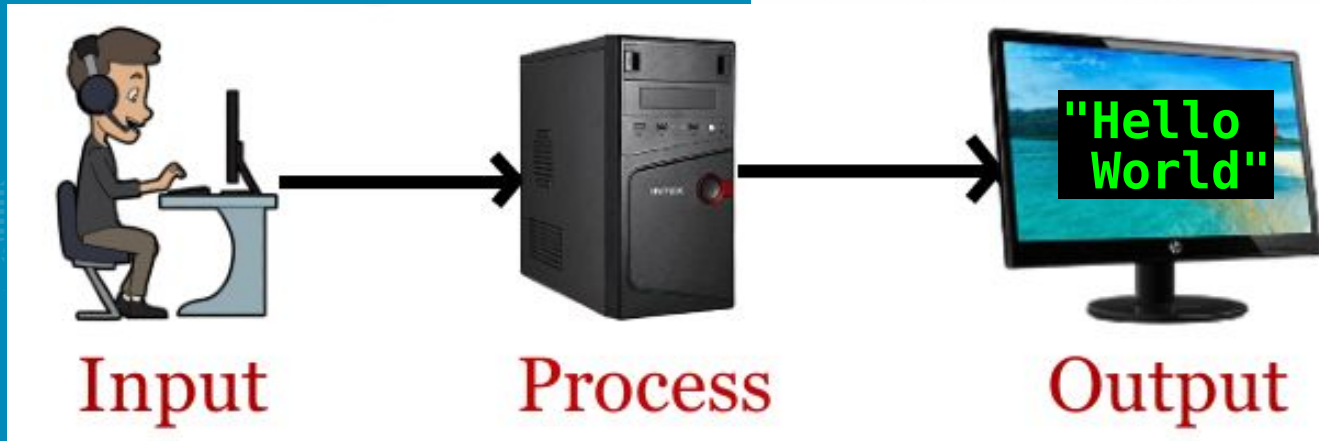
Steve Jobs

1955-2011

“Everyone should
learn how to code,
it teaches you
how to think.”



코딩 - What & why?



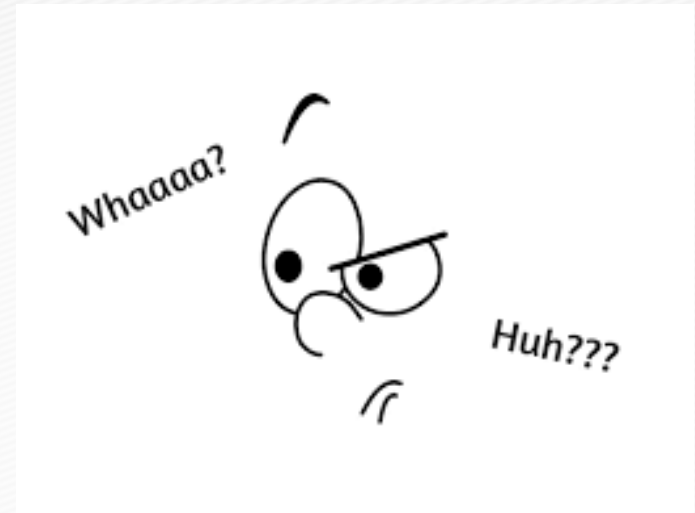
<https://www.educatecity.com/2020/06/what-are-computersr.html>





The longest English word?

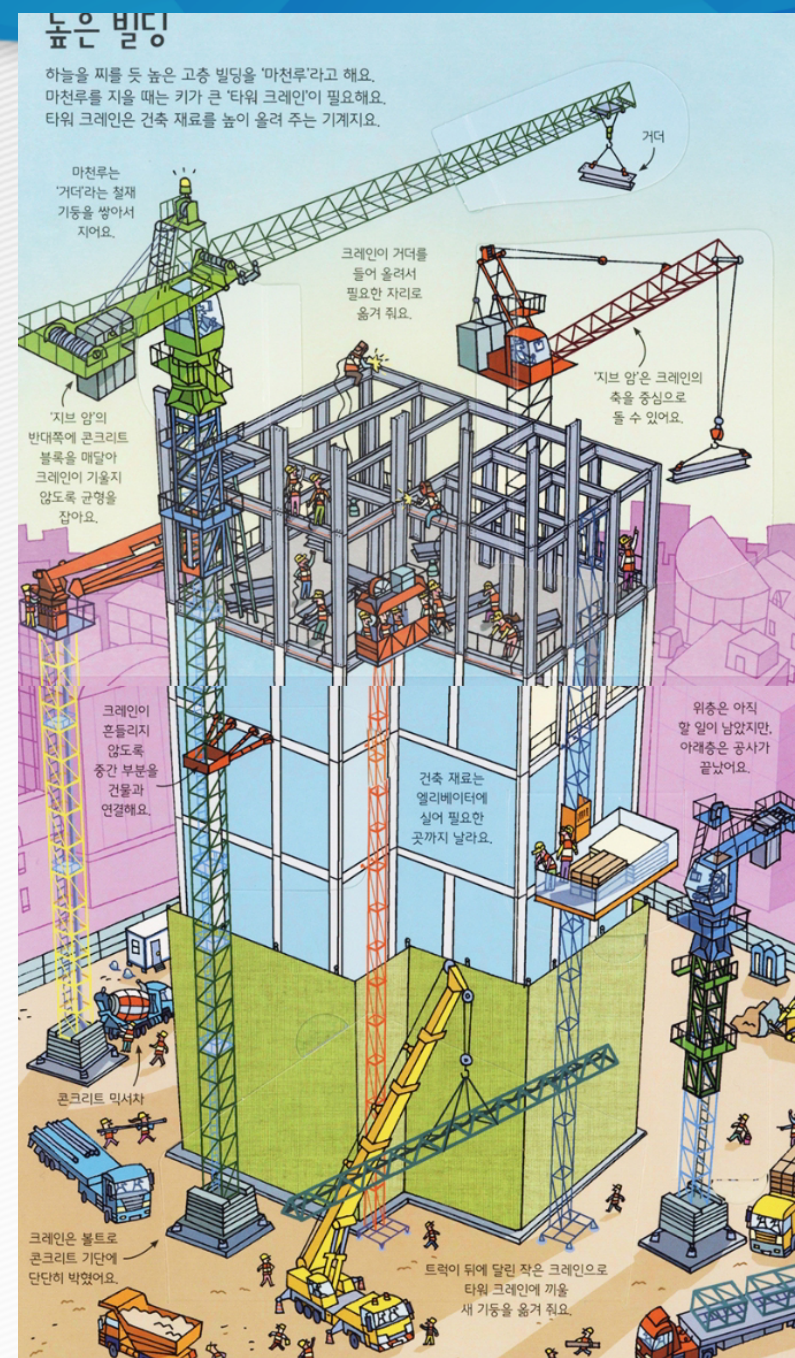
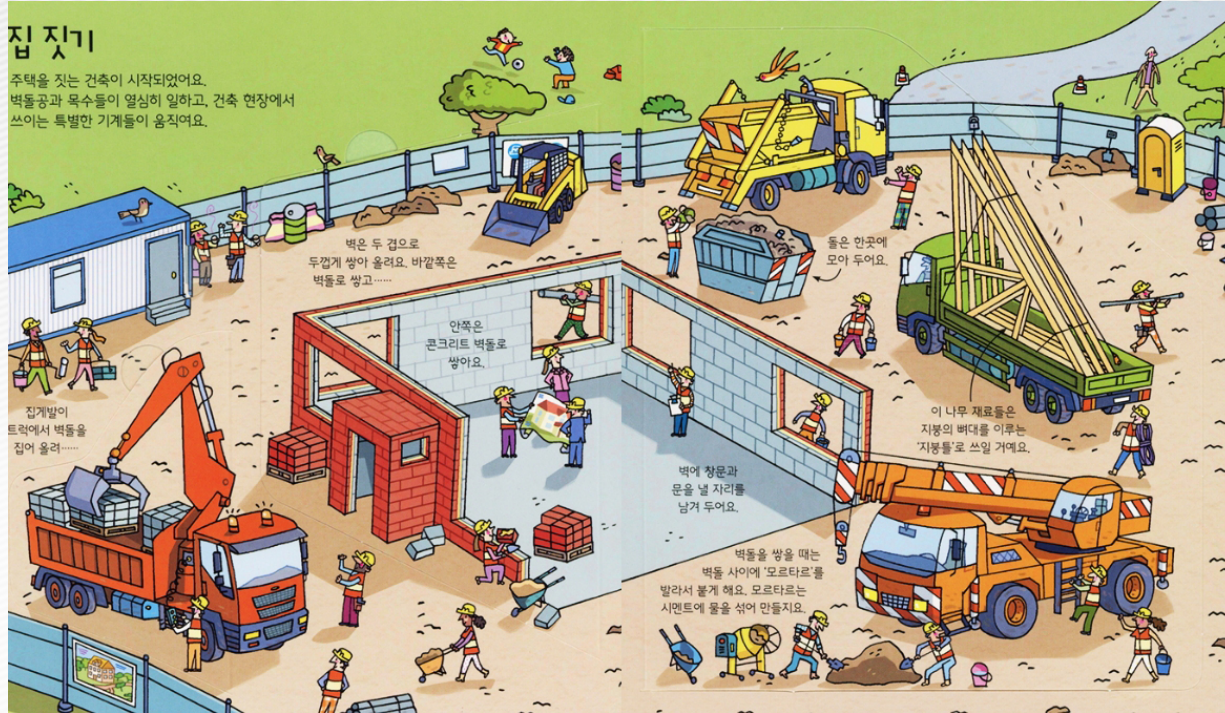
Pneumonoultramicroscopicsilico volcanoconiosis





