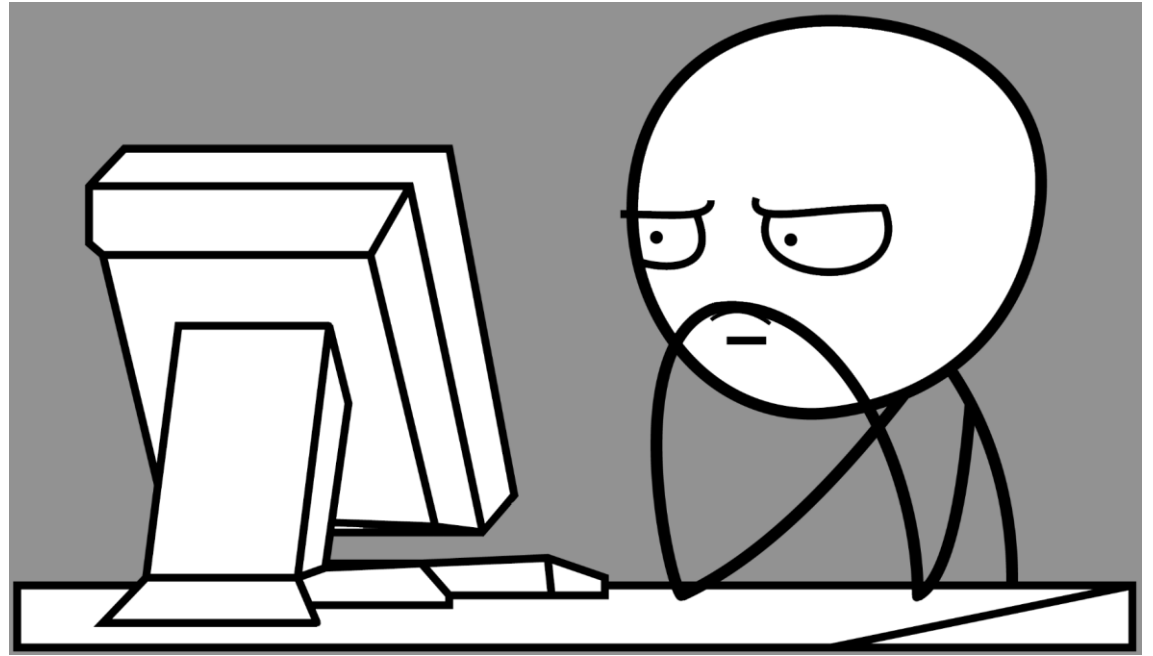


3월 21일 튜터링

하현욱

10분만 잡소리

# 프로그래밍?



# 프로그래밍?

고난의 연속

옆친구는 되고 나는 안되고

코딩에 '코'도 모르는 내가?

왜 해야하지?

# 프로그래밍!

컴퓨터를 실행시켜 무언가 만드는 것이 아닌

문제의 해결방법을 찾아가는 것

=> 문제해결력을 높여준다.

# 프로그래밍!

Excel Online OneDrive Documents Intro to PivotTable Workbook.xlsx

File Edit View Insert Format Data Tools

100% \$ % .0

Client

	A	B	C
1	Client	Project Type	Date Completed
2	Envato	Tutorials	3/30/2017
3	Contoso	Screencasts	10/31/2016
4	Northwinds	Accounting	10/21/2016
5	Schoen-Jacobson	Training	1/25/2017
6	Kozey-Mann	Legal Work	7/5/2016
7	Rogahn LLC	Tutorials	1/10/2017
8	Schoen-Jacobson	Screencasts	8/27/2016
9	Schoen-Jacobson	Screencasts	12/17/2016
10	Rosenbaum-Glover	Tutorials	2/2/2017
11	Russel and Sons	Accounting	4/29/2016

```
tests.py
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64

"""
response = self.client.get(reverse('polls:index'))
self.assertEqual(response.status_code, 200)
self.assertContains(response, "No polls are available.")
self.assertQuerysetEqual(response.context['latest_question_list'], [])
self.test

def test_index_view_with_a_future_question(self):
    """
    test_index_view_with_a_past_question(self)
    test_index_view_with_future_question_and_past_question(self)
    test_index_view_with_no_questions(self)
    test_index_view_with_two_past_questions(self)
    """
    create_question(question_text="Future question.", days=30)
    response = self.client.get(reverse('polls:index'))
    self.assertContains(response, "No polls are available.",
                        status_code=200)
    self.assertQuerysetEqual(response.context['latest_question_list'], [])

def test_index_view_with_future_question_and_past_question(self):
    """
    Even if both past and future questions exist, only past questions
    should be displayed.
    """
    create_question(question_text="Past question.", days=-30)
    create_question(question_text="Future question.", days=30)
    response = self.client.get(reverse('polls:index'))
    self.assertQuerysetEqual(
        response.context['latest_question_list'],
        ['<Question: Past question.>']
    )

def test_index_view_with_two_past_questions(self):
    """
    """
    """
Statement seems to have no effect. Unresolved attribute reference 'test' for class 'QuestionViewTests'.
```

