

09.17 영어음성학 필기

유성음 (voiced sound) : 모든 모음, 일부 자음

vs. voiceless sound : 일부 자음

ex) Chea(p) vs. G / F vs. v

phonology 음운론 vs. **phonetics** 음성학

Phonology는 인지적인 것, 상위적인, abstract 반면 phonetics는 physical하고 하위적인

Ex)

#6.

- speech : 사람이 하는 말
- 한국어 : 음절의 연속 vs. 영어 : stress
- Articulatory vs. acoustic : 사람의 입을 떠난 공기의 흐름

#11. 각 명칭 외우기

- Pharynx : 인강
- Alveolar*****
- Hard palate, soft palate (velum)

#12.

Epiglottis : 삼키는 순간 기도 막아서 음식물이 식도로 넘어가게 함

#13.

Oral tract이 막혀 있고 velum이 lower됨, nasal tract이 열려 있을 때 비음 생성

Velum이 raised가 되면 nasal tract이 막힘 -> 모든 모음, 비음을 뺀 모든 자음들이 생성

Q. 코로 숨을 쉴 때 velum의 위치? : nasal tract이 열림, velum은 lower된 상태임

Larynx가 열리면 무성음, 막히면 유성음(voiced) - (모든 용어의 소리는 유성음과 무성음으로 나눌

수 있음)

#14.

Phonation은 voiced, voiceless를 구분

성대, velum, 혀를 중심으로

09.19 영어음성학 필기

#21. ㅍ ㅌ ㅋ 순의 조음

#22. Constrictor – lips. Tongue tip, tongue body가 해당

CD : 상, 하의 문제에 해당

CL : 앞, 뒤의 문제에 해당

#24. Constrictor가 아랫입술이라고 볼 때, b와 f는 cl의 관점에서 미세하게 다름

Cl 은 각각 constrictor의 관점에서 2, 2, 4개로 분류할 수 있음 ex) lips -> bilabial, labiodental

#25. CD의 관점에서 자음은 3가지 종류 stops, fricatives, approximants라고 이야기할 수 있음

- Approximants : r l w j(y)의 네 개가 해당

- M n 은 stops 소리에 해당

#27. Constrictor, CD CL + velum, larynx까지 고려

시험) Velum raised, larynx의 틈 glottis open, constrictor tongue tip, CD alveolar, CL stop 일 때
나는 리 : t ?

***모든 모음은 constrictor 로써 tongue body만 쓴다

모음과 같은 constrictor를 쓰는 자음 : k

자음 specify하는 문제 시험**

첫번째 소리

<-> 두번째 소리 : 성대에서 바로 녹음

:성대에서 나올 때는 음의 높낮이 정도만 다르고 아 에 이 등의 다른 소리는 입모양에 따라 결정됨을 알 수 있음

#38.

Sign wave : 가장 기본적인 형태 <- frequency, maginitud(진폭)에 의해 결정됨

- 이 세상에 존재하는 모든 sound 포함 signal은 여러 다르게 생긴 사인 sign wave의 결합으로 표현된다. Complex한 세계를 단순하게 표현할 수, 쪼갤 수 있다

● 원리 :

- 세번째 sign wave의 frequency는 첫번째 웨이브보다 상대적으로 높음.
- Magnitude는 첫번째가 가장 높음
- 합 : 가장 밑의 그림은 위의 세 웨이브의 합을 나타냄, 사인 웨이브가 아닌 복잡한 소리에 해당함 (= 복잡한 신호 혹은 소리는 단순한 사인 웨이브들의 합으로 표현될 수 있다)
- 가장 밑의 그림도 1초에 100번 반복된다고 볼 수 있음?
- Simple,/ simplex tone : 사인웨이브 맞춤

<-> Complex/ complex tone : 사인웨이브 아님, simplex의 합? 일상속에서 들리는 소리

*amplitude = magnitude

*오른쪽의 그래프 - X축 frequency Y축 amplitude 의 그래프를 **spectrum**이라고 함.

- equalizing?

*왼쪽 그래프 - X축 : 시간 y축 : value,(단순한 숫자값) voltage?

*왼쪽의 Synthesis 위의 세 그래프에서 밑의 한 그래프로 가는 것

오른쪽 가장 밑 그래프 Spectrum Analyais

#40.