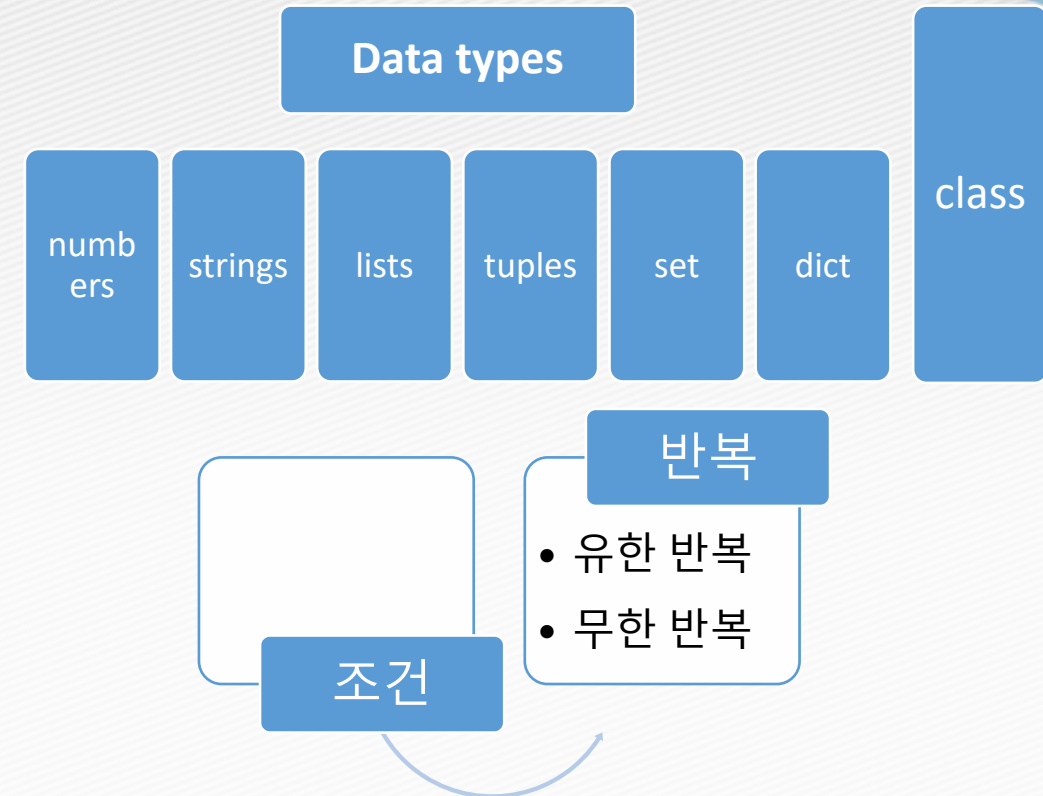
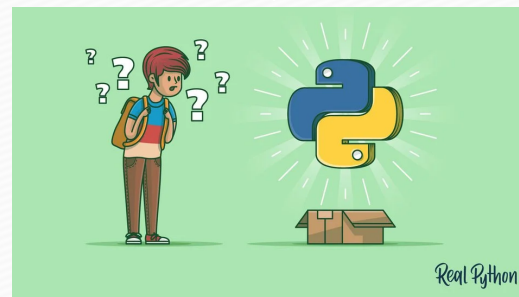
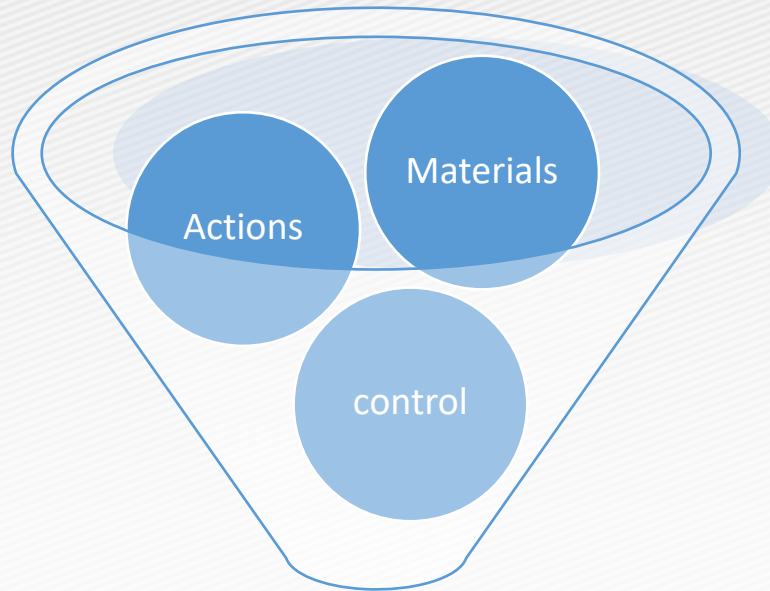
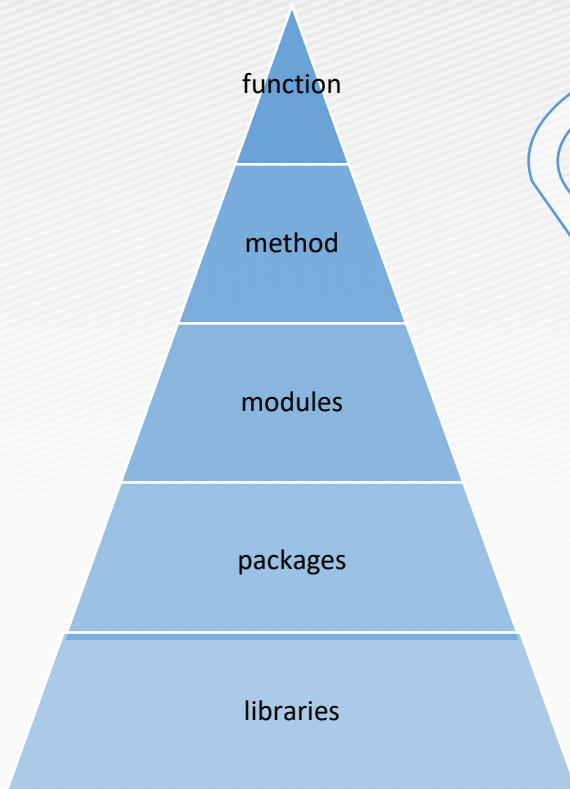
03