내가 할 일을 줄여준다!

# 프로그래밍!



손으로 8000시간?



컴퓨터로 2시간!

# 그중에서도 왜 파이썬인데?

- 쉬워서
- 쉬워서
- 쉬워서
- 개념을 익히기 쉬움
- 범용성이 넓음





파이썬 어디에 쓰는데?



**django**

# 그래서 이 이야기를 왜 하는데?

- 파이썬 쓰니까
- 프로그래밍 너무 미워하지 마세요...

# 사람들의 착각

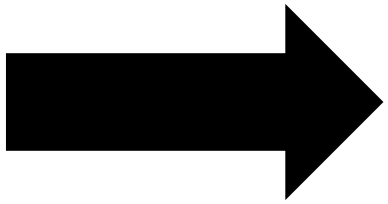
컴퓨터는 똑똑해

컴퓨터론 뭐든지  
할 수 있어

최고다! 컴퓨터!

# 아냐! 아니라고!

- 컴퓨터는 기본적으로 멍청하다.
- 인간의 뇌와 비슷한 계산처리량을 보유한 컴퓨터가 올해 초에 나옴(IBM Summit, 3600억 원)



프로그래머가 똑똑한거야



# 기억하자!

- 프로그램이 생각대로 안움직이면 프로그래머 탓
- 일반적인 프로그램은 실수가 거의 없음(받는 돈이 얼마인데...)
- 결과 오류는 컴퓨터 이상이 아니라 프로그램이 이상한 거
- 그러니까 오류뜨면 내가 짠 코드 다시 보자.
- (남탓하지말고 컴퓨터 때리지 말고)

# 간단한 컴퓨터의 구조

- 설명하는 이유는 나중에(이번 강의는 아님) 알게 됨
- 미래의 튜터를 위해서 간략히 설명함

# 컴퓨터는 비트로 움직임(01010101...)

- 전자신호는 2가지 상태를 가짐 꺼짐과 켜짐
- 이걸 0과 1로 표현
- 0과 1로 숫자를 표현(이진법)
- 회로를 통해 연산자를 만들고
- 연산자들을 쌓아 연산장치(CPU)를 만든다.

# 컴퓨터의 구조





# 컴퓨터의 구조



```
for i in people.data.users:
    response = client.api.statuses.user_timeline.get(screen_name=i.screen_name)
    print 'Got', len(response.data), 'tweets from', i.screen_name
    if len(response.data) != 0:
        ldate = response.data[0]['created_at']
        ldate2 = datetime.strptime(ldate, '%a %b %d %M:%S +0000 %Y')
        today = datetime.now()
        howlong = (today - ldate2).days
        if howlong <= daywindow:
            print i.screen_name, 'has tweeted in the past', daywindow,
            totaltweets += len(response.data)
            for j in response.data:
                if j.entities.urls:
                    for k in j.entities.urls:
                        newurl = k['expanded_url']
                        urlset.add((newurl, j.user.screen_name))
        else:
            print i.screen_name, 'has not tweeted in the past', daywindow
```



# 알고있자!

- 내가 쓴 코드가 0과 1의 형태로 메모리에 저장되어 있다.
- 데이터나 변수들이 0과 1의 형태로 메모리에 저장되어 있다.
- 나중에는 이것들을 신경쓰면서 코딩해야함

끝

튜터링 시작

# 수업과 진행순서가 다를 수 있음

- Print 먼저 다루면 힘들어서 순서를 바꾸었다.

# 변수가 뭐지?

- 아까 말했듯이 메모리 어딘가에는 우리가 사용하는 데이터가 저장됨
- 그렇다고 메모리의 010101들을 우리가 직접 읽을 순 없으니 별칭을 붙임
- 그게 변수!
- $A = 2$ 라고 말하면 메모리 어딘가에 저장된 2라는 '데이터'를 A라 부르겠다는 뜻

# 데이터 타입

## 불리언(Bool)

참 혹은 거짓만 표현  
파이썬 내부에서 True  
나 False로 인식시킬 수  
있다.

