python

스마트인재개발원
Smart Human Resources Development

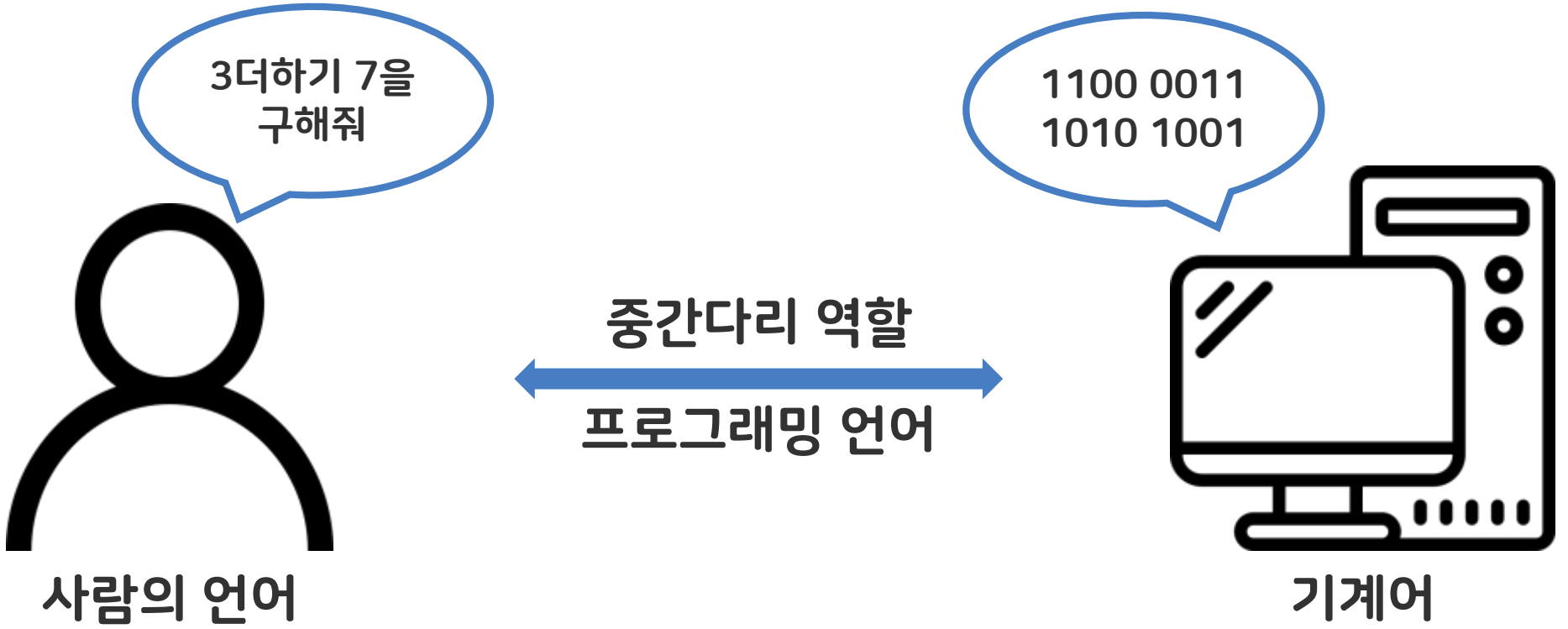
손 지 영 강사

python

학습목표

- Python의 개념, 특징을 이해한다.
- Python 개발환경구축을 할 수 있다.
- 변수 및 자료형에 대해 알 수 있다.
- 문자열 자료형을 이해하고 활용 할 수 있다.

python 프로그래밍 언어(Programming Languages)

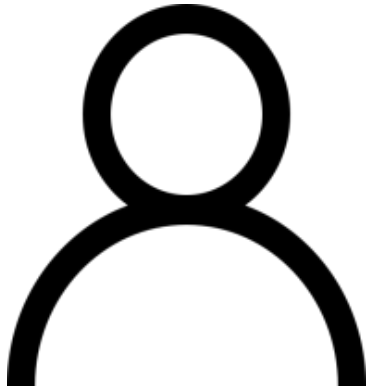


python 프로그래밍 언어(Programming Languages)

**컴퓨터를 이용하여 특정 문제를 해결하기 위한
프로그램을 작성하기 위해 사용되는 언어**

고급 언어

C, Java, Python 등



저급 언어

어셈블리어

python 프로그래밍 언어 순위

스마트인재개발원
Smart Human Resources Development

TIOBE
the software quality company

2025 12월
popularity

Dec 2025	Dec 2024	Change	Programming Language	Ratings	Change
1	1		Python	23.64%	-0.21%
2	4	↑	C	10.11%	+1.01%
3	2	↓	C++	8.95%	-1.87%
4	3	↓	Java	8.70%	-1.02%
5	5		C#	7.26%	+2.39%
6	6		JavaScript	2.96%	-1.66%
7	9	↑	Visual Basic	2.81%	+0.85%
8	8		SQL	2.10%	+0.11%
9	26	⇈	Perl	1.97%	+1.33%
10	16	⇈	R	1.96%	+0.91%

