

Part 2

영어뿐 아니라 모든 언어는 자음과 모음으로 되었다

‘ g a p ’ : 그 가 자음. (소리)

sh z - 진동의 유무 (she vision)

j(y) - yearn yaught (요트) / ear과 다른소리 (모음으로 시작)

f v - 목이 떨리는지 안 떨리는지 / f - voiceless. v- voiced

모든 모음은 voiced.

mn ng - voiced / nasal

모든 자음을 voice voiceless 로 나눠보기.

입술 두 개 쓰는 - p b

혀랑 윗니 - th

phonetics 음성학

-인지적인 것) phonology 같게 생각되는 부분 cognitive

-phonetics) 더 물리적이고 늘 차이가 있는 . physical

- a study on sound system

- articulatory) 어떻게 소리가 나올까. 폐가 고기압을 만들어 push 하고 공기를 보내줌. -

성대에서 소리가 만들어짐 - 바람은 파도와 비슷. / whisper 할때는 기문을 완전히 염.

성대에서 기문을 막고 있어서 강한 바람에 의해 펄럭펄럭 움직이는걸 진동으로 느낌.

>어떤 메커니즘에 의해서 바뀌는지 . / ‘아’에서 ‘이’로 바뀌는건 입모양. (혀의 위치. 턱의 높
낮이는 고정됐어도 입속 모양을 바꿔서 소리 바꿀수)

- 영어는 stress 강세가 있다 .

acoustic : 공기가 어떻게 타고 가는지. 물리적.

auditory : 공기가 물결치듯이 타고 오는 소리를 귓바퀴(소리 증폭)를 통해서 듣는 것. / 어떻
게 듣는가. 귓바퀴 타고 들어가 고막 ear drum 이 움직이고 치는 것이 청각세포로 전달.

Articulation

인강 후두(larynx)

alveolar - t d s z m l

epiglottis 꿀떡하는 순간 기도로 가는 길을 막는다.

vocal tract

nasal tract

m 소리르 낼 때 oral tract 는 막혀 있고 nasal tract 은 열려있다.

nasal tract는 velum 이 올라가면 막힌다. - 모든 모음/ 비음을 뺀 모든 자음

코로 숨을 쉴 때 velum 은 lowered, nasal tract 는 열림.

oro - nasal process

phonation process - voiced(막혀서 성대가 진동)/ voiceless (공기가 그대로 통과)

<9/19> part 3

phonetics- 조음 음향acoustics 청각auditory-어떻게 받아들이는가

articulators 크게 세 개.

아 할 때 그 동영상같은 현상이 일어남
larynx가 스피치에서 사용될 때 어떻게 소리가 분류되는지
-voiced (can feel vibration) / voiceless

oral-nasal process in velum
m n ng : velum lowered
숨을 쉴 때 velum이 내려가- 코로가는 길이 생김

Articulatory process in lips- b / tongue tip(혀 앞쪽) - t l s/ tongue body
실질적으로 어떻게 움직이는지 xray 찍은 사진. : 아파-tongue은 메이저로 움직이지 않고 입
술 두 개가 움직임/ 아타 - 혀끝을 침/ 아카 - 혀의 뒷부분이 치고 내려옴

Control of Constrictors

-lips tongue tip/body를 constrictors 이라고 부름.
얼마큼 constriction이 일어나냐 따라 CD를 컨트롤 할 수 있고 / 위치적으로 조금 앞에서할
건지 뒤에서 할건지 location 컨트롤가능 / tongue tip을 가지고 조금 막을건지 많이 막을건
지 앞 뒤로 갈건지에 따라 달라짐
-constrictor는 CL CD에 따라 specify
CL - 앞뒤 lips가 조금 앞으로 가면 b p (bilabial), 조금 뒤로 가면 f v(labiodental)/
tongue body 가 조금 앞으로 가면 yarn 할 때 y 조금 뒤로 가면 g / tongue tip - th 윗
니를 침 , 조금뒤로가면 alveolar 필수도 sh 조금 더 뒤. r 은 더 뒤 (4가지)
CD - 상하 얼마큼 upper part를 hit하나에 따라 CD가 결정됨.
: d - upper part를 완전히 쳐서 stop- p t k / 살짝 틈을 주는거 - s fricative / 완전히
띄우는 approximants - 4개. r l w j / vowels 모음은 막힘이 없음.
CD의 관점에서 자음은 3가지 종류 - stop/ F/ A
Q. ng 은 어디에 해당? - stop 완전히 막혀서 (m n ng 다 stop)
fricative - s z f v th sh 등..