### Do it!

True and True, True and False를 쳐보고  
수학에서의 결과와 비교해보자.

# 데이터 타입

정수(Integer)

실수(Float)

정수를 표현하는 방식

이진법으로써 정수를 표현

실수를 표현하는 방식

앞자리 숫자와 승수로 표현

다시 말해 2와 2.0은 다른 형태로 저장된다.

**Do it!**

A=2.1111111, B=3.14, C=2.555 일 때 다음 두개를 비교해보자

$(A*B)**C$  ,  $(A**C)*(B**C)$

# 기본연산자 설명

```
print(3/2) #1.5  
print(3//2) #1  
print(10%3) #1  
print(2**3) #8
```

값이 반환된다.

```
print(True and False)  
print(True or False)  
print(not True)
```

```
print(1==1)  
print(1!=1)  
print(1>2)  
print(1>=2)
```

참 혹은 거짓이 반환된다.



# Print!!

- print()는 표준 '출력' 함수(반환안함)이다. 그만큼 기능이 많다.

```
print('A','B')  
print('A','B', sep='  ')
```

```
print('A', end='  ') #Python에선 개행도 문자로 취급  
print('B')
```

```
print('A+'+' '+ 'B')
```

# Input

- 입력 함수임. 엔터를 누르면 이전에 적은 내용을 String으로 반환함

```
A = input("Type Any String")
```

```
print("String is?", A)
```

```
print("Type is?", type(A))
```

# 이거 헛갈리는데?

- 2(정수) 와 '2'(String)은 다르다.
- 명심하자. 출력이 숫자라고 데이터가 숫자라는 보장이 없다.

```
A = 2  
B = '2'
```

```
Print(A, type(A))  
Print(B, type(B))
```

# 형태 변환

- '2'를 2 로 바꾸는 가장 좋은 방법

```
A = '2'  
B = int(A)  
C = float(A)  
  
Print(A, type(A))  
Print(B, type(B))  
Print(C, type(C))
```

```
A = 2  
B = str(A)  
  
Print(A, type(A))  
Print(B, type(B))
```

# 대입에서 축약형

- ~~(이거 꼭 해야하나 모르겠지만 일단 있으니 알려드립니다)~~
- $A += 2$  는  $A = A + 2$  랑 같음
- 왜쓰냐고? 편하니까. 그것뿐.

# 분기문

Yes or No!

```
Your_ans = input('What is your answer?')
if Your_ans=='Yes':
    print('Yes')
elif Your_ans=='No':
    print('No')
else:
    print('What?')
print('end')
```

## Do it!

상대방에게 'Do you know kimchi?' 를 물어보고  
'Yes'가 나오면 'Okay'  
'No'가 나오면 'It's very delicious'  
나머지에는 'What?'  
이라고 답하는 프로그램을 짜보자.(input 필요)

# Code Block

- If구문과 같이 아래쪽 구문이 항상 실행되는  
게 아닌 경우 공간을 띄워 구분해준다.
- 스페이스 4번을 추천

```
if <condition>:  
    statement1  
elif <condition>:  
    statement2  
else:  
    statement3
```

# 모듈 사용하기

Numpy : 수치해석 라이브러리

Scipy : 머신러닝부터 잡다한 데이터 분석 도구 라이브러리

Matplotlib: 그래프 그리는 라이브러리

Tensorflow: 딥러닝 라이브러리

..... (수없이 많은 라이브러리들)

어떻게 사용? ' .'만 알면 됨!





# 모듈 사용하기

모듈, 어떻게 사용하나요?

1. 모듈을 임포트 한다.(모듈을 가져온다.)
2. 모듈의 함수를 불러와 사용한다.

```
import math  
  
print(math.pi)  
print(math.sin(math.pi))
```

# 내가 원하는 함수와 모듈을 어떻게 찾아?

- 구글검색 (ex random, square root 등등)
- Python Docs(<https://docs.python.org/3/library/>)

# 그래서 코딩을 어떻게 해야할까

- 생각을 하고 코딩하자. 코딩하면서 생각하지 말고.
- 뭘 생각해야 하는데?
  1. 사용해야할 알고리즘
  2. 저장되는 데이터들의 구조
  3. 내가 사용해야 하는 함수

# 제발 좀 달자, 주석!

- 주석은 설명(comment)를 뜻함.
- 프로그래머(심지어 나 자신 포함)에게 코드를 해석할 수 있게 함
- 이거 나혼자 하는데 왜 달아? 이러지 말고 습관으로 만들자.
- 너무 자세하면 별로 안 좋음. 나중에 읽을 사람 생각하고 쓰자.

