파이썬을 활용한 인공지능 체험하기

3주: 파이썬 리스트와 튜플/인공지능체험

한양대학교 ERICA 소프트웨어학부

조상욱 교수

swcho3@hanyang.ac.kr

차례

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

학습내용

01. 리스트에 대하여 학습합니다.
02. 리스트를 생성하고 항목을 삽입, 삭제, 변경, 탐색하는 것을 학습합니다.
03. 리스트의 구조를 이해하고 정렬하는 것을 학습합니다.
04. 반복문과 함께 리스트를 사용하는 것을 이해합니다.
05. 머신러닝체험

실습

01. 오늘의 명언
02. 스파이럴(spiral) 그리기
04. 습도 구하기

01. 리스트란?

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

- 많은 양의 자료를 사용해야 할 경우에는 일일이 선언?
- 여러 개의 데이터를 의미 있게 묶어서 저장하려면 어떻게?

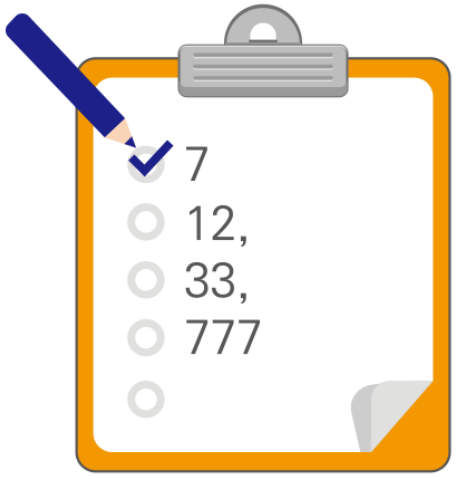
01. 리스트란?

히어로들의 이름 리스트를 생성해 보겠습니다.

heroes = [“아이언맨”, “토르”, “헐크”, “스칼렛 위치”]

01. 리스트란?

수치형 자료도 목록을 다음과 같이 쓸 수 있습니다. 행운의 수를 리스트로 생성해 보겠습니다.



numbers = [7, 12, 33, 777]

02. 리스트의 생성과 추가

- 인덱스(index) : 리스트의 항목의 위치를 알려주는 번호
- 리스트에서 인덱스는 0번부터 시작

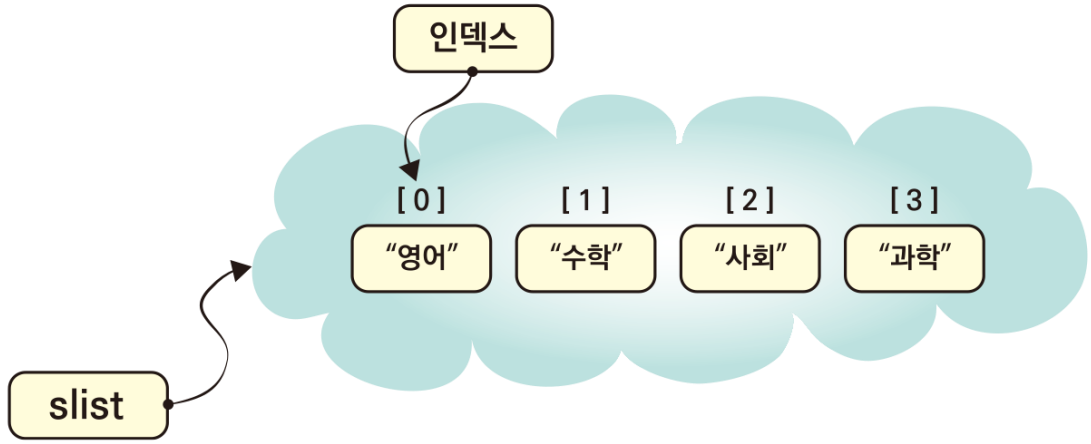
교과목 리스트는 다음과 같이 생성합니다.

코드

```
slist = [ '영어', '수학', '사회', '과학' ]
print(slist)
```

실행 결과

```
['영어', '수학', '사회', '과학']
```



02. 리스트의 생성과 추가

공백 리스트를 생성한 후 값을 추가할 수도 있습니다.