Python Building Blocks

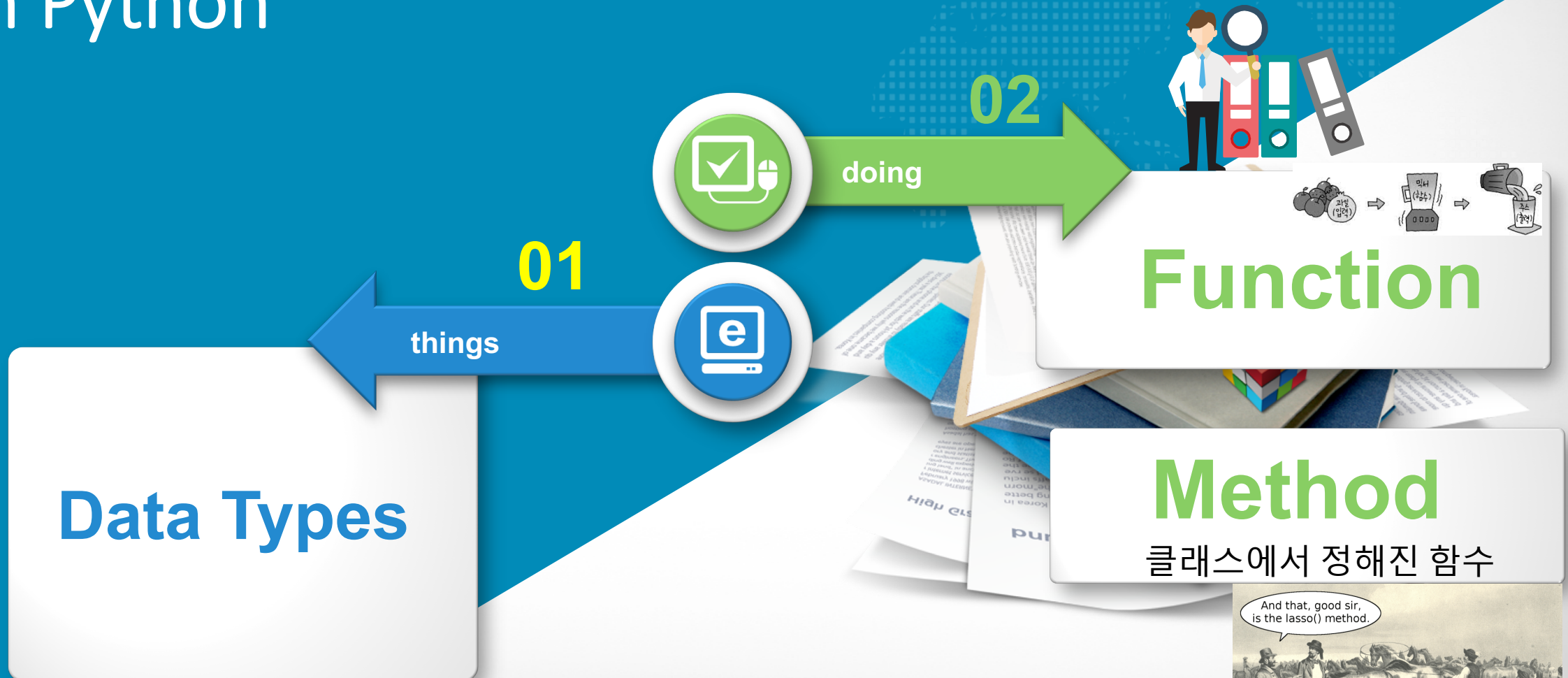


요리조리 열어 보는 집 짓는 기계들

• http://picturebook-museum.com/user/book_detail.asp?idx=17363



How to do things with Python



len(string) string.split()

- len(): 문자열, 리스트 등의 길이를 알려주는 함수
- split(): 문자열 양쪽에 불필요한 문자를 떼어 내어 리스트(list)를 만드는 메소드

공백(space), \t, \n, \r

len(' Pneumonoultramicroscopicsilicovolcanoconiosis')

결과

'사과 배 포도 오렌지 감'.split()

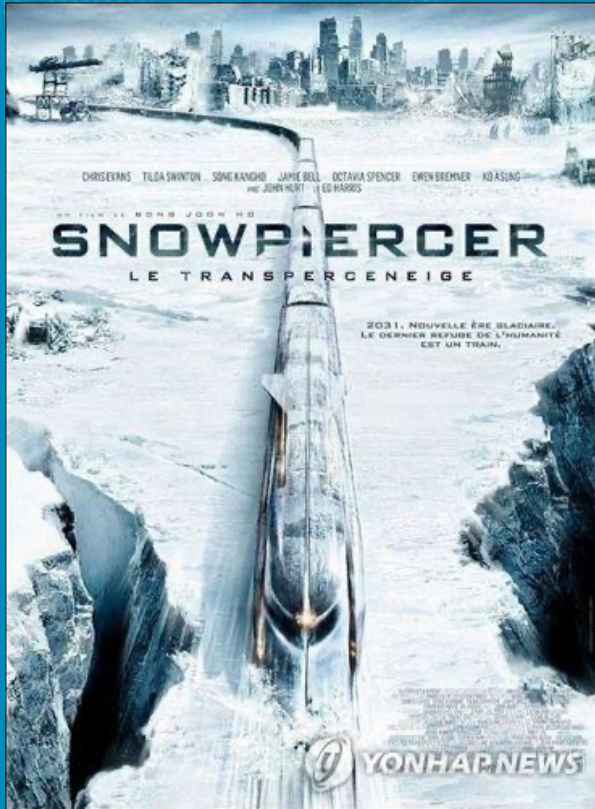
결과 ['사과', '배', '포도', '오렌지', '감']

'사과, 배, 포도, 오렌지, 감'.split()

결과 ['사과,', '배,', '포도,', '오렌지,', '감']

'사과, 배, 포도, 오렌지, 감'.split(',')

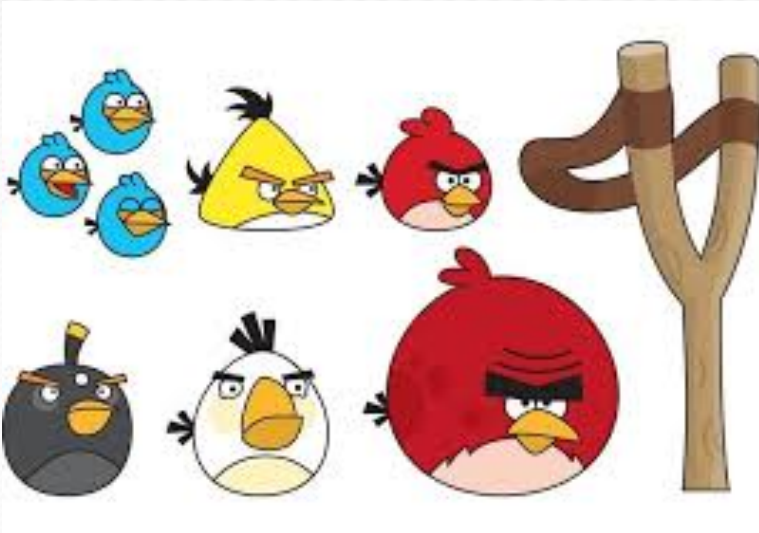
결과 ['사과', ' 배', ' 포도', ' 오렌지', ' 감']