1989년 암스테르담의 귀도 반 로섬(Guido Van Rossum)이 개발한 인터프리터 언어

인터프리터 언어란 프로그래밍 언어의 소스 코드를 바로 실행하는 컴퓨터 프로그램

python 파이썬 장점

장점

- 인간다운 언어

```
if 4 in [1,2,3,4]: print("4가 있습니다.")
```

- 문법 쉽고 코드가 간결

```
for(int i=0 ; i<10 ; i++) {
    for(int j=0 ; j<=i ; j++) {
        System.out.print("*");
    }
    System.out.println();
}
```

Java

```
for i in range(10):
    print("*"*(i+1))
```

Python

- 높은 확장성 및 이식성

- 거대한 생태계

https://www.python.org/

python Python 활용

Google

Dropbox

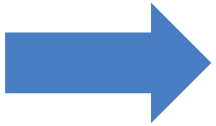
NETFLIX

YouTube

python 개발환경 구축

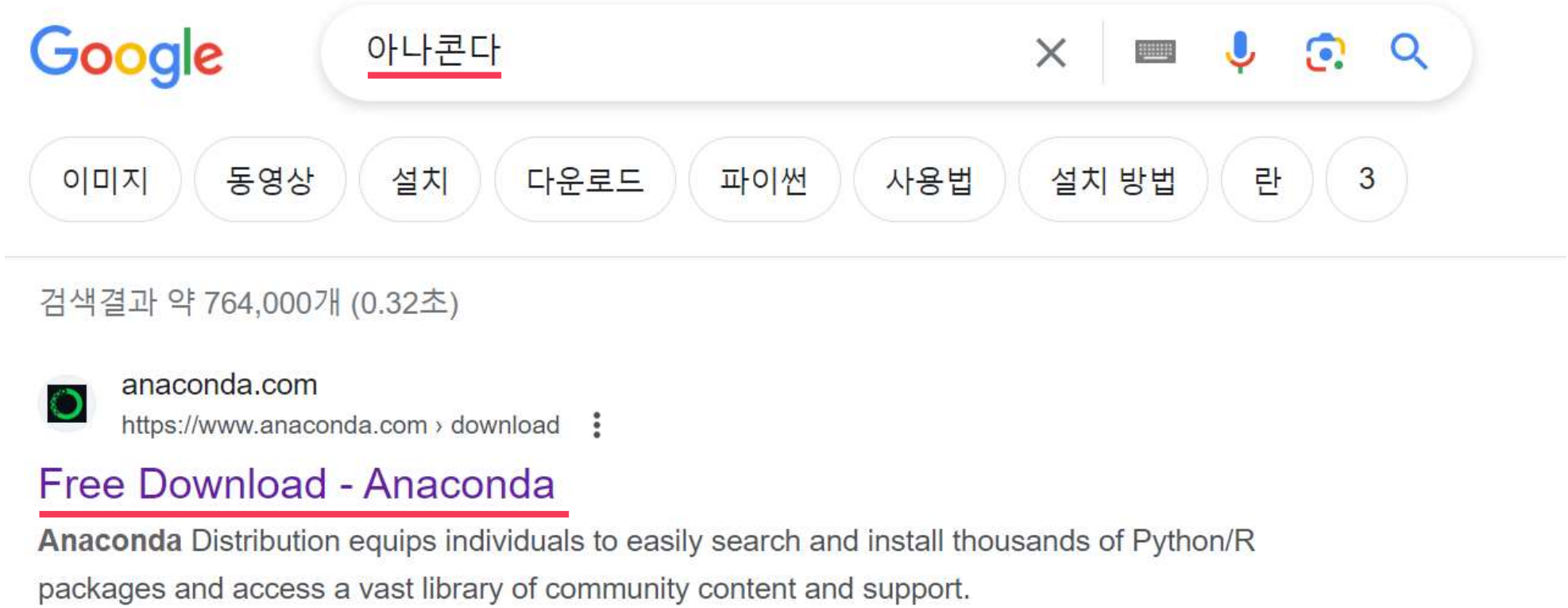
Python
패키지
(3.x)