코드

```
cart = []
cart.append("사과")
print(cart)
```

실행 결과

['사과']

코드

```
cart = []
cart.append("사과")
cart.append("세제")    #항목 추가
print(cart)
```

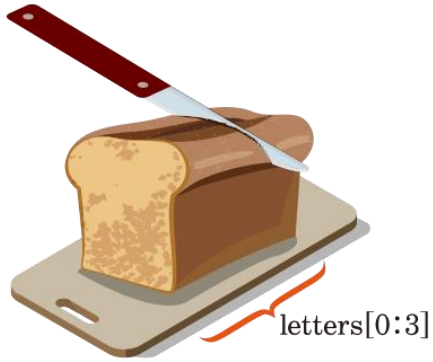
실행 결과

['사과', '세제']

03. 리스트 항목에 접근하기

- 슬라이싱(slicing) : 여러 개의 항목을 추출하는 기법

리스트 항목을 슬라이싱하여 읽어 봅시다.



코드

```
letters = ['A', 'B', 'C', 'D', 'E', 'F']
print(letters[0:3])
print(letters[ : 3])
print(letters[3 : ])
print(letters[ : ])
```

실행 결과

```
['A', 'B', 'C']
['A', 'B', 'C']
['D', 'E', 'F']
['A', 'B', 'C', 'D', 'E', 'F']
```

03. 리스트 항목에 접근하기

자칫 어려워 보일 수도 있지만, 생략된 표현은 간략하면서도 편리한 기능입니다.

코드

```
letters = ['A', 'B', 'C', 'D', 'E', 'F']
copy = letters[:]
print(copy)
```

실행 결과

```
['A', 'B', 'C', 'D', 'E', 'F']
```

04. 리스트 항목의 변경과 추가

인덱스를 사용하여 지정된 위치의 항목을 변경해 보겠습니다.

코드

```
>>> cart=['사과', '세제', '화장지', '치약']
>>> cart[1] = '섬유 유연제'
>>> print(cart)
```

실행 결과

```
['사과', '섬유 유연제', '화장지', '치약']
```

04. 리스트 항목의 변경과 추가

하지만 아직 존재하지 않는 인덱스의 항목은 당연히 변경할 수 없습니다.

코드

```
>>> cart[10]='양말'
```

실행 결과

```
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    cart[10]='양말'
IndexError: list assignment index out of range
```

04. 리스트 항목의 변경과 추가

리스트에 항목을 추가하기 위해서는 append( ) 함수를 사용합니다.

코드

```
>>> cart.append('양말')
>>> print(cart)
```

실행 결과

['사과', '섬유 유연제', '화장지', '치약', '양말']

04. 리스트 항목의 변경과 추가

리스트에 항목을 추가하기 위해서는 insert( ) 함수를 사용합니다.

코드

```
>>> cart.insert(1, '건전지')
>>> print(cart)
```

실행 결과

['사과', '건전지', '섬유 유연제', '화장지', '치약', '양말']

05. 리스트 항목 삭제하기

방법 1: 리스트에서 remove( )를 이용하여 항목을 삭제합니다.

코드

```
cart = ['사과', '세제', '화장지', '치약']
cart.remove("화장지")
print(cart)
```

실행 결과

```
['사과', '세제', '치약']
```

05. 리스트 항목 삭제하기

방법 1: 리스트에서 remove( )를 이용하여 항목을 삭제합니다.

코드

```
cart = ['사과', '세제', '화장지', '치약']
cart.remove("화장지")
print(cart)
```

실행 결과

```
['사과', '세제', '치약']
```

코드

```
cart=['사과', '세제', '화장지', '치약']
if '화장지' in cart:
    cart.remove('화장지')
print(cart)
```

05. 리스트 항목 삭제하기

방법 2: 리스트에서 del을 이용하여 항목을 삭제합니다.

코드

```
cart=['사과', '세제', '화장지', '치약']
del cart[2]
print(cart)
```

실행 결과

```
['사과', '세제', '치약']
```

05. 리스트 항목 삭제하기

방법 3: 리스트에서 pop( )을 이용하여 항목을 삭제합니다.

코드

```
cart=['사과', '세제', '화장지', '치약']
item = cart.pop( )
print(cart)
print(item)
```

실행 결과

```
['사과', '세제', '화장지']
치약
```

06. 리스트에서 항목 탐색하기

리스트에서 index( )를 사용하면 해당 항목의 인덱스 번호를 알 수 있습니다.

