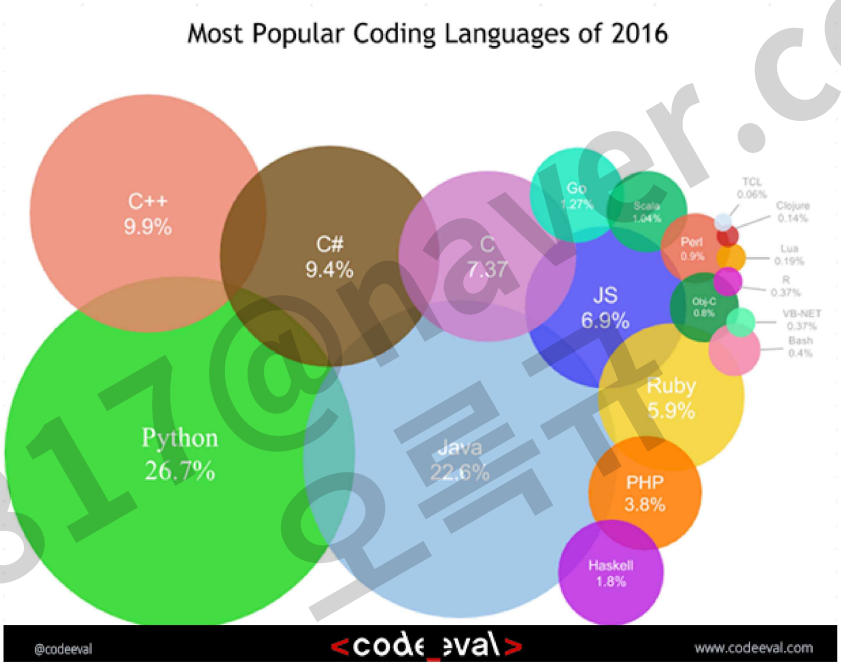
# Week 1. Introduction

# 왜 우리는 Pythonian 이 되어야 하는가

## 유명한 언어

많은 기업들(구글, 나사, 아마존, 시스코, 드랍박스, 월트디즈니 등)도 Python을 사용






## 첫 프로그래밍 언어

공학을 전공하는 거의 모든 학생들은 자의적이든 타이적이든 대학교과목으로서 1학년 혹은 2학년에 프로그래밍 수업 듣게 된다. 아마도 대부분의 선택되었던 프로그래밍언어는 'C', 'C++', 'Java' 였을 것이다.

첫 언어로 언어자체를 배우는 것이 목적이 아닌 프로그래밍을 통해 해결해야 할 과제를 접근하는 방법에 대해서 배우는 입장 이라면 위의 언어들은 적합하지 않다.

> 언어적 문제 보다는 문제해결에 집중하기 위해서는 우선 접하기 쉬운 언어야 함

## 쉬운 언어

복잡한 문법을 필요로 하는 C, C++, JAVA 에 비해 문법이 간단하다.

따라서 배우기 쉽다.

In [1]:

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

4가 있습니다

> 만약 4라는 숫자가 1,2,3,4 라는 값들 중에 있으면 `'4가 있습니다'` 를 출력해줘

## 동적 타이핑

자바는 `정적 타이핑(static typing)` 언어라 변수의 데이터 타입(type), 즉 String인지 int인지 등을 미리 지정해줘야 함

- 한 번 지정하고 나면 변수의 타입은 바뀔 수 없다.

반면 파이썬은 `동적 타이핑(dynamic typing)` 언어라 변수의 데이터 타입(type) 선언할 필요도 없음

- 처음 선언이 된 이후에도 다른 타입으로 바뀔 수 있다.

## 이것저것 따지지 않고 코드를 보면 알 수 있음

**Echo 서버 – 클라이언트 시스템**

**Java Server**

```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;

public class EchoServer {
public static void main(String[] args)
throws IOException {
        ServerSocket serverSock = new
ServerSocket(5502);
        System.out.println(serverSock +
": 서버 소켓 생성");

        while (true) {

            Socket sock = null;

            try {
                sock = serverSock.accept(
);
                System.out.println(sock +
": 연결됨");
                InputStream fromClient =
sock.getInputStream();
                OutputStream toClient =
sock.getOutputStream();
                byte[] buf = new byte[10
24];

                int count;

                while ((count = fromClien
t.read(buf)) != -1)
                    toClient.write(buf, 0, c
ount);

                toClient.close();
                System.out.println(sock +
": 연결 종료");

            } catch (IOException ex) {

                System.out.println(sock +
": 연결 종료 (" + ex + ")");

            } finally {

                try {

                    if (sock != null)
                        sock.close();

                } catch (IOException ex) {

                }
            }
        }
    }
}
```

