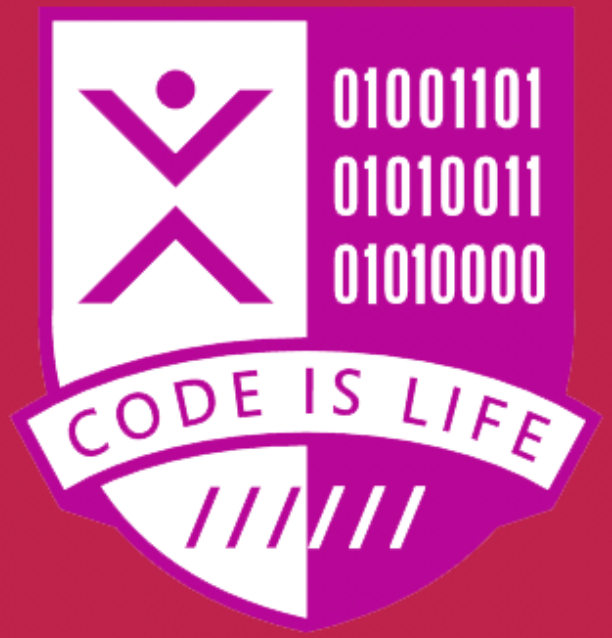
MSP evangelism 2016
장고: 분노의 개발자

사전과제 1
python

Team. 방울뱀



Microsoft

권장사항

- Windows 7, 8, 10
- Linux – Ubuntu 14.x~
- Mac OS – El Capitan ~

- Python 3.4.x 버전 *(※python 2 **사용 금지**)*

Microsoft

Python 소개

- 1989 Guido Van Rossum이 만든 언어
- **Interpreter 언어**
  별도의 번역과정 없이, 소스코드를 실행시점에 해석하여 실행속도가 느림
- **객체지향**
  모듈단위로 프로그램 작성, 객체 하나는 Method와 속성을 가짐

주목 할 만한 점

- 매우 간결한 표현, 배우기 쉽다.
- 수많은 라이브러리들이 존재
- 최근에는 C, java보다 가장 먼저 접하는 프로그래밍 언어
- 빅 데이터, 머신 러닝 등 다양한 분야에서 사용되고 있음.

Microsoft

Python 설치

Mac

- www.python.org/downloads/mac-osx/ 에서 3.4.5 버전 설치
- command+space -> 'terminal' 검색 후 실행 / python3 입력 -> python 3버전 실행 확인

Windows

- www.python.org/downloads/ 에서 3.4.5 버전 설치 (C:₩Python34 경로에 다운 받기)
- 제어판 - 시스템 환경변수 편집 - Path에 C:₩Python34₩Scripts₩과 C:₩Python34₩추가
- cmd나 powershell에서 python 명령 입력 -> 실행 여부 확인

Linux

- terminal에서 python3 실행 시 python 3버전으로 실행됨 (python 실행 시 2버전)
- 최신버전으로 업데이트 하기
  (sudo apt-get update / sudo apt-get install python3)

Microsoft

Python 실행

Mac OS / Linux

- terminal에서 python3 입력하여 실행

Windows

- cmd나 powershell에서 python 입력하여 실행

Microsoft

Python 예제 살펴보기

간단한 사칙연산 예제

- 2+4+1-(4+10)
- 2*3
- 3**2
- 3**3
- 4/2
- 5/3
- 4%3

자료형 확인 예제

- type(3.10)
- type(2)
- type('hello Microsoft')
- type('a')
- type(True)