- 0~1까지의 값이 amplitude와 일치
- 한 진폭?의 길이 440분의 1초를 의미

Spectrogram spectrum을 time을 visualize한 것

<-> Spectrum은 시간 개념이 없음, 이것을 시간축으로 계속 늘어놓은 것을 spectrogram이라고 할 수 있음

*spectrum analysis : spectrum -> view spectral slice

* Simplex tone의 가장 낮은 주파수와 pitch가 일치?. 가장 pure한, 작은 sign wave

Q. 모음을 어떻게 만드는가 ?

- Pitch에 해당하는 frequency를 안다면 sign wave를 만들고 다 합하면 나옴??????/

- 소리가 다른 simplex tone으로 구성되어 있음.가장 느린 simplex tone의 ???이 우리 말하는 피치와 일치한다. 일초에 몇번 떨리느냐와도 일치

#42.

- 성대에서 나는 소리를 그대로 캡처한 것, larynx에서 나는 소리
: source
- 소스에서 filter를 어떻게 바꾸느냐에 따라 아, 이소리의 차이

#43.

- 모든 사람의 source의 패턴 : 첫번째 보이는 주파수가 pitch와 일치(f_0 . Fundamental frequency),
= Pitch = Number of vocal cords vibration in ~ seconds
- Harmonyx : amplitude $x_1 x_2 x_3 \dots \sim$ 무한대
- 같은 구간에서 남자가 갖는 배음?의 숫자가 여자보다 많다.

#44.

- 배음의 구조는 그대로 유지, but amplitude의 패턴이 깨짐

#43.

- 웨이브와 스펙트로그램 쌍을 이룸(x축 time, y축 frequency)
- 까맣게 생긴 것이 크기가 센 것 low frequency 쪽으로 가면서 값이 커짐, 에너지 큼.
- 입모양의 필터를 거친 후

09. 26 (목) 영어음성학 필기

#38.

- Spectrum : x 축 frequency / y축 amplitude
- Wave form : 시간 / value

#40.

- Spectrogram : X축 : 시간 / y축 : frequency

#43.

Sign wave들이 배속으로 빨라짐 ?

입모양이 filter역할을 해서 다른 모음의 소리를 낼 수 있음

#45.

- Spectrogram을 볼 때 위의 그림은 저주파에서만 에너지가 높고 고주파로 가면 낮아짐
- 말의 그림도 등간격으로 harmonics가 유지되는 것을 볼 수 있음

#46. Harmonics는 sign wave의 대응으로 이루어져 있음

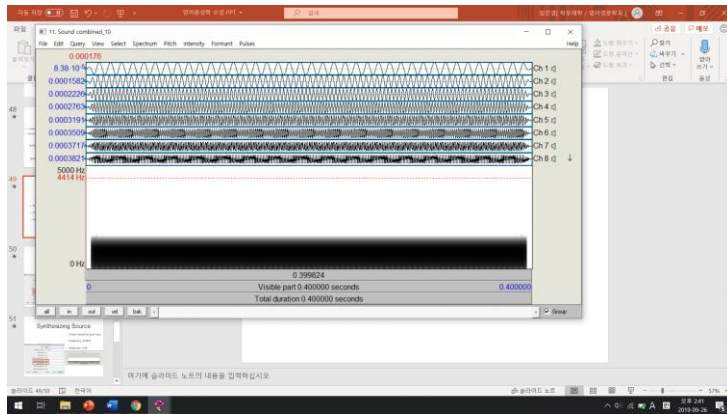
#47.

- EGG: voice cord에서 직접 녹음한 소리
- Mountain과 valley와 같은 모양을 만들어 주는 것이 filter라고 볼 수 있음
- 입모양에 따라 어디에 산맥이 나타나는지가 다름 ex) 아라는 소리를 낼 때 누가 말하던지 간에 똑 같은 패턴이 나타남?
- 첫번째 산맥에 해당되는 주파수 1 formant >> formants를 형성 $f_1, f_2, \dots, \leftrightarrow f_0$.