에디터

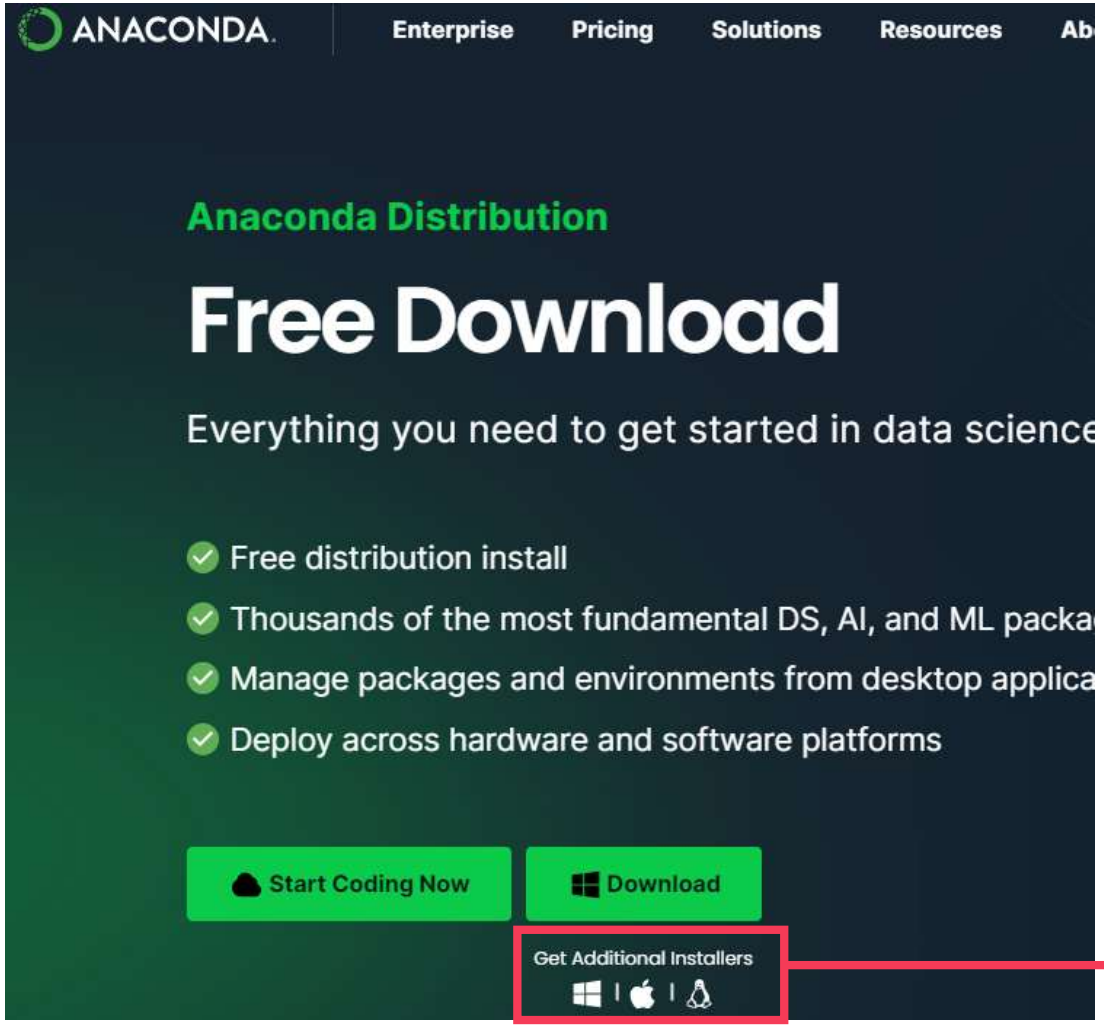


Python을 활용한
데이터 분석, 어플리케이션 개발에
도움을 주는 플랫폼

google에서 anaconda검색

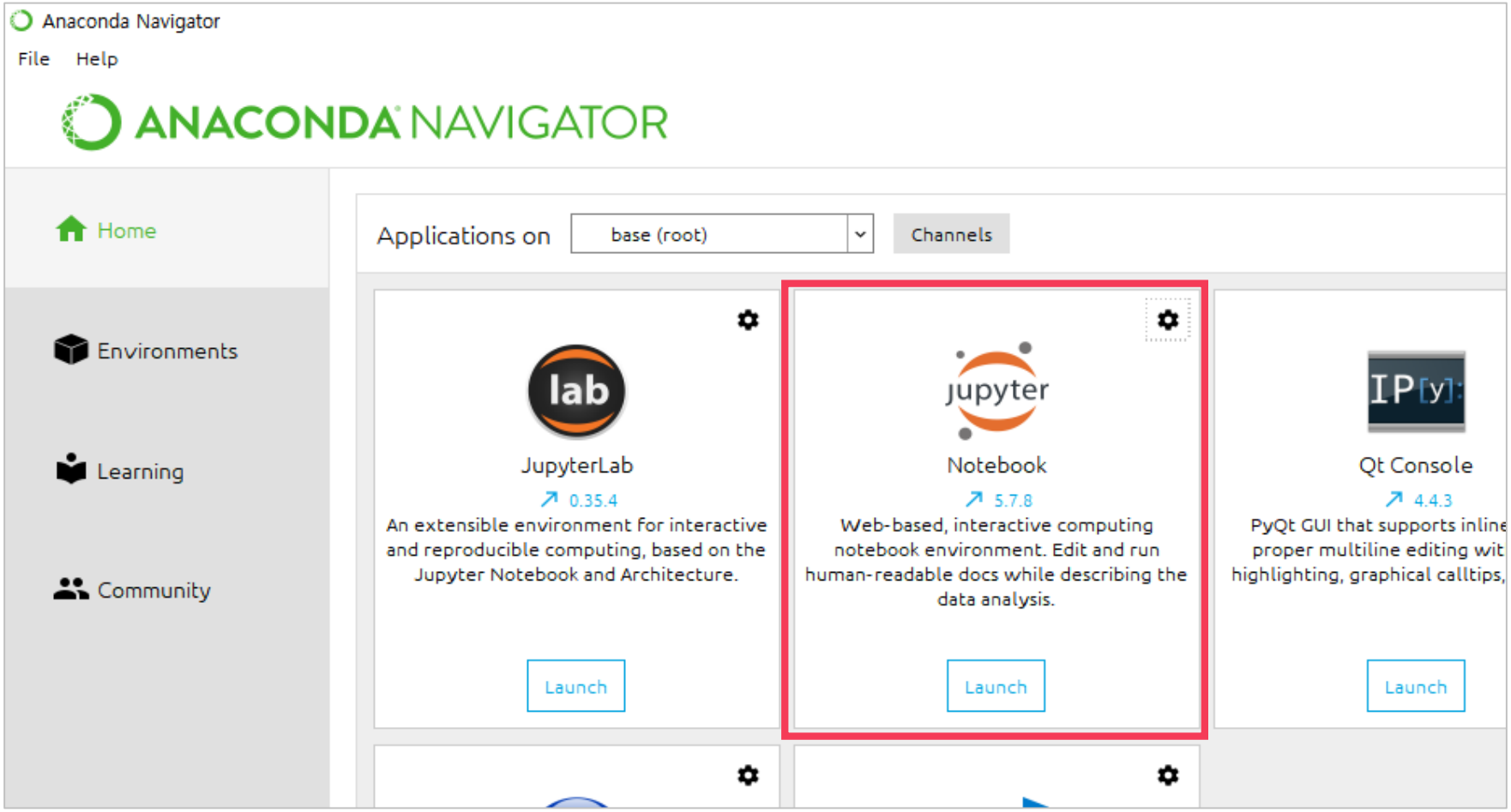