**Python Server**

```
import socket

HOST = ''
PORT = 50007
s = socket.socket(socket.AF_INET, socke
t.SOCK_STREAM)
s.bind((HOST, PORT))

s.listen(1)

conn, addr = s.accept()
print 'Connected by', addr
while 1:
    data = conn.recv(1024)
    if not data: break
    conn.sendall(data)
conn.close()
```

**Java Client**

```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;

public class EhcoClient {
    public static void main(String[] arg
s) throws IOException {


        Socket sock = null;

        try {
                    //          sock = new S
ocket(args[0], Integer.parseInt(args[1]
));
        //Run→Run Configurationa→Ar
guments

            sock = new Socket("192.16*.
0.9", 5502); //*는 IP주소 비공개처리입
니다.
            System.out.println(sock + ":
연결됨");
            OutputStream toServer = so
ck.getOutputStream();
            InputStream fromServer = s
ock.getInputStream();

            byte[] buf = new byte[1024];

            int count;

            while ((count = System.in.re
ad(buf)) != -1) {
                toServer.write(buf, 0, cou
nt);
                count = fromServer.read(
```

**Python Client**

```
import socket

HOST = 'IP '
PORT = 50007
s = socket.socket(socket.AF_INET, socke
t.SOCK_STREAM)
s.connect((HOST, PORT))
s.sendall('Hello, world')
data = s.recv(1024)
s.close()
print 'Received', repr(data)
```

```
buf);
            System.out.write(buf, 0, c
ount);
        }

        toServer.close();

        while ((count = fromServer.r
ead(buf)) != -1)
            System.out.write(buf, 0, c
ount);

        System.out.println();
        System.out.println(sock + ":
연결 종료");
    } catch (IOException ex) {

        System.out.println("연결 종료
(" + ex + ")");

    } finally {

        try {
            if (sock != null)
                sock.close();
        } catch (IOException ex) {

        }

    }
}
```

# 개발환경이 반이다

## 아나콘다 설치

아나콘다는 Continuum Analytics라는 곳에서 만든 파이썬 배포판으로, 445개 정도의 파이썬 패키지를 포함.

*https://www.continuum.io/downloads (https://www.continuum.io/downloads)*

## Pycharm 설치

Professional Edition 설치. 유료버젼이지만 학생을 대상으로는 Education 의 목적으로 무료로 배포함.

*https://www.jetbrains.com/pycharm/ (https://www.jetbrains.com/pycharm/)*

키보드 단축키와 IDE 테마 -> Darcula 테마는 약간 어두운 색상

## 프로젝트 생성

프로젝트 단위로 소스 파일을 관리

1. *Create Project* 클릭하여 프로젝트 만들기

> - 프로젝트명에 마우스 오른쪽 버튼을 클릭한 후 New 와 Python 순서대로 클릭
> - 소스코드에 마우스 오른쪽 버튼을 눌러 Run 메뉴를 클릭하거나 단축키인 `Ctrl+F9`

## Virtualenv 사용하기

`virtualenv`는 각각의 독립적인 python 환경을 제공함 가상환경을 사용함으로써 다른 python 인터프리터 및 패키지로 사용 할 수 있음

1. pycharm 에서 conda env 생성
   - `conda create -n 가상환경이름 --clone root`
2. 가상환경으로 접속
   - `activate 가상환경이름`
   - `pip install 패키지이름` 으로 가상환경 내에서 필요한 패키지 모듈 설치 가능
3. `deactivate` 로 가상환경 나가기

### 패키지 관리

`pip`는 파이썬에서 기본으로 제공하는 패키지 관리자이다. list 커맨드로 설치된 패키지 목록을 볼 수 있음.

> `pip install (패키지 이름)`

`conda`는 Anaconda 배포판에서 파이썬 패키지(라이브러리)를 설치하고 관리하는 역할을 하는 프로그램이다

> `conda install (패키지 이름)`

### 수업은 Jupyter Notebook으로!

알고리즘이나 프로그래밍을 하는 환경은 `pycharm`에서 진행을 하는것이 일반적임. 하지만 데이터분석과 강의를 위해서는 `코드블록` 단위로 실행하고 결과를 확인 할 필요가 있음.