+ velum을 내리냐 올리냐 / larynx을 올리느냐 닫느냐
Q. velum raised, glottis opened , C - tongue tip . CL - alveolar, CD- stop
▶ t
모든 모음은 반드시 constrictor로서 tongue body만 쓴다.
Q. 모음과 같은 constrictor을 쓰는 자음의 예?
>k / velum lowered - ng 성대가 올림. glottis closed

phonemes

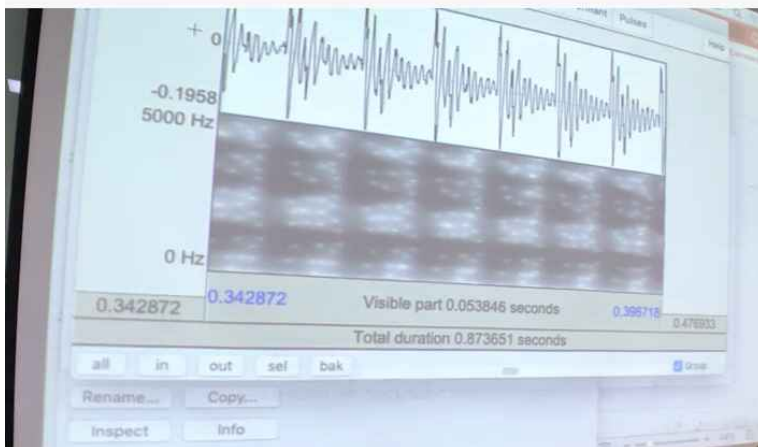
- lips 가 active하게 움직이면 > p b...
- 모든 모음 자음을 specify 할 수 있어야.

Pratt

- object (녹음하기/ 녹음된거 불러오기)
- 파란부분이 자음.
- frequency 의 관점에서 spectrum analysis 할 수 있음.
- duration/ pitch/ intensity measure할 수 있음.
- 회색바탕이 spectrogram.
- 까만띠가 4개 보임. 띠를 formant라고 함. 빨간점 / 쉼 밑에 있는 띠가 첫 번째 formant f1 / f1 f2 가 뭐냐에 따라 모음이 뭔지 결정. 어떤 모음이든지 간에 모음을 구별하는 수치적 지표로서 formant가 쓰인다. formant값이 뭔가에 따라서 이 모음은 뭐다라고 얘기할 수 있음.
- pitch 여성/남성의 목소리가 에 따라서 pitch setting에서 range를 남자 목소리라 치면 65~200. 여자는 더 위로해서 해야 잘 측정이 됨.
- formant 커서를 대면 빨간부분이 수치적으로 얼마지도 볼 수 있음.

vowel acoustics

new - record - view



밑에 파란선이 pitch

확대를 해보면 젤 큰 wave가 반복. 이 반복되는게 larynx 가 빨리 움직이는 가와 같음. 1초에 몇 번 떠는가. 큰 wave 하나가 한번 떠는.

-pitch 가 129 Hz : 1초에 내 성대가 129번 떨린다.

>반복되는 wave하나 선택해서 확대해서 drag해보면 duration이 0.007852: 1초에 이게 127번 들어감. / 1나누기 저 숫자. / 나의 pitch는 내가 1초에 몇 번 움직였냐에 대한 계산.

-new - sound - creat sound as pure tone을 해보면, tone frequncdy를 127 로 입력 (자신의 피치와 같은걸) . > view & edit > 사인 곡선 > 소리 들어보면 (modify - multiply하면 크게 들림 100정도) 높이가 똑같이 들림.

>>127로 파도가 똑같기 때문에 파도의 주기가 똑같기 때문에.