python 개발환경 구축



https://www.anaconda.com/download

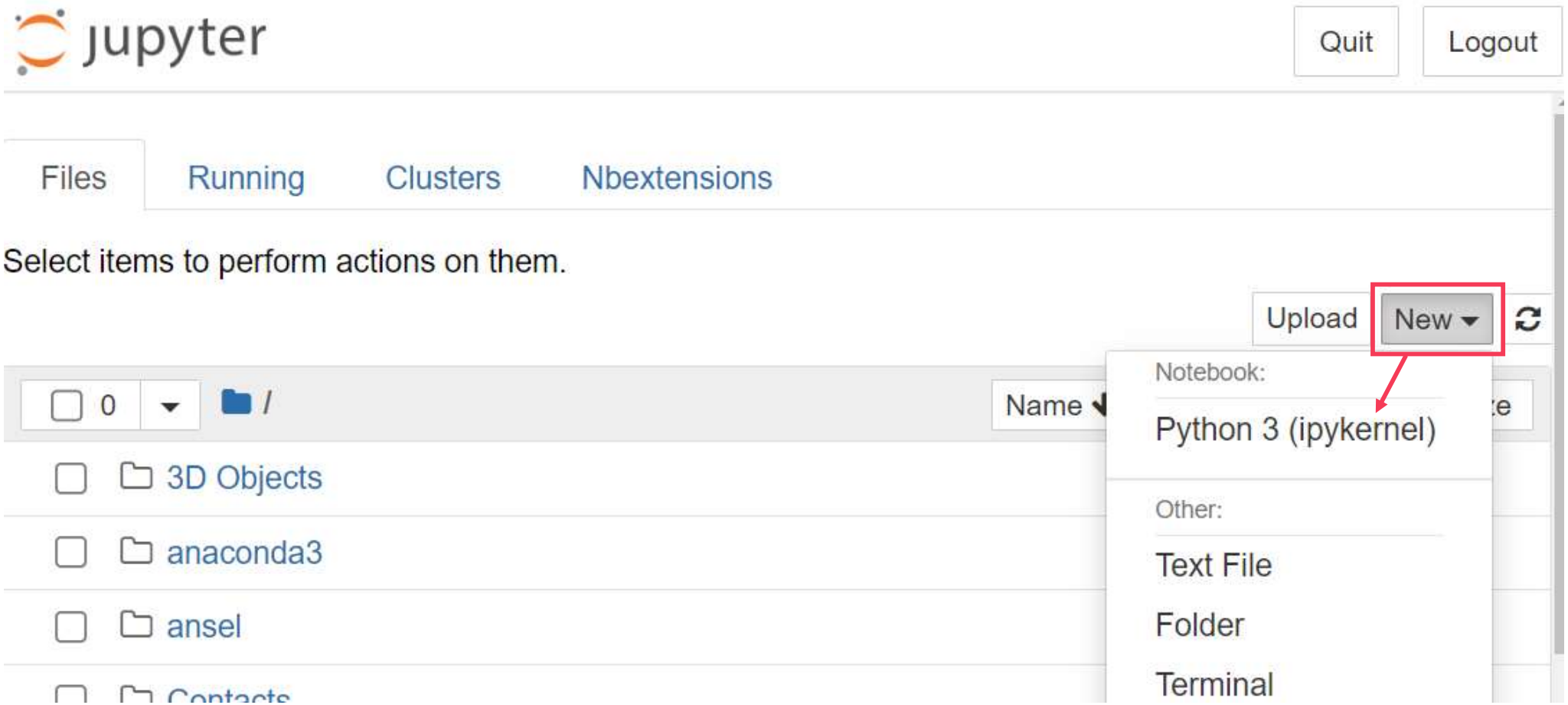
Anaconda 실행 후 Jupyter notebook Launch 클릭