코드

```
cart=['사과', '세제', '화장지', '치약']
print(cart.index("화장지"))
```

실행 결과

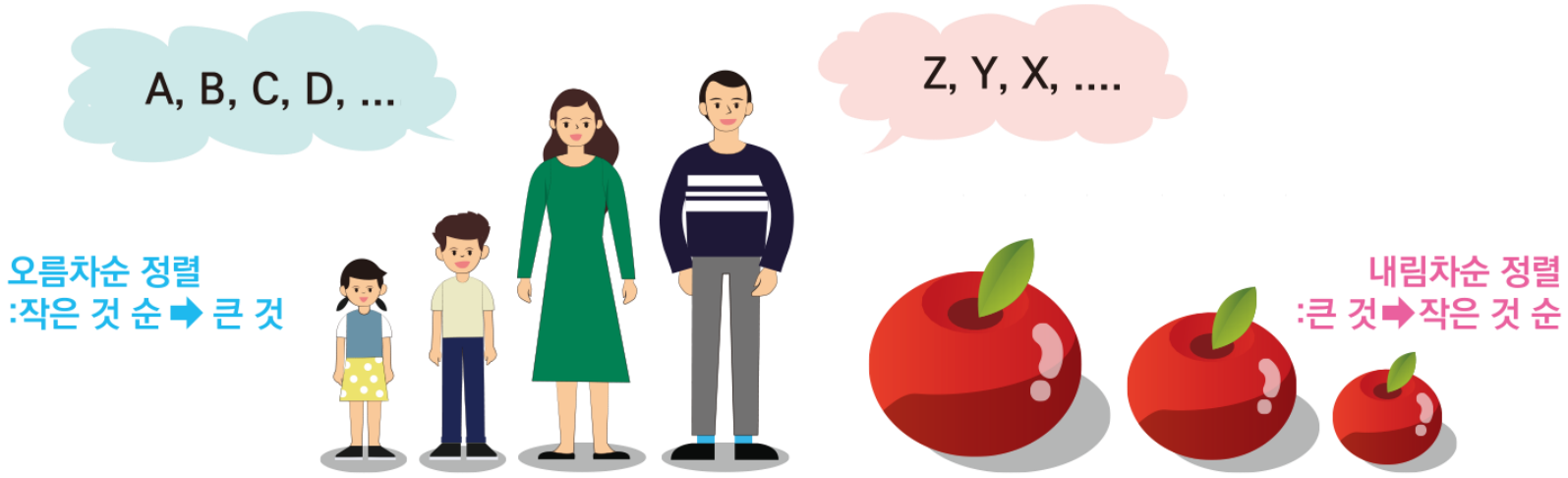
```
2
```

코드

```
cart=['사과', '세제', '화장지', '치약']

if '화장지' in cart:
   print(cart.index("화장지"))
```

07. 리스트 정렬하기

- 정렬(sorting) : 일정한 규칙에 따라 자료를 순서대로 정리하는 것



07. 리스트 정렬하기

리스트는 sort( )를 사용하여 쉽게 오름차순 정렬을 할 수 있습니다.

코드

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]
heroes.sort( )
print(heroes)
```

실행 결과

['스칼렛 위치', '아이언맨', '토르', '헐크']

07. 리스트 정렬하기

리스트는 sort( )에 'reverse = True' 옵션을 사용하여 쉽게 내림차순 정렬을 할 수 있습니다.

코드

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]
heroes.sort(reverse = True)
print(heroes)
```

실행 결과

['헐크', '토르', '아이언맨', '스칼렛 위치']

07. 리스트 정렬하기

만약 정렬된 새로운 리스트가 필요하면 sorted( ) 함수를 사용합니다

코드

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]
new_heroes = sorted(heroes)
print(heroes)
print(new_heroes)
```

실행 결과

```
['아이언맨', '토르', '헐크', '스칼렛 위치']
['스칼렛 위치', '아이언맨', '토르', '헐크']
```

09. 리스트와 반복문의 궁합

반복문을 이용하여 리스트의 내용을 출력해 봅시다.

코드

```
for i in [1, 2, 3] :
   print("i=", i)
```

실행 결과

```
i= 1
i= 2
i= 3
```

09. 리스트와 반복문의 궁합

반복문을 이용하여 효과적으로 리스트를 만들고 출력할 수 있습니다.

코드