반복문

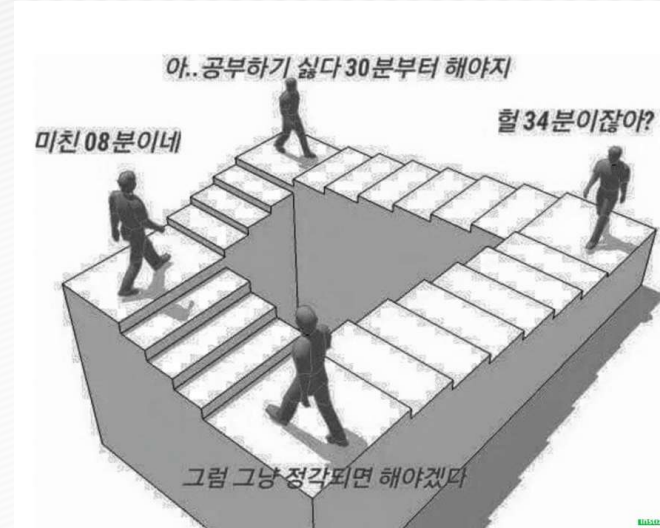
- for loop
- while loop

유한 loop



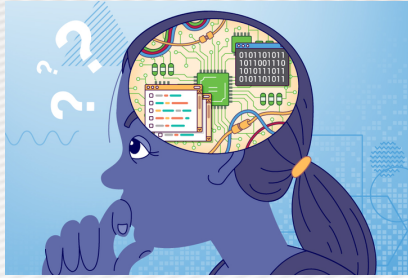
Angry Birds

무한 loop



Penrose Stairs

리스트에서 가장 긴 단어 찾기



```
words = ['hello', 'computer', 'bookshelves', 'chair', 'bicycle']
```

```
longest = len(words[0])
```

```
longest_word = words[0]
```

```
for word in words[1:]:
```

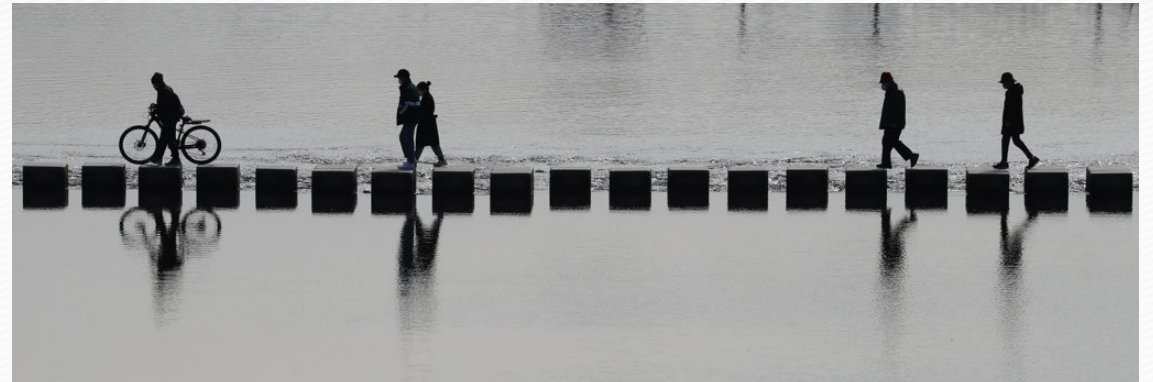
```
    if len(word) > longest:
```

```
        longest_word = word
```

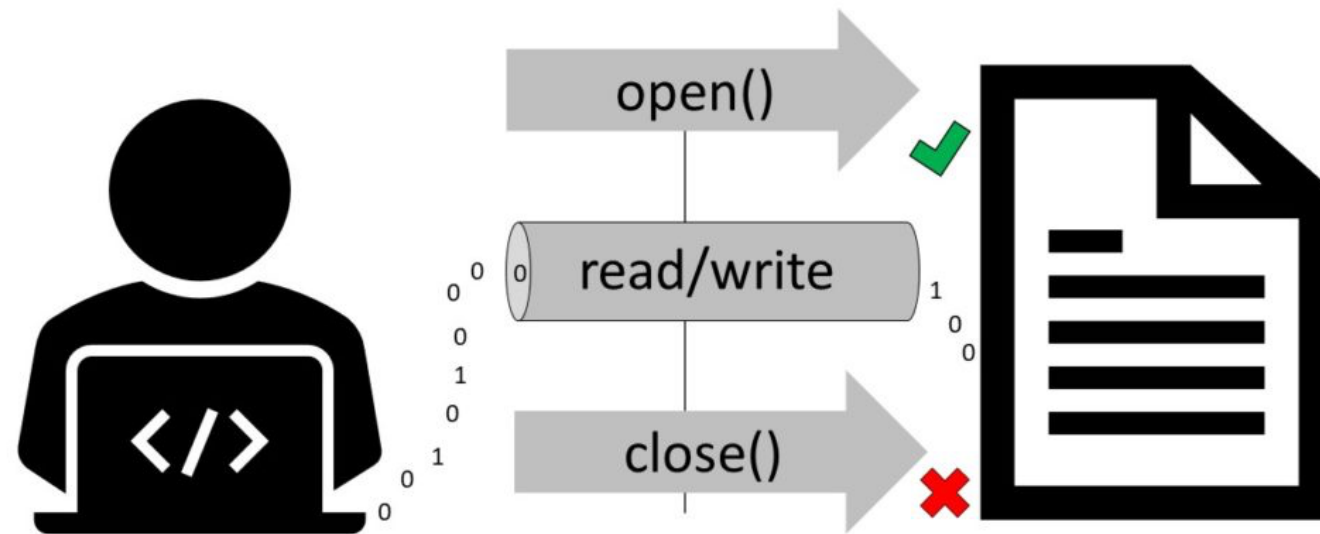
```
        longest = len(word)
```

```
print("Longest word:", longest_word)
```

```
print("Length of the word:", longest)
```



파일 입출력 과정



```
f = open('alice.txt')  
text = f.read()  
f.close()  
print(text)
```




04

Alice's Adventures in Wonderland에서 가장 긴 단어는?



```
f = open('alice.txt')
text = f.read()
f.close()
words = text.split()

longest = len(words[0])
longest_word = words[0]

for word in words[1:]:
    if len(word) > longest:
        longest_word = word
        longest = len(word)
print("Longest word:", longest_word)
print("Length of the word:", longest)
```