python Jupyter notebook 사용 방법

스마트인재개발원
Smart Human Resources Development

파이썬 노트북 생성



주석

```
1+3  #1 더하기 3  실행
```

```
4
```

- 프로그래밍에 있어 내용을 메모하는 목적으로 사용
- 소스코드를 더 쉽게 이해할 수 있게 만드는 것이 주 목적
- 컴파일러와 인터프리터에 의해 일반적으로 무시되어 프로그램에 영향 X
- 파이썬은 “#”으로 주석

python Jupyter notebook 사용 방법

스마트인재개발원
Smart Human Resources Development

Command Mode

- Enter : Edit Mode로 전환
- a : 위에 셀(Cell) 추가
- b : 아래에 셀(Cell) 추가
- m : Markdown으로 전환
- y : 뒤로 되돌리기
- dd : 셀(cell) 삭제
- Z : 셀 되돌리기

Edit Mode

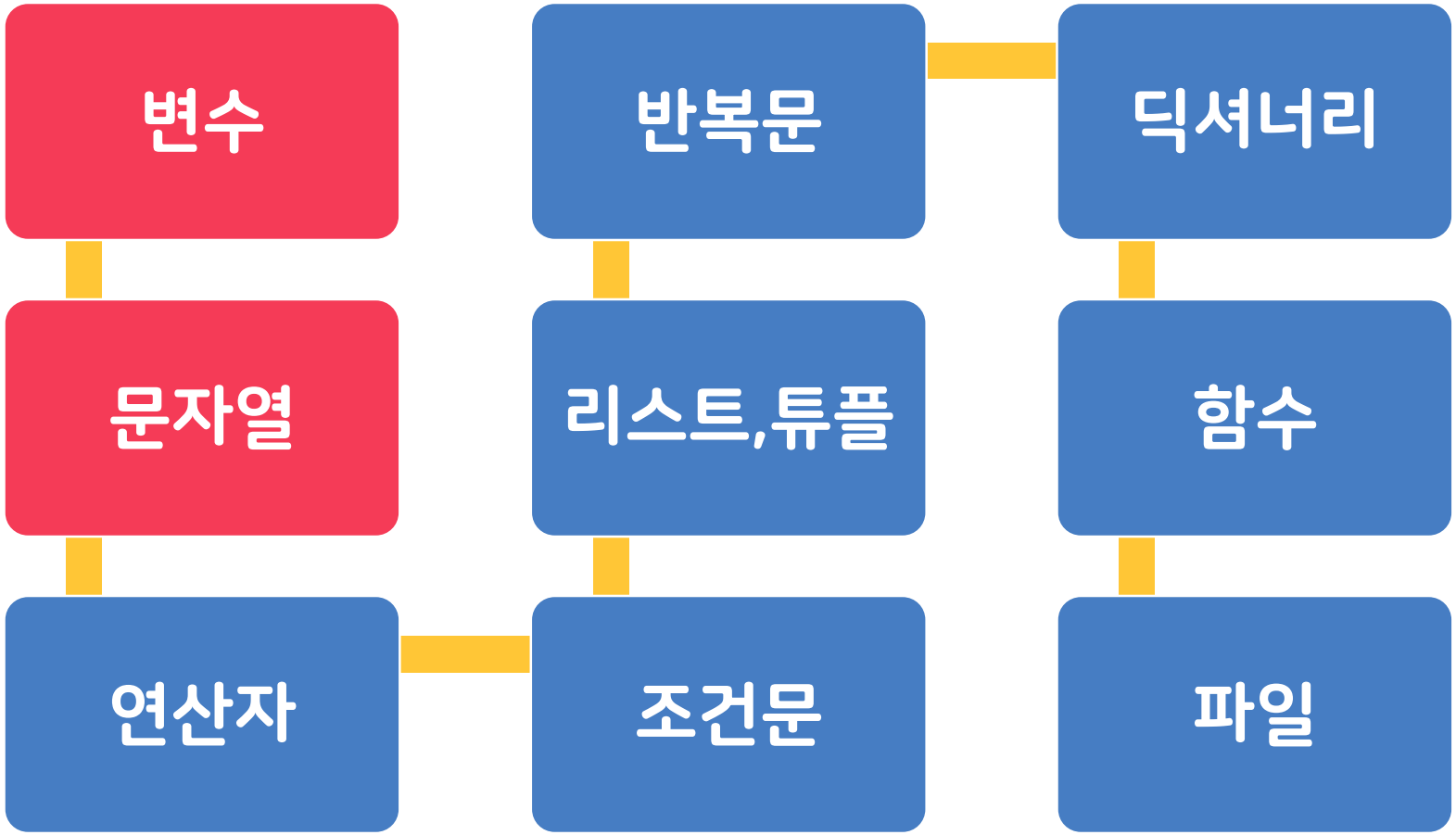
- Esc : Command Mode로 전환
- Ctrl + z : 앞으로 되돌리기
- Ctrl + y : 뒤로 되돌리기
- Ctrl + d : 커서 있는 줄 삭제