```
heroes=[ ]

for i in range(5) :
   name = input("영웅들의 이름을 입력하시오: ")
   heroes.append(name)

for i in heroes :
   print(i, end=" ")
```

실행 결과

```
영웅들의 이름을 입력하시오: 홍길동
영웅들의 이름을 입력하시오: 아이언맨
영웅들의 이름을 입력하시오: 슈퍼맨
영웅들의 이름을 입력하시오: 베트맨
영웅들의 이름을 입력하시오: 스파이더맨
홍길동 아이언맨 슈퍼맨 베트맨 스파이더맨
```

09. 리스트와 반복문의 궁합

반복문과 조건문을 이용하면 내가 원하는 항목만 출력할 수 있습니다.

코드

```
num = [100, 96, 209, 22, 30, 117]

for i in num :
  if i % 2 == 1 :
    print(i, end=" ")
```

실행 결과

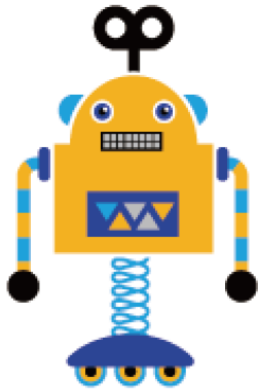
```
209 117
```

실습. 오늘의 명언

실행 결과

```
##############################
#        오늘의 명언
##############################
고생 없이 얻을 수 있는 진실로 귀중한 것은 하나도 없다.
```

고생 없이 얻을 수 있는 진실로 귀중한 것은 하나도 없다.



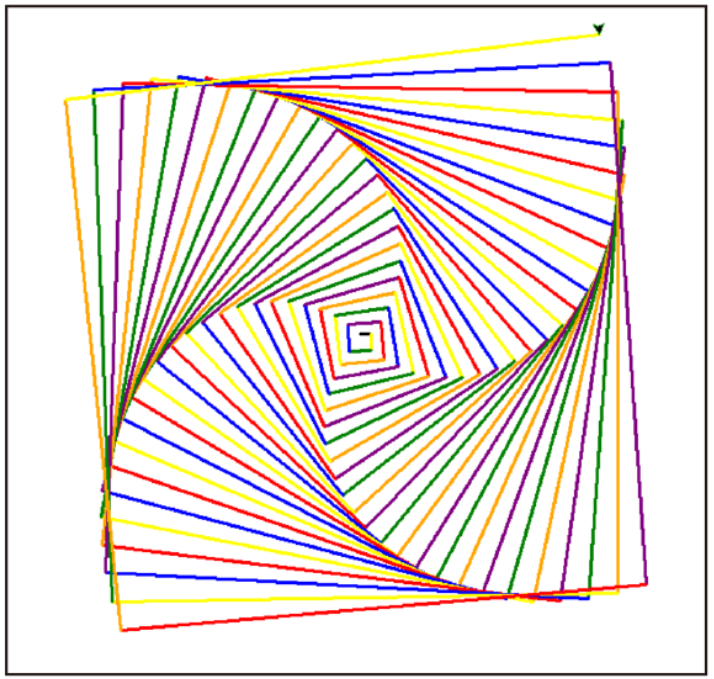
사람은 사랑할 때 누구나 시인이 된다.

꿈을 지녀라. 그러면 어려운 현실을 이길 수 있다.

Sol.


```
import random
quotes = [ ]

quotes.append("꿈을 지녀라. 그러면 어려운 현실을 이길 수 있다.")
quotes.append("분노는 바보들의 가슴속에서만 살아간다.")
quotes.append("고생 없이 얻을 수 있는 진실로 귀중한 것은 하나도 없다.")
quotes.append("사람은 사랑할 때 누구나 시인이 된다.")
quotes.append("시작이 반이다.")

