Fastcampus

컴퓨터공학 입문 스쿨

Python Basic_Day2

2017.3.28

Computer Science

숙제

- 오늘 겪은 문제로 컴퓨팅 사고 알고리즘 작성해보기
- jupyter 설치해서 실행해보기

OS, Terminal, Shell

OS - 운영체제 (Operational System)

Terminal - 서버에 접속할 수 있는 콘솔을 구현한 소프트웨어

Shell - 사용자의 명령을 해석하여 전달하는 소프트웨어

사용자 - 입력하고 출력을 받는 존재(나)

가상 개발환경

Project1: Python3.6.0, Django 1.10, ipython, ...

Project2: Python2.7.11, numpy, matplotlib, ...

...

Virtualenv, Autoenv

Virtualenv

https://github.com/pyenv/pyenv-virtualenv

```
$ git clone https://github.com/pyenv/pyenv-virtualenv.git $PYENV

~/.bashrc
$ echo 'eval "$(pyenv virtualenv-init -)"' >> ~/.bashrc
$ exec "$SHELL"
```

```
$ pyenv virtualenv 3.6.0 css-3.6.0
$ pyenv versions
```

Virtualenv, Autoenv

Autoenv

https://github.com/kennethreitz/autoenv

```
$ git clone git://github.com/kennethreitz/autoenv.git ~/.autoenv
$ echo 'source ~/.autoenv/activate.sh' >> ~/.bashrc
$ exec "$SHELL"
```

```
$ touch .env
$ vi .env

echo 'css-3.6.0 pyenv activation'
pyenv activate css-3.6.0

$ cd ./
```

Jupyter Notebook

```
$ pip install jupyter
$ pip list
```

```
$ mkdir notebook
$ jupyter notebook/
```

Hello python!

So, let's try!!

```
print("hello python!")
```

Numbers & Math

`<object> <operator> <object>`

```
print(3 + 7)
print(10 - 3)
print(15 / 7)
print(34 * 100)
```

Numbers & Math

```
print(15 / 7)
print(15 / 5)
type(15 / 5)

print(15 // 5)
type(15 // 5)

print(34 * 100)
print(3 * 2.5)
type(3 * 2.5)
```

Boolean

```
print(3 < 7)
print(10 < 3)
print(15 > 7)
print(34 == 100)

!=
>=
<=
```

Variable

```
print("hello python!")
hello = "hello"
python = "python!"
print(hello, python)
```

```
num1 = 14
num2 = 5

print(num1+num2)
print(num1-num2)
print(num1*num2)
print(num1/num2)
```

```
$ git add .
$ git commit -m "complete python basic day1"
$ git remote add origin https://github.com/{{github_username}}/{
$ git push origin master
```

오늘의 숙제!

반지름(`r=10`)을 선언한 뒤, 이를 이용하여 원의 지름, 둘레, 넓이, 부피를 각각 출력하는 파이썬 파일을 만들어보세요.(`pi=3.1415`)

Syntax

> 문법, 구조, 또는 언어 문장 내에 있는 구성요소의 순서

"나는 입니다 학생" (Syntax Error)
"나는 학생 입니다" (Syntactically Valid)

"Python"5 (Syntax Error)
3.6 * 12 (Syntactically Valid)

Semantics

> 구성요소들의 의미

"아버지가방에들어가신다"

```
room = ['bag']

def comein(where):
    where.append('father')
    where = ['mother','sister']

comein(room)
print(comein)
```

Python - Call by assignment

python은 call by assignment

- immutable object
  - int, float, str, tuple 등이 함수 argument(독립변수)로 활용되면 value로 넘어감
- mutable object
  - list, dict 등이 독립변수로 쓰이면 값이 바뀔수 있을지 없을지 잘 모름

type casting

float(3) --> int to float

int(3.6) --> float to int

str(1) --> int to string

int("12") --> string to int

input

```
name = input("What is your name? ")
print("Hi, ", name)
```

input with evaluation

```
input("How old are you? ")
eval(input("How old are you? "))

equals to raw_input(), input() in python 2.x
```

String Formatting

```
print("I have a %s, I have an %s." % ("pen","apple"))
```

```
%s - string
%c - character
%d - Integer(decimal)
%f - floating-point
%o - 8진수(Octal)
%x - 16진수(hexadecimal)
%% - %
```

Strings

```
some_string = "python"
len(some_string)
```

- index

p	y	t	h	o	n
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

```
some_string[3:5] = "hon"
some_string[1:5:2] = "yhn"
some_string[::] = some_string[0:len(some_string):1]
some_string[::-1] = "nohtyp"
```

but, strings are immutable

```
some_string = "python"

some_string[0] = "c"
some_string = "c" + some_string[1:]
```