실행 단축키

- Ctrl + Enter : 셀(cell) 실행
- Shift + Enter : 셀(cell) 실행 후 아래로 커서 이동
- Alt + Enter : 셀(cell) 실행 후 아래에 셀(cell) 추가

python

수업 진행방향



Smart Media
스마트미디어인재개발원

변수 (variable)

- 사전적 의미로는 “변화를 줄 수 있는 " 또는 “변할 수 있는 수”
- 프로그래밍에서는 데이터를 담을 수 있는 공간

1. 영문자, 숫자, 언더바(_)를 사용 가능

단, 영문자는 대문자와 소문자를 다르게 인식

```
number = 10
Number = 20
print("number :",number)
print("Number :",Number)
```

```
number : 10
Number : 20
```

```
num1_num2 = 10
num1_num2
```

```
10
```

2. 숫자로 시작 불가능

```
1a = 10
```

```
  File "<ipython-input-3-1be9f8edb7cd>", line 1
    1a = 10
     ^
SyntaxError: invalid syntax
```

3. 키워드 사용 불가능

또한 내장함수명 사용을 권장하지 않음

```
if = 30
if
```

```
  File "<ipython-input-5-23187ed1a6a1>", line 1
    if = 30
       ^
SyntaxError: invalid syntax
```

키워드 확인

```
import keyword
print(keyword.kwlist)
```

권장사항

- 변수명의 첫 글자는 항상 소문자로 지정
- 두 가지의 문자를 섞어서 변수명을 만들 경우 두 단어를 구분

  ex) number_list(Snake Case),

  NumberList(Pascal Case),

  numberList(Camel Case)

num = 3

변수명 대입 값


```
num = 3
print(num)
print(type(num))
```

동적 자료형

- 동적 자료형은 값을 넣어 줌과 동시에 자료형이 정해짐
- 변수 실행단계(Run time)에서 만들어 줌

python 자료형 종류

스마트인재개발원
Smart Human Resources Development

기본/컬렉션 구분	소구분	데이터 타입(객체 타입명)	예
기본형 (basic data type)	숫자형 (numeric type)	integer(int)	-2, -1, 0, 1, 2
		float(float)	3.2, 3.14, 0.12
	불리언형 (boolean type)	Boolean(bool)	True / False
	문자열 및 순서형 (text type, sequence type)	string(str)	'Hello World!', "Hi", "123"
컬렉션형 (collection data type)	순서형 (sequence type)	*list(list)	[], [1,2,3], ['a','b'], ['a',1]
		tuple(tuple)	(), (1,2,3), ('a','b'), ('a',0.5)
	맵핑형 (mapping type)	*dictionary(dictionary)	{'name':'jy', 'birth':'0211'}
	집합형 (set type)	*set(set)	{1,2,3}, {'a','b','c'}

변수에 숫자형 데이터 대입하고 출력해보기

- num1 변수에 숫자 13을 대입하시오.
- num1 변수에 숫자 25를 대입하시오.
- num2 변수에 숫자 3.1425을 대입하시오.
- num3 변수에 숫자 1.25를 대입하시오.

변수에 문자열 대입하고 출력해보기

- str1 변수에 문자열 “funny python”을 대입하시오.

- str2 변수에 숫자 ‘easy python’를 대입하시오.

각각의 변수에 값 하나씩 대입하기

```
a = 10
b = 15
print(a)
print(b)
```

```
10
15
```

```
a, b = 10, 15
print(a)
print(b)
```

```
10
15
```

여러 변수에 같은 값 “python” 문자열 대입하기

```
str1 = "python"
str2 = "python"
print(str1)
print(str2)
```

```
python
python
```

```
str1 = str2 = "python"
print(str1)
print(str2)
```

```
python
python
```

아래 코드는 오류 없이 잘 실행이 되지만, 문제가 있다. 그 이유는 뭘까?

```
x = 100
y = 200
sum = x + y
print(sum)
```

```
300
```

Python 에서 자료형을 확인하는 방법

- type(대상)
- isinstance(대상,자료형)

```
x = 100
```

```
print("==========x==========")
print(type(x))
print(isinstance(x,int))
print(isinstance(x,float))
print(isinstance(x,str))
```

문자열 데이터 다루기

- 문자열 정의 기호 이해하기
- 문자열 데이터 접근하기
- 문자열 관련 함수 사용하기

문자열 안에 작은따옴표나 큰따옴표를 포함시키고 싶을 때

she's gone

```
s = 'she's gone'
s
```

```
  File "<ipython-input-17-c9d183e05d09>", line 1
    s = 'she's gone'
            ^
SyntaxError: invalid syntax
```

문자열 안에 작은따옴표나 큰따옴표를 포함시키고 싶을 때

she's gone

```
s1 = 'she₩' gone'
print(s1)
```

```
she' gone
```

이스케이프 코드

- 기존 문자가 가진 의미를 벗어나 부가적인 기능을 사용할 때 쓰이는 문자여서 escape(탈출)
- 프로그래밍 할 때 사용할 수 있도록 미리 정의해둔 특수한 "문자 조합"
- 역슬래시(₩ = ＼) 기호와 조합하여 사용