dailyQuote = random.choice(quotes)
print("################################")
print("#          오늘의 명언          ")
print("################################")
print("")
print(dailyQuote)
```

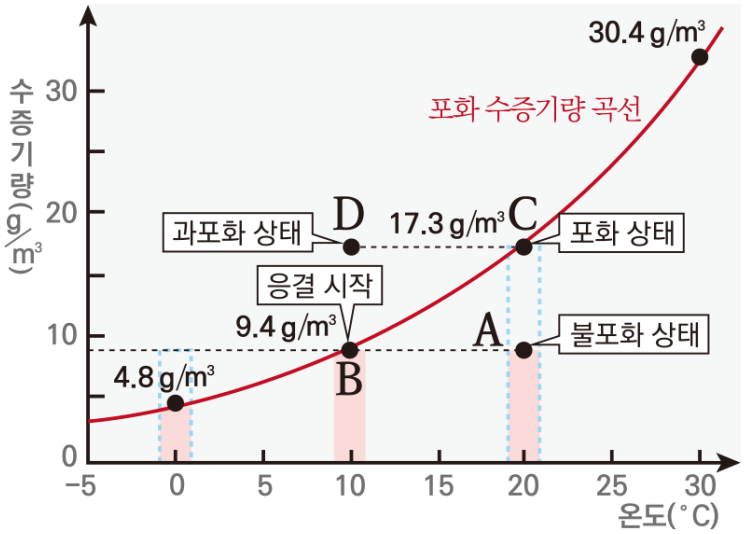
실습. 스파이럴(spiral) 그리기



Sol.


```
import turtle
t = turtle.Turtle()

t.speed(0) # 거북이의 속도는 0으로 설정하면 최대가 됩니다.
t.width(3) #거북이가 그리는 선의 두께는 width()를 호출합니다.

length = 10        # 초기 선의 길이는 10으로 합니다.

# 색상은 리스트에 저장했다가 하나씩 꺼내서 변경하도록 합니다.
colors = ["red", "purple", "blue", "green", "yellow", "orange" ]

# while 반복문이다. 선의 길이가 500보다 작으면 반복.
while length < 500:
    t.forward(length)                    # length만큼 전진합니다.
    t.pencolor(colors[length%6])      # 선의 색상을 변경합니다.
    t.right (89)                         # 89도 오른쪽으로 회전합니다.
    length += 5                          # 선의 길이를 5만큼 증가합니다.
```

실습. 습도 구하기



$$\text{습도} = \frac{\text{현재 수증기량}}{\text{포화수증기량}} \times 100$$

$$\text{공기 A의 습도} = \frac{9.4\text{g/m}^3}{17.3\text{g/m}^3} \times 100 \fallingdotseq 54.3\%$$

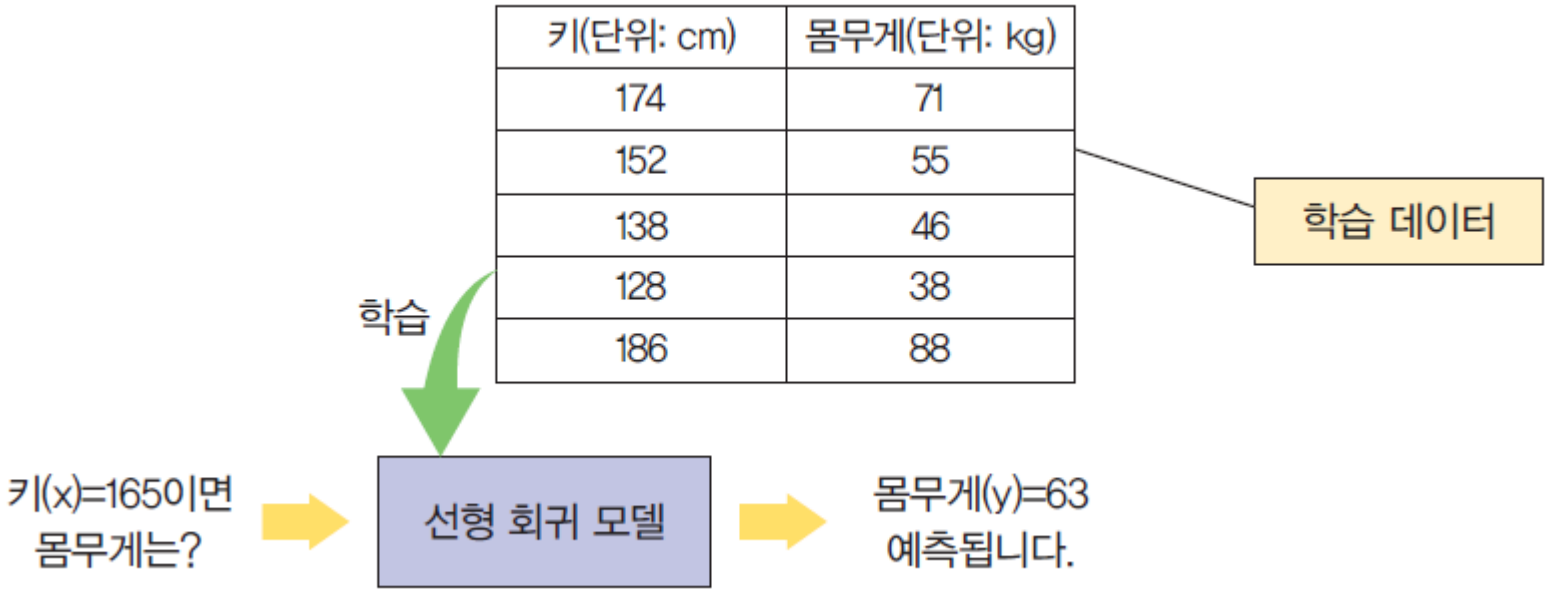
기온(℃)	0	10	20	30
포화수증기량(g/m³)	4.8	9.4	17.3	30.4

Sol.


