

초심자를 위한

Python

1주차

Index

1.진행 방향

2.Python 소개

3.Python 사용법

4.기본 문법

진행 방향

**1회 ~ 3시간
주 1회
총 6주**

**기본적으로 C, JAVA등과 같은 다른 프로그래밍 언어를
한 번 이상 접한 사람을
대상으로 합니다.**

Python 소개



스크립트 언어

간결한 언어

Interpreter 언어

1989년도에 Guido Van Rossum이 만든 언어

객체지향 언어

대화형 언어

장점 : 매우 간결 하다. 그래서 배우기 쉽다.

자바에서는 ..

```
1  class Example
2  {
3      public static void main(String[] args)
4      {
5          System.out.println("Hello World!");
6      }
7  }
```

파이썬에서는!

```
1  print("Hello World!")    #python 3에서의 문법
2  print "Hello World!"     #python 2에서의 문법
```

Python 3 vs. Python 2

1. print의 함수화

Python 2 : `print "안녕하세요?"`

Python 3 : `print("안녕하세요?")`

2. long 자료형이 사라지고, int 자료형으로 통합

Python 2 : `type(2**40)` `#type 'long'`

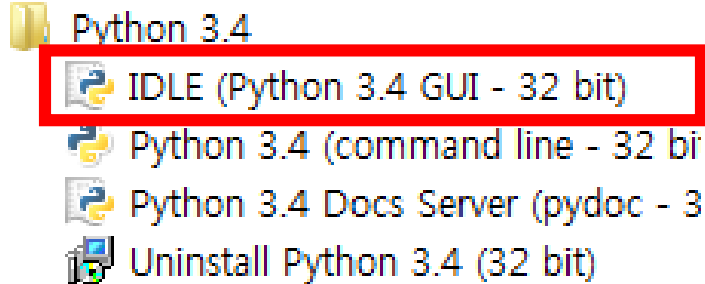
Python 3 : `type(2**40)` `#class 'int'`

이외에도 몇 가지 차이가 있으며, 나머지는 궁금하시면 찾아보세요.

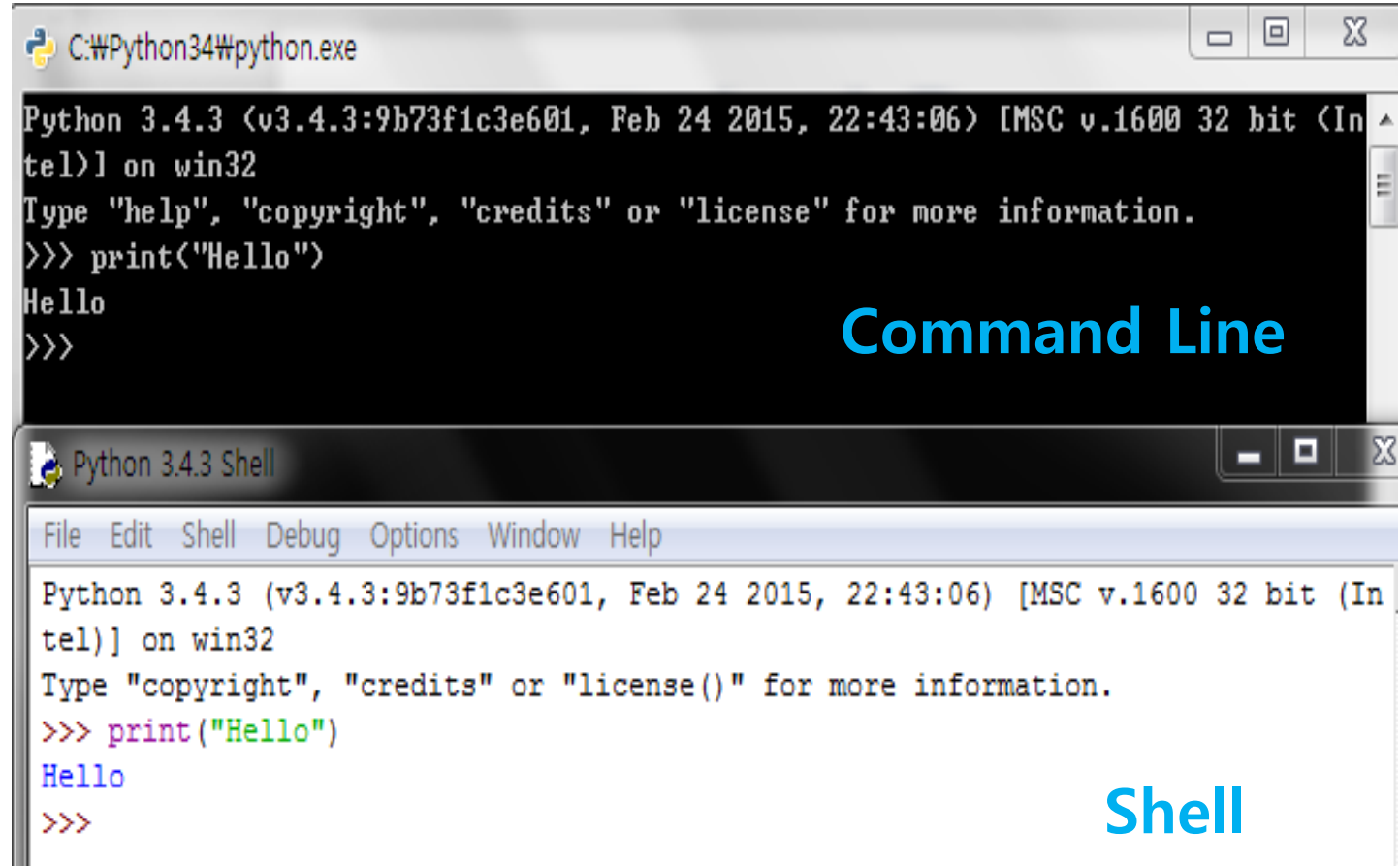
** python 2.7 기준으로 수명 다함.. python 3.x 버전 사용 권장! (python 2와 python 3은 서로 호환 안 됨.)*