코드	설명
₩n	개행(줄바꿈)
₩t	수평 탭
₩₩	문자 "₩"
₩'	단일 인용부호(')
₩"	이중 인용부호(")

문자열 안에 작은따옴표나 큰따옴표를 포함시키고 싶을 때

she's gone

```
s1 = "she's gone"
s2 = "she\'s gone"
print(s1)
print(s2)
```

```
she's gone
she's gone
```

문자열 안에 작은따옴표나 큰따옴표를 포함시키고 싶을 때

he said that "she is gone"

```
s1 = 'he said that "she is gone"'
s2 = "he said that \"she is gone\""
print(s1)
print(s2)
```

```
he said that "she is gone"
he said that "she is gone"
```

여러 줄인 문자열을 변수에 대입 하고 싶을 때

방법1. 이스케이프코드 활용

```
                                ?
print(s)
```

파이썬
- 동적언어
- 직관적, 명시적, 간결함

여러 줄인 문자열을 변수에 대입 하고 싶을 때

방법2. 따옴표 3개를 연속으로 써서 양쪽 둘러 싸기

```
?
print(s)
```

파이썬
- 동적언어
- 직관적, 명시적, 간결함

python 문자(열)에 접근하는 방법

인덱싱(indexing)

- 무엇인가를 '가리킨다'는 의미

슬라이싱(Slicing)

- 무엇인가를 '잘라낸다'는 의미

인덱스 번호(위치) 기반 접근

```
s = "My name is JY"
```

M	y		n	a	m	e		i	s		J	Y
0	1	2	3	4	5	6	7	8	9	10	11	12
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

문자 인덱싱(indexing)

M	y		n	a	m	e		i	s		J	Y
0	1	2	3	4	5	6	7	8	9	10	11	12
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
s = "My name is JY"
print(s[0])
print(s[8])
```

```
s = "My name is JY"
print(s[-2])
print(s[-1])
```

문자열 슬라이싱(Silcing) - “My”, “name”, “is” 문자열 가져오기

M	y		n	a	m	e		i	s		J	Y
0	1	2	3	4	5	6	7	8	9	10	11	12

```
s = "My name is JY"
print(s[0:2])
print(s     )
print(s      )
```

```
My
name
is
```

python 특정 문자(열)에 접근하는 방법

스마트인재개발원
Smart Human Resources Development

문자열 슬라이싱(Silcing)

- "is JY" 문자열 가져오기

M	y		n	a	m	e		i	s		J	Y
0	1	2	3	4	5	6	7	8	9	10	11	12
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
s = "My name is JY"
print(s[8:13])
print(      )
print(      )
```

```
is JY
is JY
is JY
```

python 문자열 실습 예제

다음과 같은 문자열에서 날짜와 날씨를 출력하시오.

```
day = "2020년 3월 3일의 날씨는 맑음입니다."
```

```
날짜: 2020년 3월 3일
날씨 : 맑음
```

문자열 포매팅(Formatting)

- 문자열 안의 특정한 값을 바꿔야(삽입) 할 경우가 있을 때 사용

```
s = "오늘은 7월 7일 입니다."

s = "오늘은 7월 8일 입니다."
```

- %기호 포매팅
- fomat 함수 포매팅
- f문자열 포매팅

%기호 포매팅

- "%d"를 이용해서 정수 대입

```
day = 7
s = "오늘은 7월 %d일 입니다."%day
s
```

```
'오늘은 7월 7일 입니다.'
```

```
day = 8
s = "오늘은 7월 %d일 입니다."%day
s
```

```
'오늘은 7월 8일 입니다.'
```

%기호 포매팅

- 2개 이상의 값을 포매팅 할 경우

```
month = 7
day = 7
s = "오늘은 %d월 %d일 입니다."%(month,day)
s
```

```
'오늘은 7월 7일 입니다.'
```

문자열 포맷 코드

- 문자열 내 값 삽입

함수	설명
%s	문자열(string)
%c	문자 1개
%d	정수(Integer)
%f	실수(float-point)
%%	Literal % (문자 '%'자체)

format 함수를 사용한 포매팅

```
month = 7
day = 7
s = "오늘은 {}월 {}일 입니다.".format(month,day)
print(s)
```

```
오늘은 7월 7일 입니다.
```

f 문자열을 사용한 포매팅

```
month = 7
day = 7
s = f"오늘은 {month}월 {day}일 입니다."
print(s)
```

```
오늘은 7월 7일 입니다.
```

lang변수에 'python'을 대입하고, 아래와 같이 출력하시오.

```
?
print(s)
```

```
Life is too short, You need 'python'
```

변수 x에는 100을 대입, 변수 y에는 200을 대입 후 변수 sum2에는 두 변수의 합을 대입하고 포매팅을 이용하여 아래와 같이 출력하시오.

```
x = 100
y = 200
sum2 = x + y
print(                    )
```

```
100와 200의 합은 300입니다.
```

실행시점의 현재시간(시,분,초)을 출력하는 출력문을 작성하시오.