```
temp_list = [0, 10, 20, 30]
vapor_list = [4.8, 9.4, 17.3, 30.4]

vapor = float(input("현재 수증기량 입력: "))
temp = int(input("현재 온도 입력: "))

if temp in temp_list :
    humidity = ( vapor / vapor_list[temp_list.index(temp)] ) * 100
    print("현재 습도는" , humidity , "% 입니다.")

print("프로그램을 종료합니다.")
```

머신러닝 체험

- 인간의 키와 몸무게는 어느 정도 비례할 것으로 예상된다. 아래와 같은 데이터가 있을 때, 선형 회귀를 이용하여 학습시키고 키가 165cm일 때의 예측 값을 얻어보자. 파이썬으로 구현하여 본다.

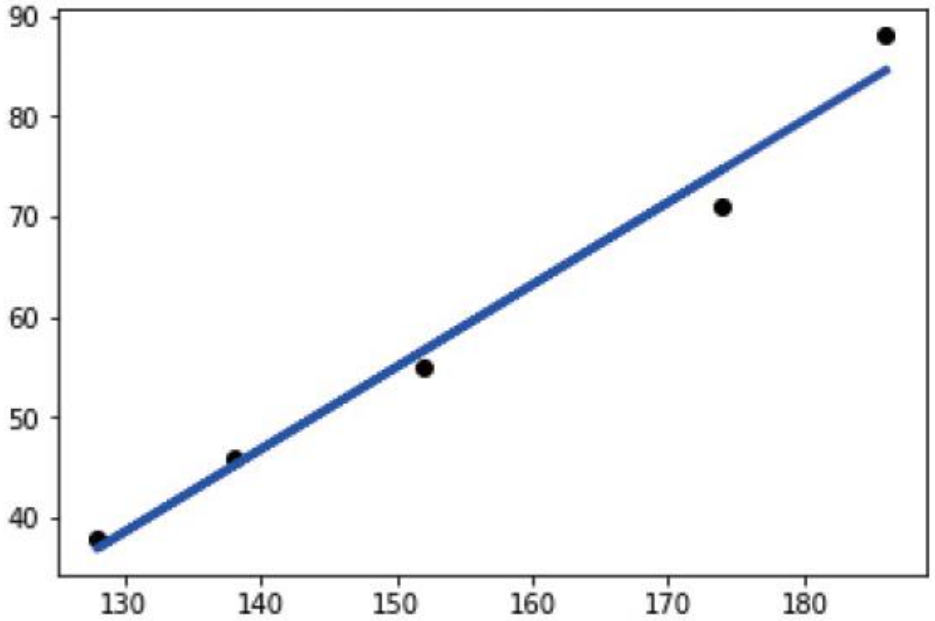


Sol.


```
import matplotlib.pylab as plt
from sklearn import linear_model

reg = linear_model.LinearRegression()

X = [[174], [152], [138], [128], [186]]
y = [71, 55, 46, 38, 88]

reg.fit(X, y)                                    # 학습

print(reg.predict([[165]]))

# 학습 데이터와 y 값을 산포도로 그린다.
plt.scatter(X, y, color='black')

# 학습 데이터를 입력으로 하여 예측값을 계산한다.
y_pred = reg.predict(X)

# 학습 데이터와 예측값으로 선그래프로 그린다.
# 계산된 기울기와 y 절편을 가지는 직선이 그려진다.
plt.plot(X, y_pred, color='blue', linewidth=3)
plt.show()
```

Sol.


```
[67.30998637]
```



감사합니다.