Python 사용법

1. IDLE의 사용

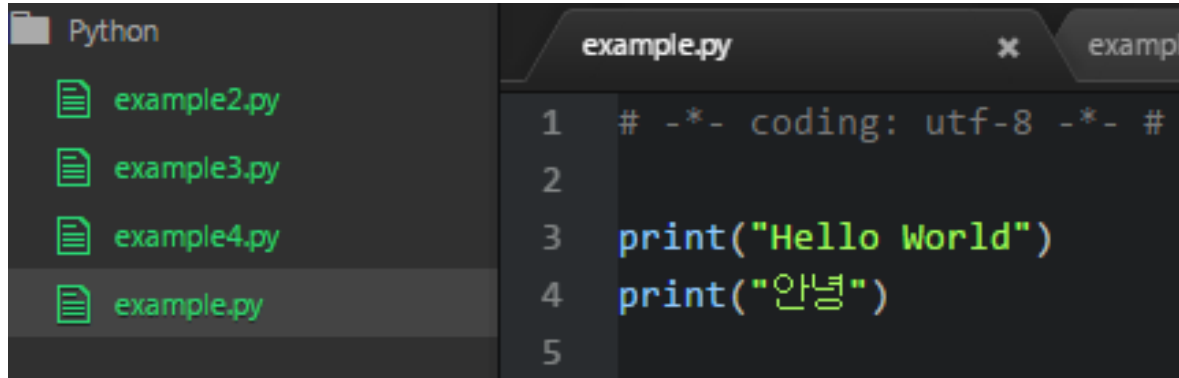


‘IDLE’이라고 있습니다.



밑에 있는게 더 예쁘더라고요.

2. Text Editor+CMD 사용



```
Python
example2.py
example3.py
example4.py
example.py

example.py
1 # -*- coding: utf-8 -*- #
2
3 print("Hello World")
4 print("안녕")
5
```



```
C:\Users\W\Desktop\Python>py example.py
Hello World
안녕
```

* 환경 변수 설정 필요

초반부에는 IDLE을 사용하고,
더 나아갈수록
Text-Editor와 cmd를 사용하고자 합니다.

**Mac 사용자는.. 저도 몰라요..ㄷㄷㄷㄷ*

기본 문법

1. 변수 사용

```
>>> a=10
>>> b=1.1
>>> c=True
>>> d=False
>>> e="Hello"
>>> f='Hi'
```

자유롭게 선언하여
사용합니다.

```
>>> type(a)
<class 'int'>
>>> type(b)
<class 'float'>
>>> type(c)
<class 'bool'>
>>> type(e)
<class 'str'>
>>> type(f)
<class 'str'>
```

type() 함수로
어떤 자료형인지
확인할 수 있습니다.

```
>>> 2ex=10
SyntaxError: invalid syntax
>>> _ex=10
```

변수선언 시,
첫글자로 숫자는 불가능,
언더바(_)는 가능!

```
>>> a=10; b=20; c=30;
```

여러 변수를 한꺼번에 선언!

2. 사칙 연산, 논리연산과 비교연산

```
>>> a=10
>>> b=20
>>> a+b
30
```

```
>>> a=6
>>> b=4
>>> a%b
2
```

```
>>> a=5.62
>>> b=4.2
>>> a-b
1.42
```

```
>>> a=3.14
>>> b=4
>>> 2*b*a
25.12
```

```
>>> a=5
>>> b=3
>>> a/b
1.6666666666666667 -> int형/int형의 결과는 float형이 됩니다.
```

사칙연산입니다.