# 주석 예시

```
def combine_subjects(fix_subjects_with_num, req_subject_codes, sel_subject_codes):  
    '''  
    Function to make combination of subjects list.  
    Structure of input is following;  
    fix_subjects_with_num: ["Class Code", "Class Num"], ....]  
    req_subject_codes: ["Class Code", ....]  
    sel_subject_codes: ["Class Code", ....]  
  
    and output is this  
    [[[Class Code, Class Num], ...], ...Each possible ombination of subject...]  
  
    No possible combination will return []  
    When len(req_subject_codes)>5 or len(sel_subject_codes)>5 will return [] due to block too much computation  
    '''  
  
    if len(req_subject_codes)>10: return "Length of req subs list is over 10" # Check fix_subjects length  
    if len(sel_subject_codes)>5: return "Length of sel subs list is over 5" # Check req_subjects length  
  
    ...
```

# 디버깅

- 디버깅은 매우 중요한 과정

But, 초보자는 뭐가 뭔지 알기도 힘들다

그래서 준비한 초보자가 자주 틀리는 에러!!

(원 바보같은게 다있어 할 수 있는데 다 한번씩 하고 그러는거야)

# 일반적인 에러

```
freind = 1  
print(friend)
```

```
print(math.sin(math.cos(math.pi))  
print('hello')
```

```
for i in range(20)  
    print(i)  
print('finish')
```

```
A = [ i for i in range(10)]  
print(A[10])
```

```
fopen('asdf.txt')
```

```
print('A')  
print('B')
```

```
print('hello')
```

```
a=input('Number')  
print(a%2)
```

```
a=1  
if a=1:  
    print(True)
```

```
print('Don't use '.)
```

```
<tab> print('A')  
<space>*4 print('B')
```

# 특수한 에러


1. 라이브러리 버전이 안맞음
2. 하드웨어가 안맞음
3. 알고리즘 상에 오류
4. 라이브러리 제작자가 라이브러리 잘못만듬(...)

저거 어떻게 암? 어떻게 수정함?



# 에러를 고쳐보자!

```
dl-box@DL-Box: /usr/local/cuda-8.0$ $CUDA_HOME
bash: /usr/local/cuda-8.0: Is a directory
dl-box@DL-Box: /usr/local/cuda-8.0$ vim ~/.bashrc
dl-box@DL-Box: /usr/local/cuda-8.0$ source ~/.bashrc
dl-box@DL-Box: /usr/local/cuda-8.0$ python
Python 2.7.6 (default, Oct 26 2016, 20:30:19)
[GCC 4.8.4] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/usr/local/lib/python2.7/dist-packages/tensorflow/__init__.py", line 23, in <module>
    e>
    from tensorflow.python import *
  File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/__init__.py", line 48, in
  <module>
    from tensorflow.python import pywrap_tensorflow
  File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/pywrap_tensorflow.py", li
  ne 28, in <module>
    _pywrap_tensorflow = swig_import_helper()
  File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/pywrap_tensorflow.py", li
  ne 24, in swig_import_helper
    _mod = imp.load_module('_pywrap_tensorflow', fp, pathname, description)
ImportError: libcudart.so.7.5: cannot open shared object file: No such file or directory
>>>
```



ImportError: libcudart.so.7.5:  
cannot open shared object  
file: No such file or  
directory

# 디버깅할때 유용한 사이트

Google

**GitHub**



**stackoverflow**

# 디버깅의 순서!

1. 오타와 같은 일반적인 에러인지 확인한다.
- ~~2. (운다)~~
3. 에러 메시지를 그대로 구글에 복사하여 해결방법을 찾아본다.
- ~~4. (또다시 눈물을 흘린다.)~~
5. 키워드를 추출하여 다시 검색한다  
(전치사, be동사들을 제외한 에러메세지 단어들과 라이브러리 이름을 매칭한다.)  
Ex) python tensorflow libcudart error
6. 계속 검색한다.
7. Stack overflow, github에 질문을 올린다.
- ~~8. (튜터에게 문의한다.)~~