그래서 우리는 `Jupyter Notebook` 을 사용함

`주피터 노트북(Jupyter Notebook)`은 웹 브라우저를 사용하여 문서와 코드를 동시에 지원하는 개발 도구이다. 웹서버의 형태로 구현되어 있다.



# Python 쌩기초

## 변수(variables)

- 프로그래밍 언어의 가장 강력한 기능 중 하나는 변수를 다루는 능력
- 대입문은 새 변수를 만들고 값을 부여함

## 변수의 선언

`identity = '지구인'` 변수이름 = 값

> 즉, `identity`라는 변수에 `'지구인'`이라는 값을 저장하라는 명령

파이썬에서는 변수 이름의 대문자와 소문자를 구분하기 때문에 주의하여야 한다. 즉, `apple` 과 `Apple` 과 `APPLE` 은 모두 서로 다른 변수

In [2]:

```
#정체와 다리의 수를 출력하는 코드입니다.

"""
여러줄을
한 번에
주석처리할때는 이렇게 따옴표 3개로
내용을 감싸주세요.
"""

identity = '지구인'
number_of_legs = 2 #다리의 수
print(number_of_legs)

#이 아래 줄은 주석처리 되었기 때문에 실행되지 않습니다.
#print('너는 누구니?')
```

```
2
```

## 사칙연산(연산자)

In [3]:

```
a = 5
b = 2

# 더하기 +
adding = a + b

# 곱하기 *
multiply = a * b

# 나누기 /
divide = a / b

# 거듭제곱 **
power = a ** b

# 몫
mock = a // b

#나머지 %
remainder = a % b


print(type(a))
print('a + b =', adding)
print('a * b =', multiply)
print('a / b =', divide)
print('a ** b =', power)
print('a // b = ', mock)
print('a % b =', remainder)
```

```
<class 'int'>
a + b = 7
a * b = 10
a / b = 2.5
a ** b = 25
a // b =  2
a % b = 1
```



**간단한 문제!**

$$48320 - (365 - 5 \times 9) \div 16 \times 987 = ?$$

$$((34 - 3 \times 7)\%5 + 4)^2 = ?$$

In [4]:

```
print(48320 - (365 - 5 * 9)/16*987)
print(((34-3*7)%5 + 4)**2)
```

```
28580.0
49
```

# 불리언(bool) - 참 거짓

파이썬은 참과 거짓을 계산하는 부등식 연산도 가능하다. 파이썬에서는 참과 거짓을 `True` 또는 `False` 라는 값으로 나타낸다.

In [5]:

```
a = True
print(type(a))
print('--------------------')

print('True and True:', True and True)
print('True or False:', True or False)
print('a and !a:', a and not a)
print('bool(0) bool(1):', bool(0), bool(1))
```

```
<class 'bool'>
--------------------
True and True: True
True or False: True
a and !a: False
bool(0) bool(1): False True
```

## 비교

In [6]:

```
print('2 == 2', 2 == 2)
print('5 != 2', 5 != 2)
print('5 <= 2', 5 <= 2)
```

```
2 == 2 True
5 != 2 True
5 <= 2 False
```

# None

- None은 파이썬에서 사용하는 널(null)값이다. 만약 어떤 함수에서 명시적으로 값을 반환하지 않으면 묵시적으로 None을 반환한다.
- None은 예약어가 아니라 NoneType의 유일한 인스턴스이다.

In [7]:

```
a = None
b = 5

print('a is None:', a is None)
print('b is None:', b is None)
```

```
a is None: True
b is None: False
```

# 문자열(string)

In [8]:

```
string = 'python program'

print(type(string))
print('string:', string)

# 문자열 반복
print(string * 5)

# 추후 indexing에 대한 활용을 배울 것 python에서 index의 시작은 `0`부터
print(string[1:4])

# 숫자를 문자열로
number = 7
print('number 변수 타입: ', type(number) )
print('string으로 변환한 number 변수 타입: ', type(str(number)))

# strip()함수로 문자열 버리기
print('http://www.python.org'.strip('http://'))
```

```
<class 'str'>
string: python program
python programpython programpython programpython programpython program
yth
number 변수 타입:  <class 'int'>
string으로 변환한 number 변수 타입:  <class 'str'>
www.python.org
```

> **간단한 문제!**
>
> n을 변수로 하여, n개의 "@"문자를 출력 해 보세요!