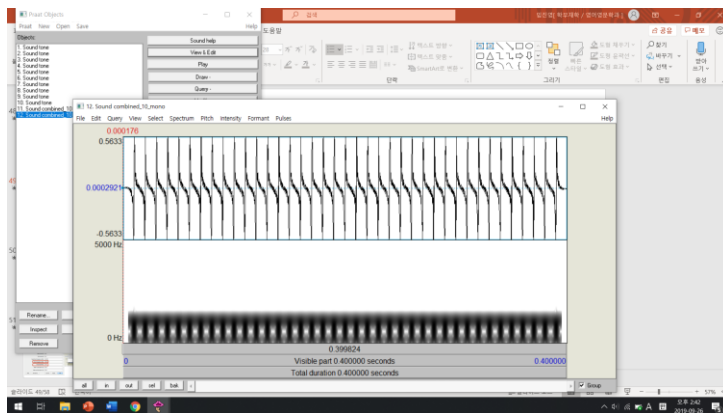
#48.

- Harmonics가 되어서 나는 소리 : 목에서 나는 소리, 기타 소리... cf. 벽을 두드리는 소리
- 기타 소리 : complex tone \leftrightarrow pure tone은 fundamental 만 따서 나는 소리

#49.



- 독립적으로 stereo로 존재하는 상태 <-> mono



- Complex tone의 상태 : 반복되는 패턴의 반복, 반복 주기 – 만들었던 sign wave 중의 첫 번째 것과 일치 , 인지적으로 100hz와 똑같은 인식 (1000hz는 인식하지 못함)
- Sign wave를 무한대로 계속해서 합하면
- 무한대로 합하면 pick 하나 – 000의 반복이 완성 : perse? Train

#54.

- 밑의 가장 첫번째 그림에 도장을 찍는다고 생각하기
 <- 도장 찍는 역할 : filter, 입모양의 역할
- Output spectrum에서 f_1, f_2, \dots 에 해당하는 소리는 peak들에 해당

#57.

그림의 f_1, f_2 가 서로 다른 도장이라고 생각하면

F_1 : 모음의 높낮이, 혀의 높낮이를 결정 (h)

f_2 : front, back.

10.01 영어음성학 필기

- 코딩 – 자동화

- 단어, combine(문법)

- 단어 : 그 속에 의미를 포함, 정보를 담는 그릇 -> computer language의 변수(variable)
- 컴퓨터 문법 : 1. 변수라는 그릇에다가 정보를 넣는 것(assign) - variable assignment / 2. Conditioning에 대한 문법 (~ 일 때 ~하라 , if conditioning) / 3. 여러 번 반복하는 것 (for ~ 문법)
 - ➔ 4. 함수 (어떤 입력(마우스 클릭 etc.)을 넣으면, 내가 원하는 출력(소리가 나옴 etc.)이 나오는 것), packaging.

- Variable (정보)의 종류 : 1. 숫자 2. 글자(문자)

- = : 오른쪽에 있는 정보를 왼쪽에 있는 정보로 assign 한다, 같다는 표시 x

- 숫자

Ex) 1이라는 정보를 a라는 variable에다가 넣는다. a = 1 -> Run 버튼 누르기

- Print (a) -> run 버튼 : 변수를 그 안에 넣으면 그 속에 있는지 무엇인지를 print out 해줌 / 입력의 표시 (__)

(python에서 모든 함수는 누군가가 만들어 놓아야 함? Anaconda는 유용한 함수를 다 모아놓은 것)

Tip) 바깥 쪽 선택하고 b 치면 , below에다가 셀을 만들어줌 / a 치면 above에 셀 만들어줌 / 지우고 싶으면 선택하고 x 치면 됨