THANK YOU

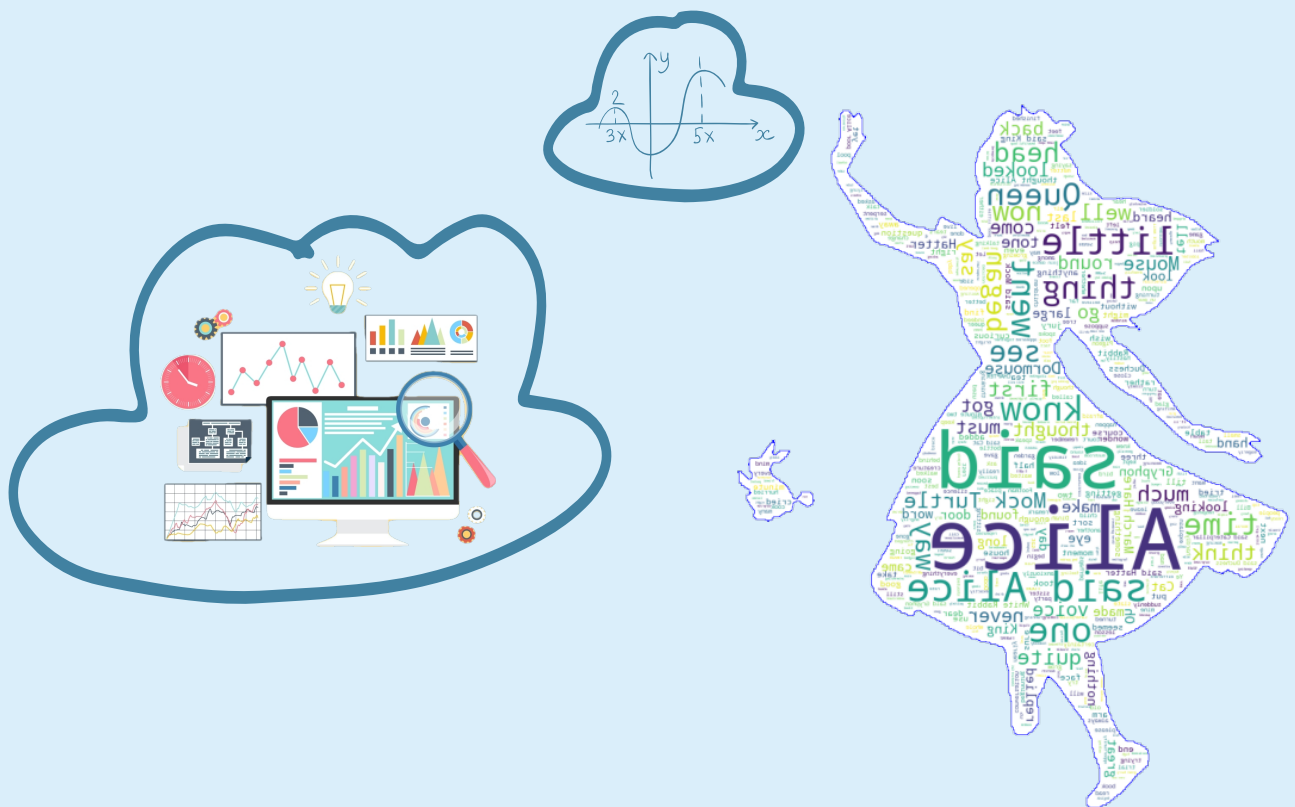




DAY 2

2-1 파이썬을 이용한 미분의 활용 시각화

2-2 파이썬을 이용한 Word Cloud 만들기



파이썬을 이용한 미분의 활용 시각화

1. 함수의 그래프의 개형 그리기

함수의 극값

x_0 를 포함하는 열린 구간이 존재하여, 이 구간의 모든 점 x 에 대하여 $f(x_0) \geq f(x)$ 를 만족할 때,

f 는 x_0 에서 **극댓값(relative maximum)** 을 갖는다고 하고, $(x_0, f(x_0))$ 를 **극대점**이라고 한다.

x_0 를 포함하는 열린 구간이 존재하여, 이 구간의 모든 점 x 에 대하여 $f(x_0) \leq f(x)$ 를 만족할 때,

f 는 x_0 에서 **극솟값(relative minimum)** 을 갖는다고 하고, $(x_0, f(x_0))$ 를 **극소점**이라고 한다.

극값은 극댓값 또는 극솟값을 뜻한다.

</br>

함수의 임계점과 정류점

함수 f 가 x_0 에서 미분불가능이거나 $f'(x_0) = 0$ 일 때, $(x_0, f(x_0))$ 를 **임계점(critical point)** 이라한다.

임계점 중에서 $f'(x_0) = 0$ 인 점, 즉 수평접선을 갖는 점 $(x_0, f(x_0))$ 를 **정류점(stationary point)** 이라 한다.

</br>

예제 1 함수 $f(x) = 3|x|^4 - 16|x|^3 + 30|x|^2 - 24|x|$ 의 그래프의 개형을 그리고, 임계점을 모두 찾으시오.

또 정류점, 극대점, 극소점을 모두 찾으시오.

[풀이]

$f(x) = 3|x|^4 - 16|x|^3 + 30|x|^2 - 24|x|$ 의 그래프는 y 축에 대하여 대칭이므로, $x \geq 0$ 인 범위에서의 그래프의 개형을 그려보자.

$x \geq 0$ 일 때, $f(x) = 3x^4 - 16x^3 + 30x^2 - 24x$ 이므로,

$f'(x) = 12x^3 - 48x^2 + 60x - 24 = 12(x-1)^2(x-2) = 0$ 은 $x = 1, x = 2$ 에서 성립한다.

그리고 $f(1) = -7, f(2) = -8$ 이다.

이 결과로 4차 다항함수의 그래프의 $f(x) = 3x^4 - 16x^3 + 30x^2 - 24x$ 를 $x \geq 0$ 범위에서 그린 후, y 축에 대칭으로 $x < 0$ 인 범위에서도 그려주면 다음과 같은 그래프를 얻는다.

따라서 임계점은 $(-2, -8), (-1, -7), (0, 0), (1, -7), (2, -8)$ 이다.

또 정류점은 $(-2, -8), (-1, -7), (1, -7), (2, -8),$

극대점은 $(0, 0),$

극소점은 $(-2, -8), (2, -8)$ 이다.

참고로, $(-1, -7), (1, -7)$ 은 변곡점이다.

[풀이 끝]

파이썬 예제 1 파이썬의 matplotlib.pyplot 모듈을 이용하여 앞의 예제 1의 함수의 그래프의 개형을 그려보자.

[풀이]

```
In [4]: import numpy as np
import matplotlib.pyplot as plt
from math import *

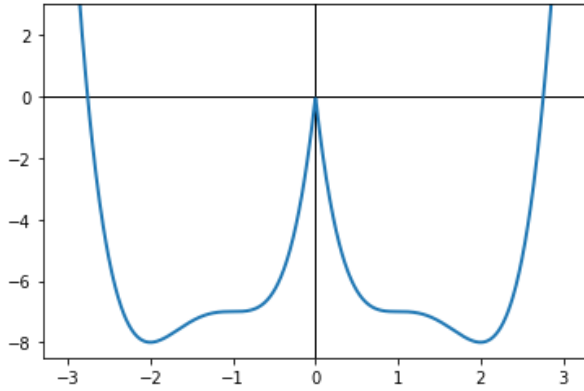
plt.figure(figsize=(6,4))
plt.axhline(0, color='black', linewidth=1)
plt.axvline(0, color='black', linewidth=1)

x=np.linspace(-3, 3, 1000)
plt.ylim(-8.5, 3)

y=3*abs(x)**4-16*abs(x)**3+30*abs(x)**2-24*abs(x)
```

```
plt.plot(x,y,linewidth=2)

plt.show()
```



[풀이 끝]

2. 뉴턴의 근사법(Newton's method)

뉴턴의 근사법은 $f(x) = 0$ 을 만족하는 근 $x = r$ 의 근삿값을 찾는 방법으로,

접선을 반복적으로 이용한다.

구하려는 실제 근에 가까운 값 x_0 를 f 의 정의역에서 하나 잡고 시작하여,

수열 $x_1, x_2, \dots, x_n, \dots$ 을 구성하여 r 로 수렴하게 만드는 방법으로 다음과 같은 점화식을 이용한다.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

이 점화식은 $y = f(x)$ 의 $x = x_n$ 인 점에서의 접선

$$y - f(x_n) = f'(x_n)(x - x_n)$$

이 x 축과 만나는, 즉 $y = 0$ 일 때의 x 좌표를 x_{n+1} 로 잡은 것이다.

예제 2 $f(x) = x^2 - 2 = 0$ 의 해의 근삿값을 구하기 위하여 뉴턴의 근사법을 이용하여 $x_0 = \frac{3}{2}$ 로 잡고 x_1, x_2 를 계산하시오.

[풀이]

$f'(x) = 2x$ 이므로

$$x - \frac{f(x)}{f'(x)} = x - \frac{x^2 - 2}{2x} = \frac{x}{2} + \frac{1}{x} \text{ 이다.}$$

$$\text{따라서 } x_1 = \frac{x_0}{2} + \frac{1}{x_0} = \frac{3}{4} + \frac{2}{3} = \frac{17}{12} \approx 1.4167,$$

$$x_2 = \frac{x_1}{2} + \frac{1}{x_1} = \frac{17}{24} + \frac{12}{17} = \frac{577}{408} \approx 1.41421568627451 \text{ 이다.}$$

[풀이 끝]

파이썬 예제 2 파이썬의 sympy, numpy 라이브러리에서 뉴턴의 근사법을 구현해보자.

근을 구하고자 하는 함수 f 의 식과 시작점을 입력하면 된다.

[풀이]

앞의 예제 2에서 살펴본 $f(x) = x^2 - 2$ 의 근을 구해보자. 시작점은 $x = 1.5$ 로 해보았다.