In [9]:

```
print('I Like', string)
print('I Like ' + string)
print('I', 'Like', string)
```

```
I Like python program
I Like python program
I Like python program
```

**쪼개기**

In [10]:

```
print(string.split())
print('-------------')
print(string.split('o'))
```

```
['python', 'program']
-------------
['pyth', 'n pr', 'gram']
```

### 긴 문자열

In [11]:

```
######### long string ###############
long_string = '''
This is long string
Very . Long . Wow
'''
print('long_string:', long_string)
```

```
long_string:
This is long string
Very . Long . Wow
```

### 치환

In [12]:

```
######## replace ###########
print(string)

replace_string = string.replace('py', 'mara')
print(replace_string)
```

```
python program
marathon program
```

### % 포맷 사용하기

In [13]:

```
######## formatting ###########
format_string = 'I have %d apples and 3 %s'
print(format_string)

print(format_string %(5, 'bananas'))
print(format_string %(2, 'oranges'))
```

```
I have %d apples and 3 %s
I have 5 apples and 3 bananas
I have 2 apples and 3 oranges
```

### .format 포맷 사용하기

- 문자열을 형식화하는 새로운 방법은 .format 메서드를 이용하는 것입니다.

In [14]:

```
# 이 메서드를 이용하는 방법을 더 선호함
format_string1 = "{0} can be {1}".format("strings", "formatted")

# 자릿수를 세기 싫다면 키워드를 이용
format_string2 ="{name} wants to eat {food}".format(name="Bob", food="lasagna")

print(format_string1)
print(format_string2)
```

```
strings can be formatted
Bob wants to eat lasagna
```

# Python의 변수.... 주의

아래 예제를 보면

In [15]:

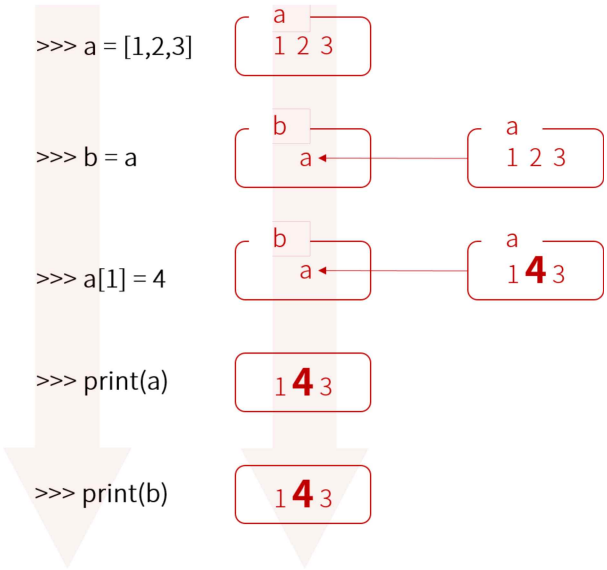
```
a, b = ('python', 'life')
print(type(a), a, b)

a, b = b, a
print(a, b)

print('---------------')
### 변수 복사 ###
a = [1,2,3]
b = a
a[1] = 4
print('a,b :',a, b)
```

```
<class 'str'> python life
life python
---------------
a,b : [1, 4, 3] [1, 4, 3]
```



- b라는 변수에 a라는 변수를 대입하였다.
- 그런 다음 a 리스트의 a[1] (a의 값 뭉텅이 중 두번째) 을 4라는 값으로 바꾸면 a 리스트만 바뀌는 것이 아니라 b 리스트도 똑같이 바뀐다.
- 그 이유는 b안에는 값 [1, 2, 3]이 아니라, a라는 변수를 가지고 있기 때문이다.
- a, b는 이름만 다를 뿐이지 완전히 동일한 것을 가리키고 있는 변수이다.

일종의 pointer의 개념이며, 객체지향 인스턴스의 특징임.




그럼 어떻게 해야할까?

- `a[:]`
- `a.copy()`

In [16]:

```
a = [1, 2, 3]
b = a[:] # b = a.copy()
a[1] = 4
print('a,b :',a, b)
```

```
a,b : [1, 4, 3] [1, 2, 3]
```