String Functions

```
func = "python is easy programming language"
func.count('p')

func.find('p')

comma = ","
func = comma.join('python')

func.split(',')

python_is_easy = "python is easy"
python_is_easy.split()

python_is_easy.replace("python", "golang")
```

List, Tuple, Dictionary

List

```
animals = ['','','']
```

Tuple

```
animals = ('','','')
```

Dictionary

```
user = {'name':'fastcampus','age':'27',city:['seoul','busan','in
```

List detail

```
python_is_easy[start:end]

python_is_easy[0:5]
python_is_easy[:5]
python_is_easy[5:]
python_is_easy[37:-3]

easy_to_learn = python_is_easy[:36]
can_do_this = python_is_easy[37:]
```

List

빈 list를 선언합니다. 선언과 동시에 값을 채워넣을 수 있습니다.

```
lang = ["python", "c", "java", "golang"]
```

lang = []

list에 요소를 추가합니다.

lang.append("python")
lang.append("java")
lang.append("golang")
print(lang)

혹은 특정한 위치에 원하는 값을 추가할 수 있습니다.

```
lang.insert(1, "c")
print(lang)
```

특정 요소를 삭제할 수도 있습니다.

```
lang.remove("golang")
print(lang)
```

혹은 리스트에 있던 값을 빼낼 수도 있습니다.

```
java = lang.pop(2)
print(lang)
print(java)
```

리스트를 정렬하는 법을 알아봅니다.

```
numbers = [2, 1, 4, 3]
print(numbers)

numbers.sort()
print(numbers)
```

리스트를 역순으로 출력하고 싶을땐 이렇게 한답니다.

```
numbers = [2, 1, 4, 3]
numbers.reverse()
print(numbers)
```

특정 값의 위치를 출력할땐 이렇게 합니다.

```
index_of_two = numbers.index(2)
print(index_of_two)
```

리스트끼리 더할 땐 extend를 활용합니다.

```
numbers += [5, 6]
print(numbers)
numbers.extend([7, 8])
print(numbers)
```

Tuple

print("=" * 40)

Tuple은 괄호를 이용해 선언할 수 있습니다.

tuple1 = (1, 2, 3, 4)

tuple은 삭제나 추가가 불가능합니다.

```
del tuple[1]
tuple1[1] = 'c'
```

tuple끼리 더하거나 반복하는 것은 가능합니다.

tuple2 = (5, 6)

print(tuple1 + tuple2)

print(tuple1 * 3)

print("=" * 40)

tuple은 값을 편하게 바꿀 수 있습니다.

```
x = y
y = x (x)

temp = x
x = y
y = temp

(x,y) = (y,x)
```

혹은 함수에서 하나 이상의 값을 반환할 때 사용합니다.

```
def quot_and_rem(a,b):
    quot = x // y
    rem = x % y
    return (quot, rem)

(quot, rem) = quot_and_rem(3,10)
```

dictionary의 선언

```
dict1 = {}
print(dict1)
```

dictionary는 key와 value로 이루어져 있으며, 추가하는 법은 다음과 같습니다.

```
dict1 = {'name': 'foo bar'}
print(dict1)

dict1 = {'korean': 95, 'math': 100, 'science': [80, 70, 90, 60]}
print(dict1)

dict1['english'] = "pass"
print(dict1)
```

요소 삭제는 del을 활용합니다.

```
del dict1['math']
print(dict1)
```

key를 활용해 value를 출력하는 법을 알아봅시다.

```
print(dict1['korean'])
```

key만 출력하는 법을 알아봅시다.

```
print(dict1.keys())
```

value만 출력할땐 이렇게 합니다.

```
print(dict1.values())
```

key와 value를 함께 출력합니다.

```
print(dict1.items())
```

조건문

If

```
if 조건:
        실행문

if 조건1 and 조건2:
        실행문

if 조건1 or 조건2:
        실행문
if not 조건:
        실행문
```

Comparison Operators

```
x == n
x != n

x < n
x > n
x <= n
x >= n
```

else

```
if 조건:
        실행문1
else:
        실행문2
```

else if

```
if 조건1:
        실행문1
else:
        if 조건2:
                실행문2
        else:
                실행문3
```

elif

```
if 조건1:
        실행문1
elif 조건2:
        실행문2
elif 조건3:
        실행문3
...
else:
        실행문n
```

numguess

```
import random


answer = random.randint(1,100)
print(answer)
```

numguess

```
username = input("Hi there, What's your name?? ")
guess = eval(input("Hi, "+ username + "guess the number: "))

if guess == answer:
        print("Correct! The answer was ", str(answer))
else:
        print("That's not what I wanted!! The answer was ", str(
```

numguess advanced!!

how to make it with more fun??