근데 아쉬워서
나머지 연산도 예시로 넣어줬어요.

2. 사칙 연산, 논리연산과 비교연산

```
>>> y=0
>>> if(not y):
    print("GOOGLE")
```

GOOGLE

```
>>> x=10
>>> y=0
>>> if(x or y):
    print("GOOGLE")
```

GOOGLE

```
>>> x=10
>>> y=20
>>> if(x and y):
    print("GOOGLE")
```

GOOGLE

0의 경우는 False,
0 이외의 경우는 True 처리

2. 사칙 연산, 논리연산과 비교연산

```
>>> a=10
>>> b=20
>>> a>b
False
>>> a==b
False
>>> a<=b
True
>>> a>=b
False
>>> a!=b
True
```

```
>>> a='google'
>>> b='goo'
>>> a==b
False
>>> b='google'
>>> a==b
True
```

```
>>> a='facebook'
>>> b='facebook'
>>> id(a)
44958368
>>> id(b)
44958368
>>> a is b
True
```

```
>>> a='facebook'
>>> b='face'
>>> id(a)
44958368
>>> id(b)
45011232
>>> a is b
False
>>> a is not b
True
```

'is'는 동일한 객체인지 판별합니다.

즉, a와 b가 같은 참조인지 판별하는 것 입니다.

**'is not'은 다른 객체인지 판별합니다.
'is'의 반대라고 보면 간단합니다.**

** id()함수는 메모리 주소 값을 반환합니다.*

3. 출력 하기

```
>>> print("Hi")
Hi
>>> a='google man'
>>> print("Hi %s"%a)
Hi google man
>>> a='facebook'
>>> b='google'
>>> print("Hello %s %s"%(a, b))
Hello facebook google

>>> a=10
>>> print("Oh number %d"%a)
Oh number 10

>>> a=3.14
>>> print("Wow number %f"%a)
Wow number 3.140000
```

```
>>> print("\t google \n\t facebook")
      google
      facebook
```

\t는 'Tab'
\n은 '개행'

%s=문자열 / **%d**=정수 / **%f**=실수

4. 데이터 형

bool : True / False

int : 정수

float : 실수

str : 문자열

List : []

Dictionary : { }

Tuple : ()

```
>>> list_ex=['google', 'facebook', 'naver']
```

```
>>> type(list_ex)
```

```
<class 'list'>
```

```
>>> dict_ex={'kakao':1, 'nate':2, 'twitter':3}
```

```
>>> type(dict_ex)
```

```
<class 'dict'>
```

```
>>> tuple_ex=('instagram', 'tumblr', 'soundcloud')
```

```
>>> type(tuple_ex)
```

```
<class 'tuple'>
```

5. 분기문

```
if(조건1):  
    조건1이 참일 경우 실행할 문장  
elif(조건2):  
    조건1이 만족되지 않고, 조건2가 참일 경우 실행할 문장  
else:  
    위의 모든 조건을 충족하지 못할 경우 실행할 문장
```

if만 사용해도 되고, if~elif를 사용해도 되고,
if~else만 사용해도 됩니다.

python에서는 들여쓰기가 중요한
문법요소이기 때문에 주의해야 합니다.

```
>>> def google(a, b, c):  
    if(a>b):  
        print("%d > %d"% (a, b))  
    elif(b>c):  
        print("%d > %d"% (b, c))  
    else:  
        print("google!!")
```

```
>>> google(1, 2, 3)  
google!!  
>>> google(3, 2, 1)  
3 > 2  
>>> google(1, 3, 2)  
3 > 2
```

**기존에 C나 JAVA에서 'else if'쓰던 것과는 다르게
'elif'로 짧게 줄여쓰는 것을 확인할 수 있습니다.*

6. 반복문

```
>>> for i in range(0, 5):  
    print("google %d"%i)
```

```
google 0  
google 1  
google 2  
google 3  
google 4
```

```
>>> for i in range(1, 9):  
    for j in range(1, 9):  
        print("%d * %d = %d"%(i, j, (i*j)))
```

```
1 * 1 = 1  
1 * 2 = 2  
1 * 3 = 3
```

range(x, y)인 경우, $i=x\sim(y-1)$ 까지의 값을 가지고 반복됩니다.

```
>>> i=0  
>>> while(i<10):  
    i+=1  
    print("google %d"%i)
```

```
google 1  
google 2  
google 3
```

7. 함수 사용

```
>>> def google():  
    print("Love google")
```

```
>>> google()  
Love google
```

매개변수 x
반환 값 x

```
>>> def facebook(num):  
    if (num>3):  
        print("big number")  
    else:  
        print("small number")
```

```
>>> facebook(4)  
big number  
>>> facebook(2)  
small number
```