비교 연산자 예제

- a=10; b=20;
- a>b
- a<=b
- a==b
- a!=b

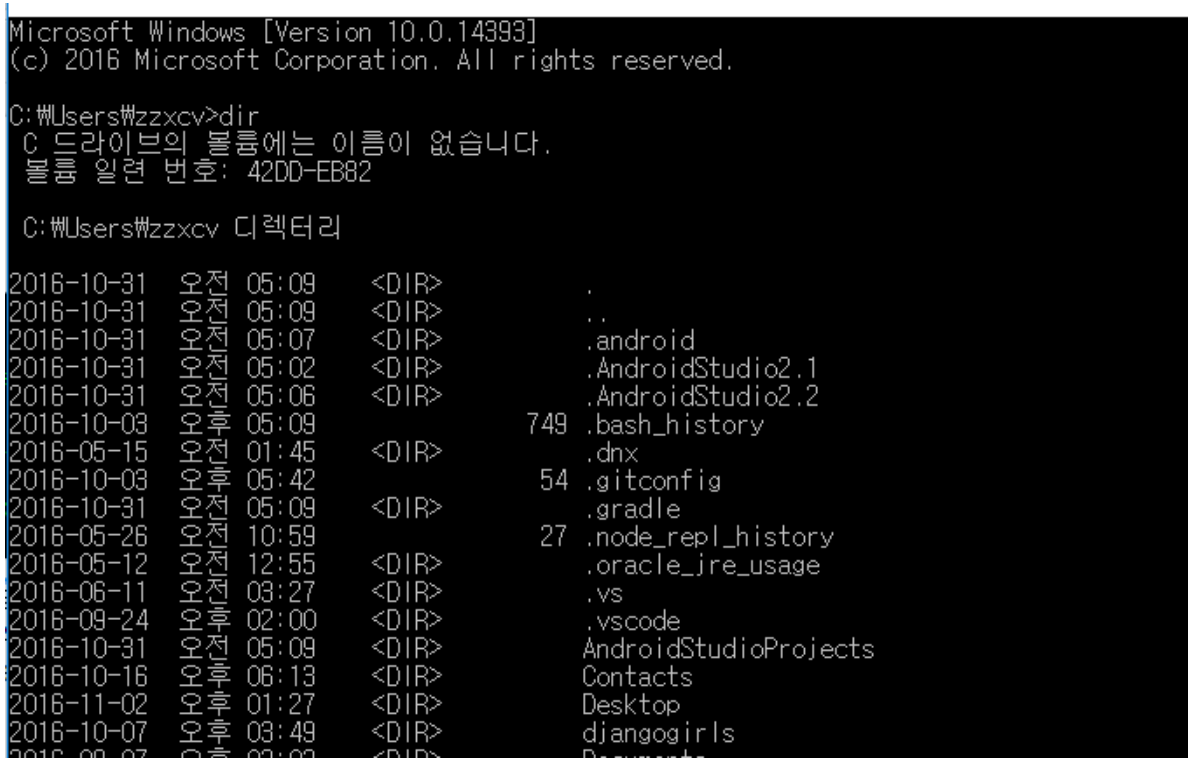
논리 연산자 예제

- a=True; b=False;
- a or b
- a and b
- not a
- c=1;d=0;e=-1;
- c and e
- c or d

Microsoft

CLI (Command Line Interface)

Windows powershell/ cmd



- **dir** : 현재 디렉토리의 파일, 폴더 목록 출력
- **cls** : CLI의 명령결과 내역 모두 지우기 (청소)
- **cd [폴더이름]** : 해당 폴더로 이동
  ex. cd seminar
- **cd ..** : 상위 폴더로 이동
- **mkdir [폴더이름]** : 해당 폴더 생성
- **del [파일이름]** : 해당 파일 삭제
- **rmdir [폴더이름]** : 해당 폴더 삭제

Microsoft

CLI (Command Line Interface)

Linux/ MacOS terminal

- **ls** : 현재 디렉토리의 파일, 폴더 목록 출력
- **clear** : CLI의 명령결과 내역 모두 지우기 (청소)
- **cd [폴더이름]** : 해당 폴더로 이동 *ex. cd seminar*
- **cd ..** : 상위 폴더로 이동
- **mkdir [폴더이름]** : 해당 폴더 생성
- **rm -rf [파일/폴더이름]** : 해당 파일, 폴더 영구 삭제

Microsoft

Editor + python

Visual studio code

- MS에서 출시한 무료 텍스트 편집기. 다양한 언어 지원 및 크롬 디버거 탑재
- Office 제품을 제외하면 MS 최초의 크로스플랫폼 어플리케이션
  (Windows, Linux, MacOS 모두 지원)

설치 및 사용

- https://code.visualstudio.com/Download
- https://code.visualstudio.com/docs/setup/setup-overview 참고
- VS Code로 test.py 코드 작성 후 terminal 혹은 shell에서
  (Linux/MacOS) python3 test.py
  (Windows) python test.py 명령으로 실행 가능

Microsoft

주석 사용

```
3+2 #이것은 한 줄 주석입니다
```

```
'''
이것은
여러줄 주석입니다
'''
```

```
"""
이것도
여러줄 주석입니다
"""
```

Microsoft

기본 문법

조건문 (if ~ elif ~ else)

```
a=3
if (a>5):
    print('a>5')
elif (a==3):
    print('a=3')
else
    print('Hi')
```

```
if (조건1):
    #내용 ..
elif (조건2):
    #내용2 ..
else:
    #내용3 ..
```

반복문1 (while)

```
a=1
while(a<5):
    print(a)
    a=a+1
```

```
while(반복 조건):
    #반복할 내용 ..
```

반복문2 (for)

```
for i in range(0, 5):
    print(i)
for j in range(1, 10, 2):
    print(j)
```

결과를 직접 확인해보세요!