In []:

```
from sympy import *
import numpy as np

x = Symbol('x')
```

```

f = x**2-2          # 뉴턴의 근사법을 적용할 함수 입력

df = f.diff(x)

    # to transform sympy expressions to lambda functions which can be used to calculate numerical values very
f = lambdify(x,f)
df = lambdify(x,df)

x = 1.5              # 시작점 입력
print(x)

h = f(x)/df(x)

while abs(h) >= 0.000001:    # 오차의 한계를 원하는 값으로 설정
    h = f(x)/df(x)
    x = x - h                # x_(i+1) = x_i - f(x_i) / f'(x_i)
    print(x)

print("An approximate value of the root is : ", x)

```

```

1.5
1.4166666666666667
1.4142156862745099
1.4142135623746899
1.4142135623730951
An approximate value of the root is :  1.4142135623730951

```

[풀이 끝]

파이썬 예제 3 파이썬의 sympy, numpy 라이브러리와 matplotlib.pyplot 모듈을 이용해서 뉴턴의 근사법을 구현하고, 시각화해볼 수 있다.

원하는 함수와 시작점을 입력하여 실행시켜 보자.

[풀이]

먼저, $f(x) = \ln x - 1$ 를 입력하여 근의 근삿값을 구해보자. 시작점은 $x = 1$ 로 해보았는데, 다른 값으로 바꾸어 실행시켜 볼 수도 있다.

```

In [ ]:
import matplotlib.pyplot as plt
import numpy as np

from sympy import *

x = Symbol('x')

f = log(x)-1          # 뉴턴의 근사법을 적용할 함수 입력

f_to_print = str(f)

fig = plt.figure(num=1,figsize=(6,5),dpi=100)
ax = fig.add_subplot(1, 1, 1)
plt.xlabel('x')
plt.ylabel('y')
plt.title('Root of: ' + f_to_print)
plt.axhline(0, color='black', linewidth=1)
plt.axvline(0, color='black', linewidth=1)

xp = np.linspace(0.5,100)    # 그래프를 그리기 위해서 수치적 변수를 xp, yp 로 나타내었음
plt.ylim(-4, 3)              # 그래프를 그릴 리스트를 만들 때는 numpy 함수로 입력

yp = np.log(xp)- 1

plt.plot(xp,yp,color='blue')

df = f.diff(x)

    # to transform sympy expressions to lambda functions which can be used to calculate numerical values very
f = lambdify(x,f)
df = lambdify(x,df)

x_0 = 1                  # 시작점 입력
print(x_0)
plt.plot(x_0,0,color='red',marker='o',markersize=4)

x = x_0
y1= f(x)+df(x)*(xp-x)      # 첫번째 접선의 그래프

```

```
plt.plot(xp,y1,color='black', linewidth=1)

h = f(x)/df(x)
x=x-h
print(x)
plt.plot(x,0,color='red',marker='o',markersize=4)

y2 = f(x)+df(x)*(xp-x)      # 두번째 접선의 그래프
plt.plot(xp,y2,color='black', linewidth=1)

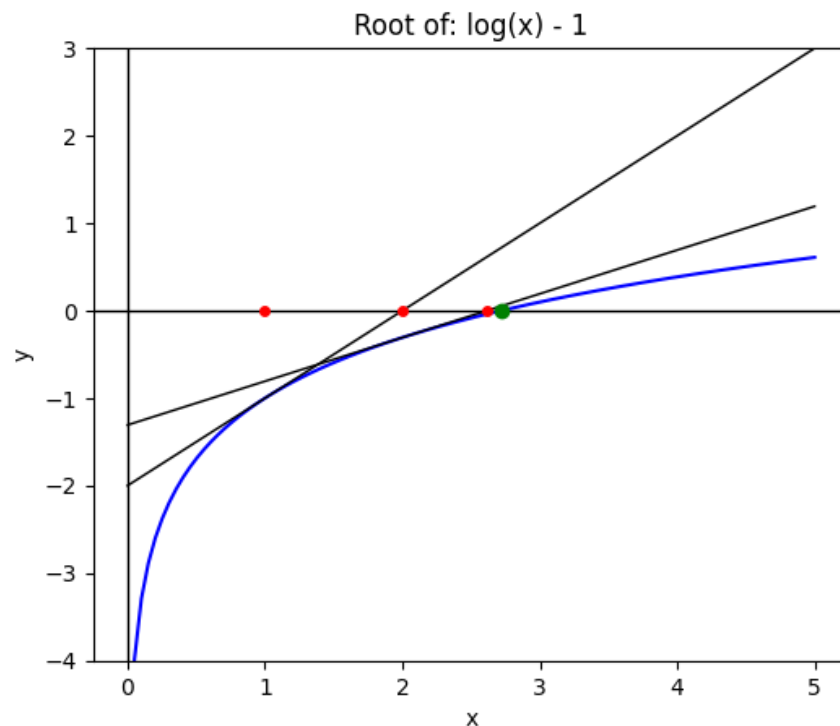
#list_x_vals=[]

while abs(h) >= 0.000001:    # 오차의 한계를 원하는 값으로 설정
    h = f(x)/df(x)
    x = x - h                # x(i+1) = x(i) - f(x) / f'(x)
    plt.plot(x,0,color='red',marker='o',markersize=4)
    print(x)

plt.plot(x,f(x),color='green',marker='o',markersize=6)

print("An approximate value of the root is : ",x)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:23: RuntimeWarning: divide by zero encountered in log
1
2.0
2.613705638880109
2.7162439263557903
2.718281064358138
2.7182818284589376
An approximate value of the root is : 2.7182818284589376
```



다음으로, $f(x) = \sin x$ 를 입력하여 근의 근삿값을 구해보자. 시작점은 $x = 4$ 로 해보았는데, 다른 값으로 바꾸어 실행시켜 볼 수도 있다.

```
In [3]: import matplotlib.pyplot as plt
import numpy as np

from sympy import *

x = Symbol('x')

f = sin(x)      # 뉴턴의 근사법을 적용할 함수 입력

f_to_print = str(f)

fig = plt.figure(num=1,figsize=(6,5),dpi=100)
ax = fig.add_subplot(1, 1, 1)
plt.xlabel('x')
plt.ylabel('y')
```



```

plt.title('Root of: ' + f_to_print)
plt.axhline(0, color='black', linewidth=1)
plt.axvline(0, color='black', linewidth=1)

xp = np.linspace(-0.5,5,100)      # 그래프를 그리기 위해서 수치적 변수를 xp, yp 로 나타내었음
plt.ylim(-2, 2)                  # 그래프를 그릴 리스트를 만들 때는 numpy 함수로 입력

yp = np.sin(xp)

plt.plot(xp,yp,color='blue')

df = f.diff(x)

# to transform sympy expressions to lambda functions which can be used to calculate numerical values very
f = lambdify(x,f)
df = lambdify(x,df)

x_0 = 4                          # 시작점 입력
print(x_0)
plt.plot(x_0,0,color='red',marker='o',markersize=4)

x = x_0
y1= f(x)+df(x)*(xp-x)             # 첫번째 접선의 그래프
plt.plot(xp,y1,color='black',linewidth=1)

h = f(x)/df(x)
x=x-h
print(x)
plt.plot(x,0,color='red',marker='o',markersize=4)

y2 = f(x)+df(x)*(xp-x)            # 두번째 접선의 그래프
plt.plot(xp,y2,color='black',linewidth=1)

#list_x_vals=[]

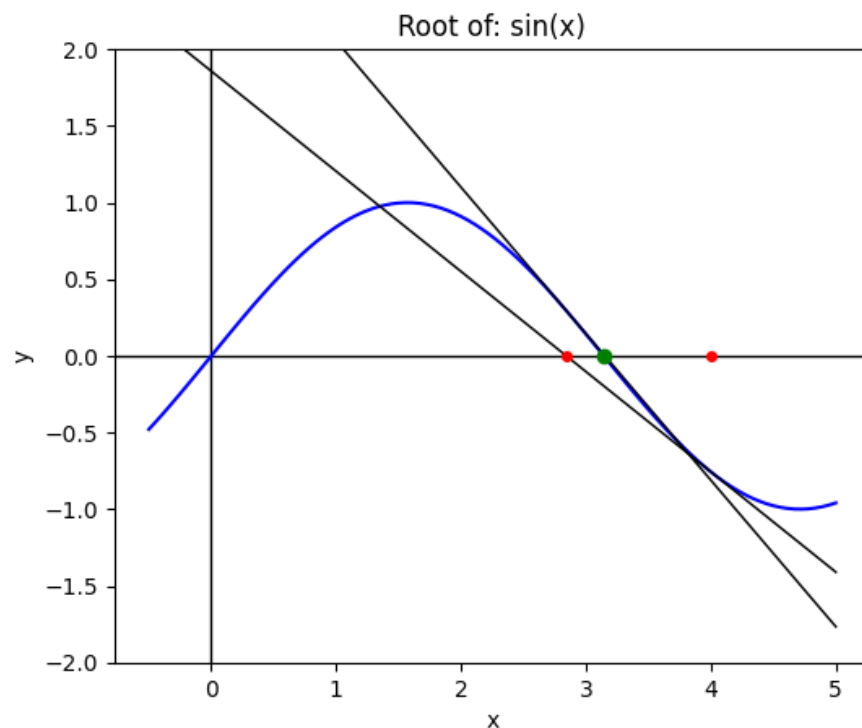
while abs(h) >= 0.000001:         # 오차의 한계를 원하는 값으로 설정
    h = f(x)/df(x)
    x = x - h                     #  $x(i+1) = x(i) - f(x) / f'(x)$ 
    plt.plot(x,0,color='red',marker='o',markersize=4)
    print(x)

plt.plot(x,f(x),color='green',marker='o',markersize=6)

print("An approximate value of the root is : ",x)