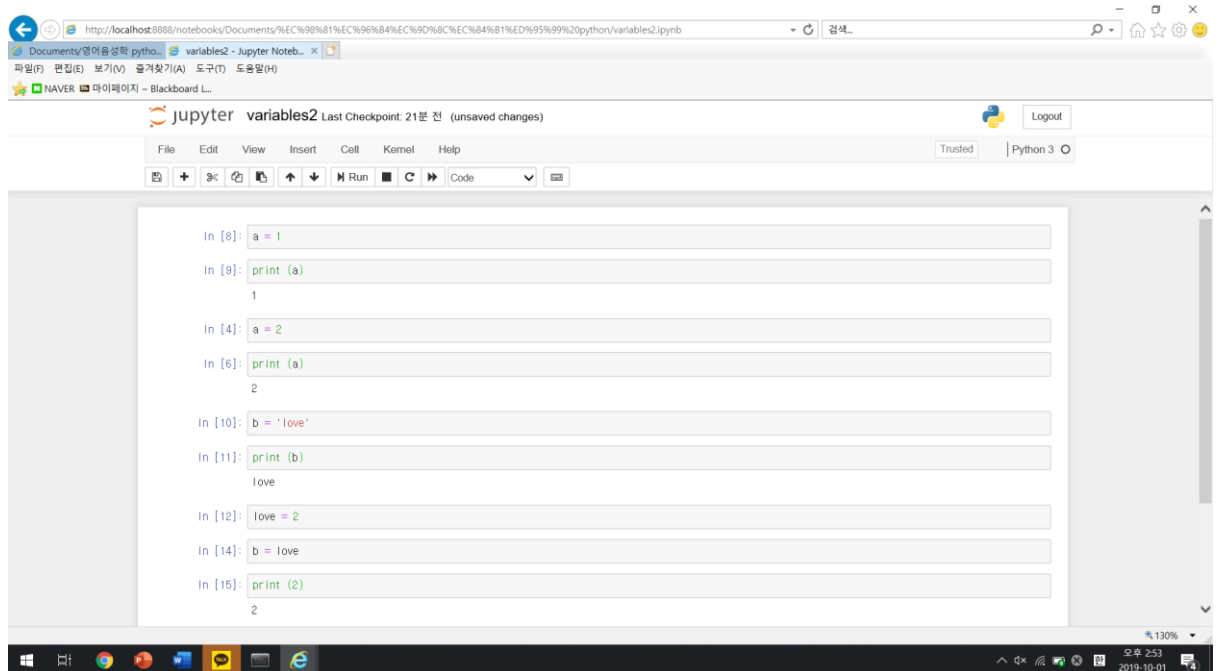
새로운 변수를 만들고 싶으면 이름을 다르게 하면 됨, a =1 하고 나중에 a =2라고 할 때 2가 들어가 있음 / 실행단축키 시프트 엔터

- 문자

Ex) b = love는 틀린 예시 영어를 쓰면 무조건 변수, 숫자를 쓰면 숫자 -> 문자는 '_'로

표시해주어야

Love = 2 love라는 변수에 숫자 2를 넣는 것



A screenshot of a Jupyter Notebook interface. The browser address bar shows a local host URL. The notebook title is 'variables2'. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for file operations, running, and code execution. The code area contains the following cells:

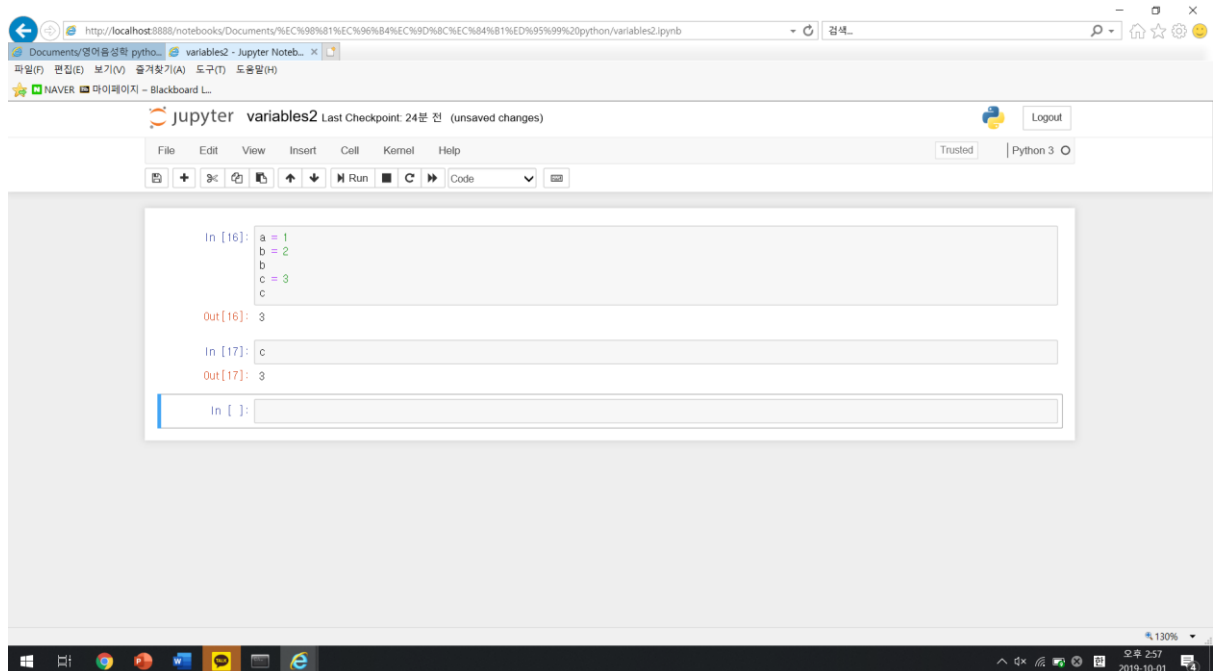
```
In [8]: a = 1
In [9]: print (a)
1

In [4]: a = 2
In [6]: print (a)
2

In [10]: b = 'love'
In [11]: print (b)
love

In [12]: love = 2
In [14]: b = love
In [15]: print (2)
2
```