단, 아래 코드를 참고하여 현재 시간을 알아내시오.

```
from datetime import datetime
current_time = datetime.now()
current_time.hour
current_time.minute
current_time.second
```

```
from datetime import datetime
current_time = datetime.now()
print(current_time) # hour, minute, second 변수 사용
print(                    ?
```

```
2022-11-07 15:34:44.410353
실행 당시 현재 시간은 15시 34분 44초입니다.
```

python 문자열 함수

함수	설명
len()	문자열의 문자 개수 반환
upper()	소문자를 대문자로 바꾸기
lower()	대문자를 소문자로 바꾸기
isupper()	문자열이 대문자인지 여부 반환
islower()	문자열이 소문자인지 여부 반환
isdigit()	문자열이 숫자인지 여부 반환
count('찾을 문자열')	'찾을 문자열'이 몇 개 있는지 개수 반환
find('찾을 문자열')	'찾을 문자열' 위치 반환
index('찾을 문자열')	'찾을 문자열' 위치 반환
strip(), lstrip(), rstrip()	양쪽 공백 제거, 왼쪽, 오른쪽 공백 제거
replace('문자1', '문자2')	문자열1을 문자열 2로 바꾸기
split()	문자열 나누기

python 문자열 함수

- count('문자')
문자열에 포함된 문자 개수 세기

```
s = 'python is very easy.'

# e 문자열이 s 문자열에 몇번 들었을까요 ?
s.count('e')
```

executed in 6ms, finished 15:03:47 2023-07-14

2

python 문자열 함수

- find('문자')
- index('문자')

문자 위치 알려주기

```
s = 'python is very easy.'
# 해당 문자열의 인덱스 위치 반환
s.find('p')
```

```
0
```

```
# find는 문자열이 없을때 : -1 반환
s.find('z')
```

```
-1
```

```
# 해당 문자열의 인덱스 위치 반환
s.index('p')
```

```
0
```

```
# index는 문자열이 없을때 : 오류 메세지 출력
s.index('z')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (r
<ipython-input-40-180b0228ee70> in <module>
      1 # index는 문자열이 없을때 : 오류 메세지 출력
----> 2 s.index('z')

ValueError: substring not found
```

python 문자열 함수

- '삽입할 문자'.join(문자열)
각각의 문자 사이에 문자 삽입기능

```
s
```

```
'python is very easy.'
```

```
'-'.join(s)
```

```
'p-y-t-h-o-n- -i-s- -v-e-r-y- -e-a-s-y-.'
```

- upper() 대문자로 변경
- lower() 소문자로 변경

```
print('대문자로 변경 전:',s)
s2 = s.upper()
print('대문자로 변경 후:', s2)
```

```
대문자로 변경 전: python is very easy.
대문자로 변경 후: PYTHON IS VERY EASY.
```

```
print('소문자 변경 전:', s2)
s2 = s2.lower()
print('소문자 변경 후:', s2)
```

```
소문자 변경 전: PYTHON IS VERY EASY.
소문자 변경 후: python is very easy.
```

python 문자열 함수

- strip()
문자열 양 옆에 공백 제거
- rstrip()
문자열 오른쪽 공백 제거
- lstrip()
문자열 왼쪽 공백 제거

```
# 전체 문자열 길이 23
# 문자열 전체 앞뒤 공백 제거
print(s.strip(), len(s.strip()) ) # -3
# 오른쪽 공백제거
print(s.rstrip(), len(s.rstrip())) # -2
# 왼쪽 공백 제거
print(s.lstrip(), len(s.lstrip())) # -1
```

```
python is very easy. 20
 python is very easy. 21
python is very easy.   22
```

python 문자열 함수

- replace('찾을값','바꿀값')
찾을값을 찾아서 바꿔주는 기능

```
s = 'python is very easy.'
```

```
# . 을 찾아서 ! 로 바꿔줌
s.replace('.','!')
```

```
'python is very easy!'
```

- split('구분문자')
구분 문자를 기준으로 문자열은 분리하여 리스트 형태로 반환

```
s2 = 'Hi, My name is JY'
```

```
s2.split(',')
```

```
['Hi', ' My name is JY']
```

python 문자열 종합 실습 예제

스마트인재개발원
Smart Human Resources Development

article문자열에 있는 기사입력 일자를 인덱스 번호를 이용하여 추출하고 아래와 같이 출력하시오.

article = "[요기요, 화이트데이 맞이 '선물하기' 이벤트 진행] 기사입력 2023.03.10. 작성자 이나영기자 요기요는 화이트데이를 앞두고 요기요 선물하기 서비스를 이용하는 고객들을 위한 '너와 나의 선물고리' 이벤트를 진행한다고 10일 밝혔다."

```
start = ?
end = ?
day = ?
print( ? )
```

```
기사가 업로드된 일자는 2022.03.10입니다.
```

풀이 접근 방식

1. 문자열 함수 활용 특정 문자열 시작 인덱스번호 확인
2. 인덱스번호 이용 특정 문자열 슬라이싱 하기
3. 슬라이싱 된 문자열을 포매팅 문장 연결하기

python

다음시간에는?

연산자

Smart Media
스마트미디어인재개발원