```

4
 2.8421787176504223
 3.1508729396853696
 3.1415923871630587
 3.141592653589793
 An approximate value of the root is : 3.141592653589793



고교생 소프트웨어 - 쉽게 배우는 파이썬

02

Python을 이용한 word cloud 만들기

윤태진 교수
성신여자대학교 영어영문학과



강의안내

01

영어 단어 중 빈도수가 가장 높은 것은?

02

Image as Numbers

03

Library & Class



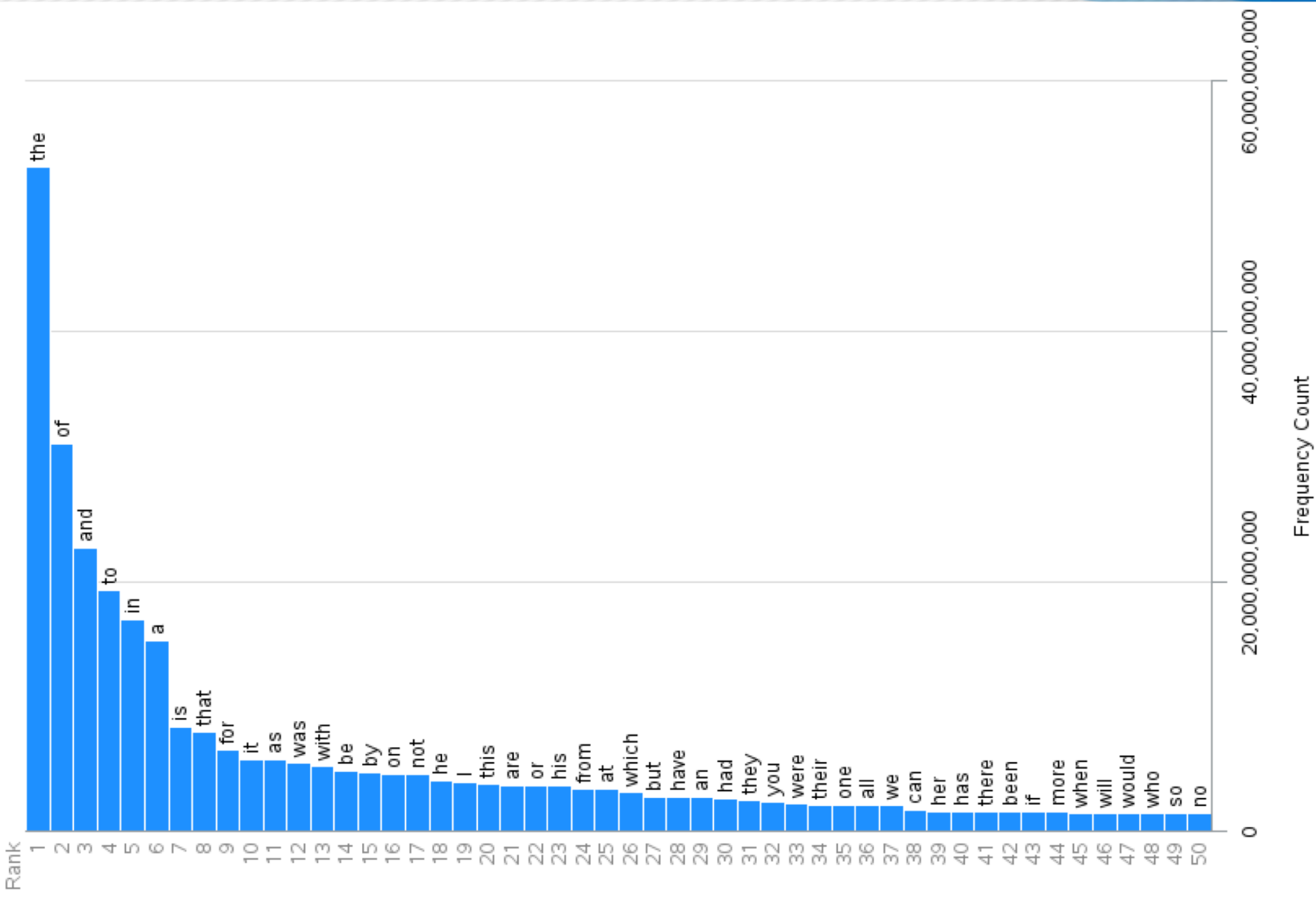


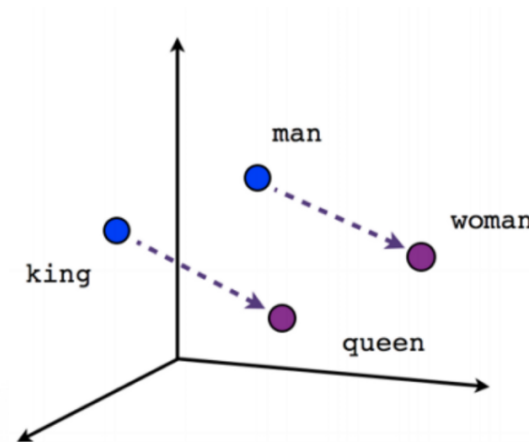
01

영어 단어 중 빈도수가 가장 높
은 것?

50 Most Frequent Words in English Writing

Based on Google books data





<https://medium.com/@khulasaandh/word-embeddings-fun-with-word2vec-and-game-of-thrones-ea4c24fcf1b8>