제일 마지막에 변수명을 치면 printout 해줌



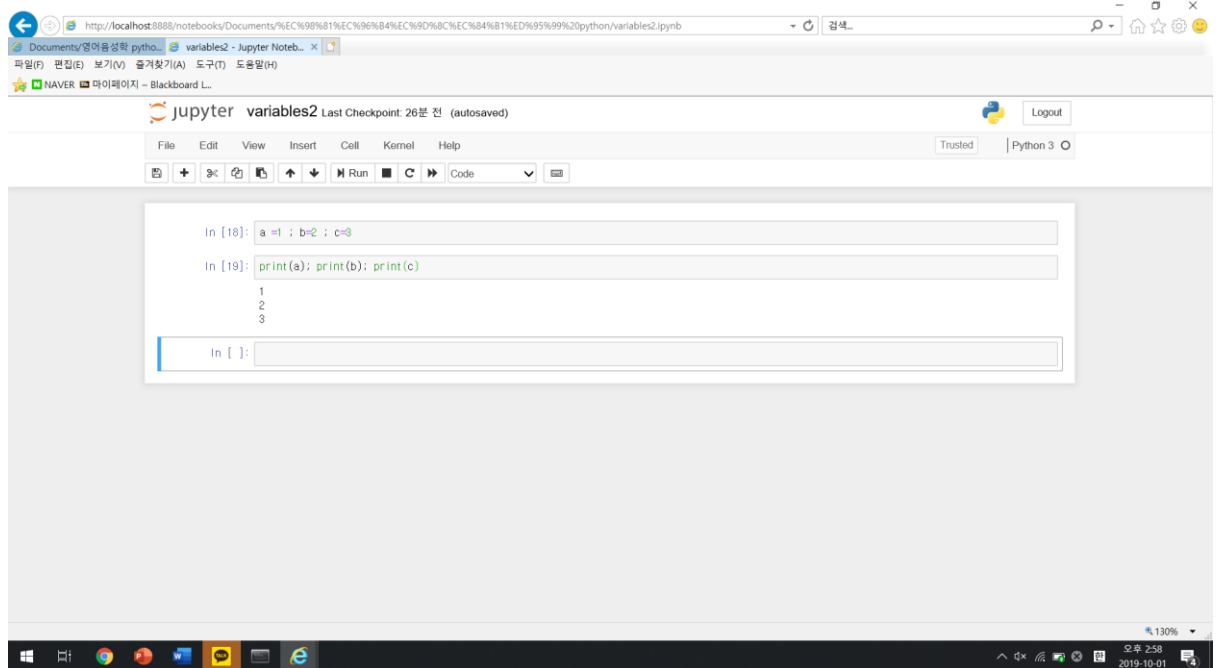
A screenshot of a Jupyter Notebook interface, similar to the previous one. The notebook title is 'variables2'. The code area contains the following cells:

```
In [16]: a = 1
          b = 2
          c = 3
          c
Out[16]: 3

In [17]: c
Out[17]: 3

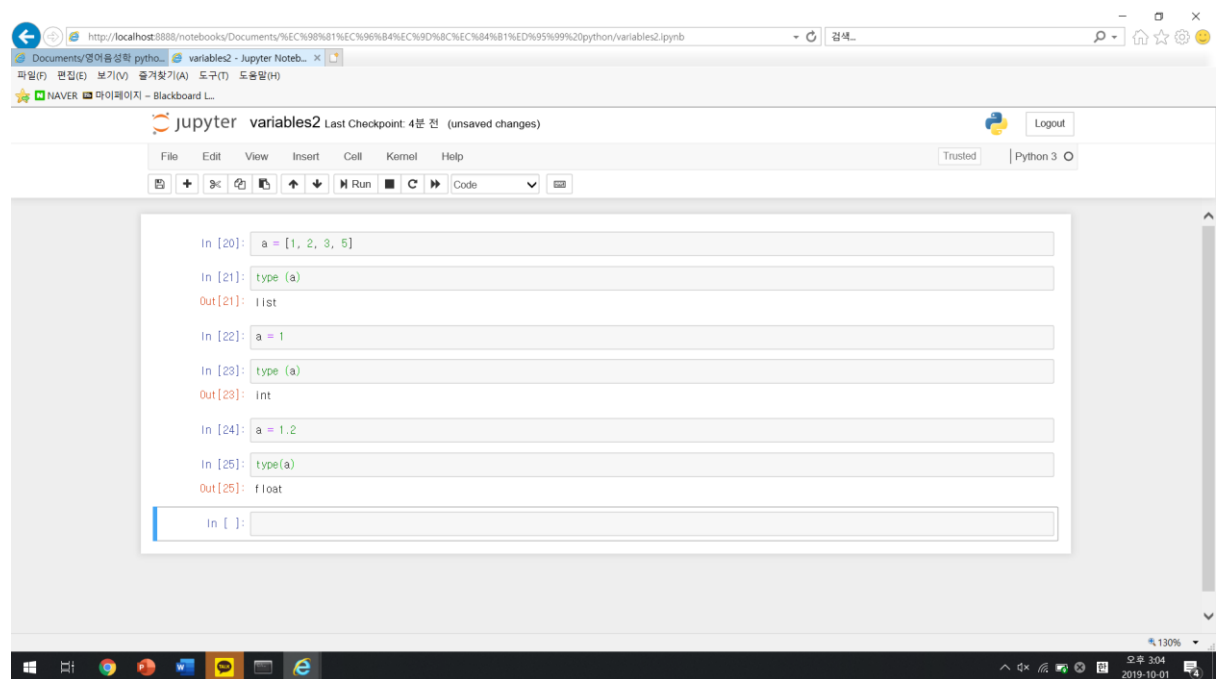
In [ ]:
```