매개변수 o
반환 값 x

```
>>> def nate(num):  
    return (num*2)
```

```
>>> print(nate(3))  
6
```

매개변수 o
반환 값 o

```
def 함수이름 (매개변수):  
    함수 내용  
    return 반환할 값 (반환할 값이 있는 경우에만 사용)
```

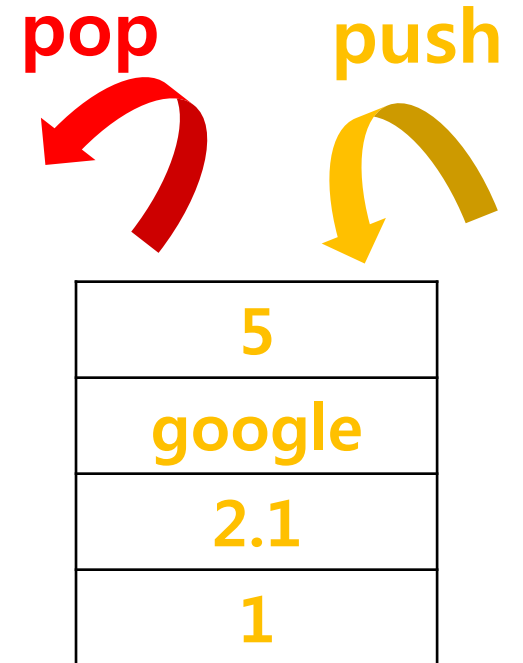
8. List, Dictionary and Tuple

(1) List

```
>>> list_ex=[1, 2.1, 'google']
>>> list_ex[0]
1
>>> list_ex.append(5)
>>> print(list_ex)
[1, 2.1, 'google', 5]
>>> list_ex.pop(3)
5
>>> print(list_ex)
[1, 2.1, 'google']
>>> list_ex.append(2.1)
>>> print(list_ex)
[1, 2.1, 'google', 2.1]
>>> list_ex.count(2.1)
2
```

```
>>> list_ex=[1,7,3,5,8,2]
>>> list_ex.sort()
>>> print(list_ex)
[1, 2, 3, 5, 7, 8]
>>> list_ex.sort(reverse=True)
>>> print(list_ex)
[8, 7, 5, 3, 2, 1]
```

```
>>> list_ex=[1,6,4,7]
>>> list_ex.remove(6)
>>> print(list_ex)
[1, 4, 7]
```



8. List, Dictionary and Tuple

(2) Dictionary

```
>>> dict_ex={1:'google',2:'facebook',3:'gundam'}
>>> dict_ex.values()
dict_values(['google', 'facebook', 'gundam'])
>>> dict_ex.keys()
dict_keys([1, 2, 3])
>>> dict_ex.items()
dict_items([(1, 'google'), (2, 'facebook'), (3, 'gundam')])
>>> dict_ex.get(2)
'facebook'
>>> print(dict_ex)
{1: 'google', 2: 'facebook', 3: 'gundam'}
>>> dict_ex.clear()
>>> print(dict_ex)
{}

>>> dict_ex={'google':1,'facebook':2,'gundam':3}
>>> dict_ex['google']
1
>>> del dict_ex['google']
>>> print(dict_ex)
{'gundam': 3, 'facebook': 2}
```


이름, 핸드폰 번호와 같이 Key가 될 만한 식별자를
이용해서 값을 꺼내야할 때

순서를 지키지 않아도 될 때

Dictionary

순서를 지켜야 할 때

List

8. List, Dictionary and Tuple

(3) Tuple

```
>>> tuple_ex=('a', 'b', 'c')
>>> tuple_ex+('d', 'e', 'f')
('a', 'b', 'c', 'd', 'e', 'f')
>>> print(tuple_ex)
('a', 'b', 'c')
>>> tuple_ex*2
('a', 'b', 'c', 'a', 'b', 'c')
>>> print(tuple_ex)
('a', 'b', 'c')
```

9. 비트연산

```
>>> a=0b110
>>> bin(a<<2)
'0b11000'
```

우측 Shift

```
>>> a=0b110
>>> bin(a>>1)
'0b11'
```

좌측 Shift

```
>>> a=0b110
>>> b=0b100
>>> bin(a|b)
'0b110'
```

OR 연산

```
>>> a=0b000
>>> bin(~a)
'-0b1'
```

NOT 연산

```
>>> a=0b110
>>> b=0b100
>>> bin(a&b)
'0b100'
```

AND 연산

```
>>> a=0b110
>>> b=0b100
>>> bin(a^b)
'0b10'
```

XOR 연산

bin()는 결과를 binary(2진)으로 보여주도록 해줍니다.

** 0b : 2진수 / 0o : 8진수 / 0x : 16진수 표기*

See **you** later.