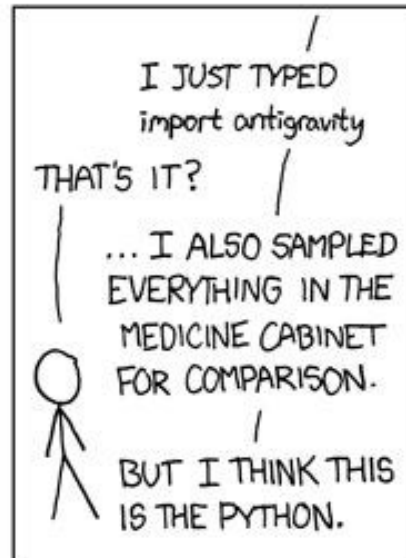
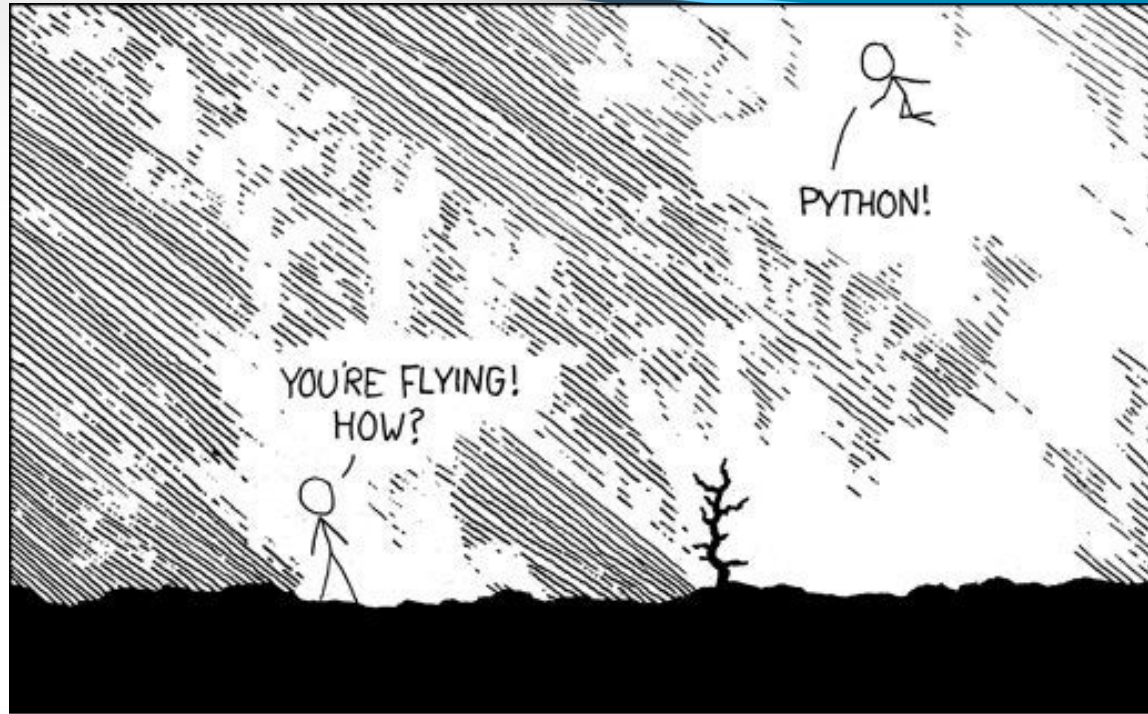


02



Library & Class

import 사용하기



`pip3 install wordcloud`

```
text = "Sungshin Sungshin Sungshin Sungshin University University"
```

```
import wordcloud
```

```
wc_obj = wordcloud.WordCloud()
```

library

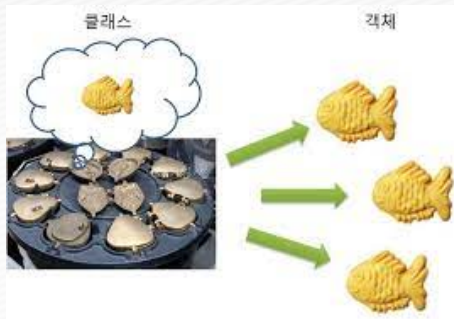
class

```
wc = wc_obj.generate(text)
```

method

```
wc.words_
```

attribute





03

Image as Numbers

Image as numbers

- 흑백 그림은 픽셀의 밝기 값을 0-255 사이의 값으로 표현한 이미지



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

pixel

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218



```
import matplotlib.pyplot as plt
```

```
plt.figure()
```

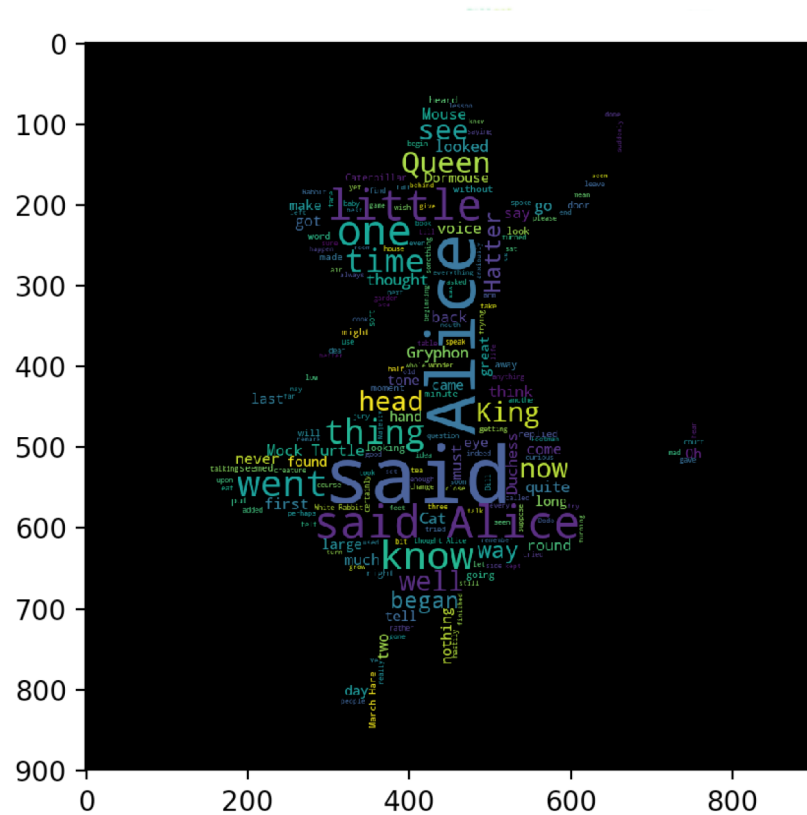
```
plt.imshow(wc)
```

```
plt.axis("off")
```

```
plt.show()
```

```
plt.savefig("wc_sungshin.png")
```

```
import matplotlib.pyplot as plt
```



준비물: alice_mask.png

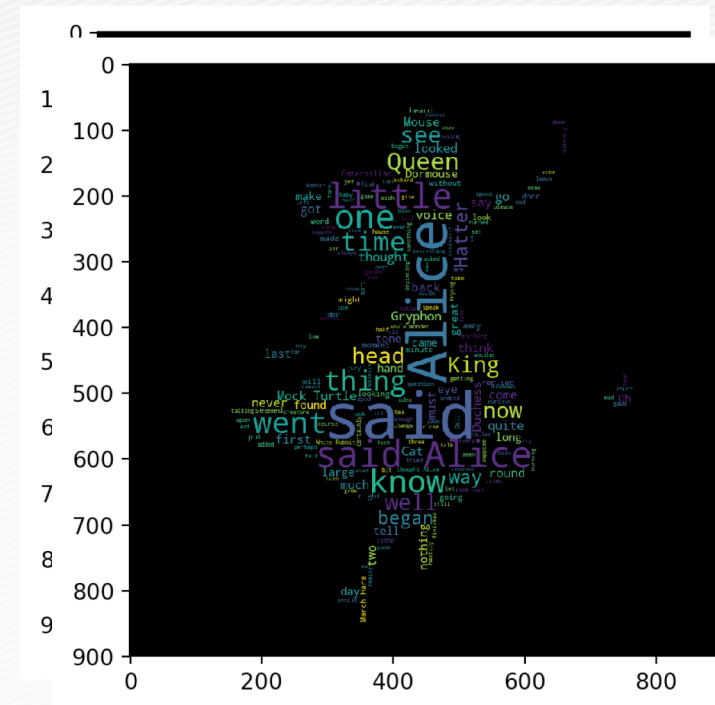

```
from PIL import Image
import numpy as np
alice_mask = np.array(Image.open('alice_mask.png'))
plt.figure()
plt.imshow(alice_mask)
plt.axis("off")
```

```
array([[255, 255, 255, ..., 255, 255, 255],
       [255, 255, 255, ..., 255, 255, 255],
       [255, 255, 255, ..., 255, 255, 255],
       ...,
       [255, 255, 255, ..., 255, 255, 255],
       [255, 255, 255, ..., 255, 255, 255],
       [255, 255, 255, ..., 255, 255, 255]], dtype=uint8)
```



```
import wordcloud
wc_obj = wordcloud.WordCloud(mask=alice_mask)
wc = wc_obj.generate(text)
```

```
plt.figure()  
plt.imshow(wc)  
plt.imshow(alice_mask)  
plt.axis('off')  
plt.show()
```



THANK YOU



MEMO



02844 서울특별시 성북구 보문로 34다길 2

Copyright 2021.성신여자대학교 All rights reserved.