문자정보는 반드시 '___' 혹은 "___" 안에 표시



- List : `[]`를 사용하여 여러 숫자 안에 넣기 -> `type()` 함수를 run : list 함수

Cf. `a=1`, `a=1.2`



A screenshot of a Jupyter Notebook interface. The browser address bar shows a local host URL. The notebook title is 'variables2'. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for file operations, running, and code execution. The code area contains the following cells:
In [26]: `a = 'love'`
In [27]: `type(a)`
Out[27]: `str`
In [30]: `a = [1, 2, 3, 5, 'love']`
In [31]: `type(a)`
Out[31]: `list`
In [33]: `a = [1, 'love', [1, 'bye']]`
The bottom status bar shows the system time as 3:08 PM on 2019-10-01.

- Tuple이 list와 완전히 비슷한 개념이지만, 보안에 더 강함

A screenshot of a Jupyter Notebook interface, similar to the one above. The code area contains the following cells:
In [34]: `a = (1, 'love', [1, 'bye'])`
In [35]: `type(a)`
Out[35]: `tuple`
The bottom status bar shows the system time as 3:09 PM on 2019-10-01.

- 종괄호 : 딕셔너리 / 개수 구분 : 콤마 / 표제어와 설명의 쌍 : (콜론)

http://localhost:8888/notebooks/Documents/%EC%98%B1%EC%96%B4%EC%9D%8C%EC%84%B1%ED%95%99%20python/variables2.ipynb

Documents/영어듣성학 pytho... variables2 - Jupyter Noteb... x

파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)

NAVER 다이렉트 이미지 Blackboard L...

jupyter variables2 Last Checkpoint: 13분 전 (autosaved) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3

Run Code

```
In [37]: a = {'a': 'apple', 'b': 'banana'}
In [38]: type(a)
Out[38]: dict
In [ ]:
```

130% 10:13 2019-10-01