*들여쓰기가 하나의 중요한 문법요소

Microsoft

기본 문법

List

```
test=['Microsoft', 11.27, 100]
print(test[0])
print(len(test))
test[2]='Student partners'
```

Tuple

값의 변경이 불가능한 리스트.
나머지는 리스트와 동일

```
test2=('MSP', 11, 11.27)
print(test2[1])
print(len(test2))
test[1]='forever' #오류발생
```

Dictionary

key와 value로 구성

{key : value, key : value, ..}

```
test3={"Microsoft":1, "Student":10, "Partners":11}
test3.keys()
test3.values()
test3["Microsoft"]
```

Microsoft

기본 문법

함수

```
def rtn_double(num):
    return num**num

def test_function():
    print('just test..')
    a=10
    print(rtn_double(a))

test_function()
```

def [함수이름]([매개변수]):
*   #내용…*

Microsoft

OOP (Object Oriented Programming)

Class

- Method(함수)나 변수를 묶어서 다룸
- . 연산자로 접근
- 여러 객체 생성가능
- Class가 붕어빵틀이라면,
  객체(object는 붕어빵틀에서 나온 붕어빵(들)

생성자

- 객체의 생성과 동시에 호출되는 메소드
- 사용자 필요에 따라서 만들 수 있으며
  필요가 없으면 만들지 않아도 됨
- 주로 객체 초기화 등에 사용

Self

클래스 내부의 객체를 가리킴

```
class Developer:
    def __init__(self, value): #생성자
        self.tech=value
    def printTech(self, tech):
        print(tech) #매개변수로 넘어온 tech
        print(self.tech) #developer 객체의 tech

dev=Developer("MSP Django")
#객체 생성 및 생성자 호출

dev.printTech("Hello Microsoft")
#클래스 안의 함수 (=메소드) 호출
```

Microsoft

OOP (Object Oriented Programming)-상속

상속

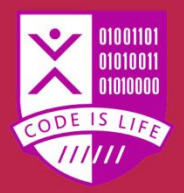
- 부모 클래스의 기능들을 자식클래스에서 그대로 이어받음.
  자식클래스는 부모클래스의 기능들(메소드 및 변수)을 모두 사용할 수 있음
  부모클래스에 이어서 자식클래스에서 추가로 기능 등을 더 구현할 수 있음

- 여러 부모를 두는 다중상속이 가능하지만, 권장하지 않음
  overloading은 불가능하지만, overriding은 가능

- class [클래스 이름] (상속받을 부모 클래스 이름): *#상속받을 경우*
  #클래스 내용..

- class [클래스 이름]: *#상속이 필요없는 경우*
  #클래스 내용..

Microsoft



OOP (Object Oriented Programming)-상속

```
class Mother:
    def __init__(self):
        self.position="mother"
    def getPosition(self):
        return self.position
    def setName(self, name):
        self.name=name
    def getName(self):
        return self.name

class Child(Mother):
    def __init__(self): #overriding
        self.position="child"
    def printer(self):
        print("Hi, i am child")
```

```
mother=Mother()
print(mother.getPosition())
mother.setName("Microsoft")
print(mother.getName)


child=Child()
print(child.getPosition())
child.setName("Student")
print(child.getName())
child.printer()
```

Microsoft

모듈 사용 - 같은 폴더 안에 있는 경우

Module import 하여 사용

모듈 안의 함수나 변수에 . 연산자로 접근

module_test.py

```
in_module="I am in module"

def printer(content):
    print('running printer…')
    print(content)
    print(ending printer…')
```

module_use.py

```
import module_test

module_test.printer('Microsoft Student partners')
print(module_test.in_module)
```

Microsoft

모듈 사용 - 하위 폴더 안에 있는 경우

/test/temp/module_test.py

```
in_module="I am in module"


def printer(content):
print('running printer…')
print(content)
print(ending printer…')
```

/test/module_use.py

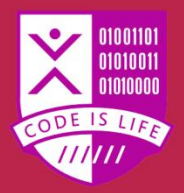
```
import temp.module_test

module_test.printer('Microsoft Student partners')
print(module_test.in_module)
```

/test/temp/__init__.py

```
#내용 없어도 되지만, 필요함
```

Microsoft



모듈 사용 - class 혹은 function만 import

/test/module_use.py

```
from temp.m1 import m1_function
from m2 import m2_function
from m3 import Developer

dev=Developer("Python")
print(dev.getTech())

print(m2_function())
print(m1_function())
```

/test/m2.py

```
def m2_function():
    str="i am m2"
    return str
```

/test/m3.py

```
class Developer:
    def __init__(self, tech):
        self.tech=tech
    def getTech(self):
        return self.tech
```

/test/temp/m1.py

```
def m1_function():
    str="i am m1"
    return str
```

/test/temp/__init__.py

```
#내용 없어도 되지만, 필요함
```

Microsoft

추가자료

Python2 vs. Python3

Python2	print "Hello world" *#print("Hello world")도 가능*
Python3	print("Hello world")

Python2	3/2 *#=1* 3/2.0 *#=1.5*
Python3	3/2 *#=1.5*

**이 외에도 더 많은 차이점들이 존재합니다.*

Microsoft

참고해 보면 좋을 자료들

tryhelloworld.co.kr/courses/파이썬-입문/

www.codecademy.com/

wikidocs.net/book/1

Microsoft