Part 4

constrictors - lips Tongue top Tongue body

velum

larynx

p 라는 소리 specify - lips. (CL - BL , CD - Stop), velum raised, larynx open.

/b/ - larynx closed 로 바뀔 위에서.

/d/- tongue tip. (CL - alveolar , CD - stop)

/z/- tongue tip (CL- CD-fricative)

/n/ - tongue tip (CL- alv CD- stop) . larynx closed. velum lowered. (nasal tract)

/? open

>>시험!

Praat 시험에 나왕

vowel acoustics (젤 중요!!)

repeating event가 어느정도 시간이 되는가 1초동안 몇 번나오는가가 pitch를 제는 방법.

(measure의 단위는 Hz)

sine wave 가 대표적인 반복되는 신호. -

1초라는 구간이 주어지면 1초동안 sine wave가 얼마나 나오느냐에 따라 주파수를 알 수 있음

sine wave)

주파수 frequency - 1초당 몇 번 반복되는가.

sine wave의 크기에 의해서도 결정됨.

> 이 두가지에 의해 형태가 결정됨.

크게 반복되는게 성대의 떨림과 일치.

a를 녹음하고 확대했을 때 성대의 떨림과 일치.

sound quality 는 다르지만. 주파수, 크기?는 같지만.

Source

장치로 성대에서 바로 직접 녹음을 하면
소리가 어떻게 만들어지는가는 입에서 결정.
아, 이 같은 거는 입모양에 의해서 바뀜.

Comple tone in spectrum

Signal .

sine wave 가 리드믹하게 반복되는 가장 기본적인 시그널의 형태.

를 결정짓는게 pitch frequency, 소리의 크기magnitude에 따라 형태가 결정됨.
이 세상에 존재하는 모든 사운드를 포함한 시그널은 여러 다르게 생긴 sine wave의 결합으로
표현된다. (모든 신호는 조금씩 다른 sine wave 의 합으로 표현될 수 있다 .) 주식시장의 흐름도
신호라고 생각할 때 모든 신호는 싸인 웨이브들의 합.

첫 번째 sine wave, 크기도 크고 frequency 는 작음. (천천함) 저음에 해당.

1초에 반복되는게 100번 들어감 . magnitude 쥔 큼

두 번째, 200hz 1초에 200 번 들어감 . 첫 번째 보다 2배 빨라 . magnitude 쥔 작음.

세 번째, frequency 상대적으로 높음. 첫 번째 보다 3배 빠름.

세 개를 합하면(더하기)/복잡한 신호로 만들 수 있다. 복잡한 신호는 단순한 다양한 사인웨이브들의 합에 의해 만들어질 수 있다.

결과적으로 나온 복잡한 신호를 볼 때, 반복되는 주기는 첫 번째 사인과 같음. 1초에 백번 반복. - complex tone

하나하나를 simplex- simplex tone(sine wave) 마지막걸 complex tone- sine wave x

하나의 simplex tone을 더 단순하게 표현한건 오른쪽거.(x축은 frequency y축은 amplitude >>스펙트럼 spectrum) /amplitude=magnitude

왼쪽의 x축은 뭘까 ? 시간 /y축은 그냥 value. 숫자값 시험 !

-오른쪽으로 표현하면 x축은 frequency y축은 amplitude로 변환.

오른쪽의 그래프 (spectrum이라고 함) 으로 변환할수 있어야.

spectrum 은 이퀄라이즈 와 같음.

spectrum 은 어떻게 이루어져 있을까?

우리 주변의 소리는 복잡한 형태. complex tone

위에서 밑으로가는. 합쳐서 밑에걸로 만드는데. 합성 synthesis. 합쳐서 마지막처럼 만드는데 밑에를 쪼개는 걸 analysis. (spectrum 으로 쪼개는)어떤 것들의 합으로 이루어졌는가 보는거.

Practice with pure tone&spectrum

sound- creat pure tone - tone frequency 440(디폴트 값) amplitude 1 - view edit

확대해보면 sine wave

value 들의 최저값 -1 최대값 1 (amplitude를 1로 했기 때문에 / 0 ~1이 A 값)

주기가 1/440 초

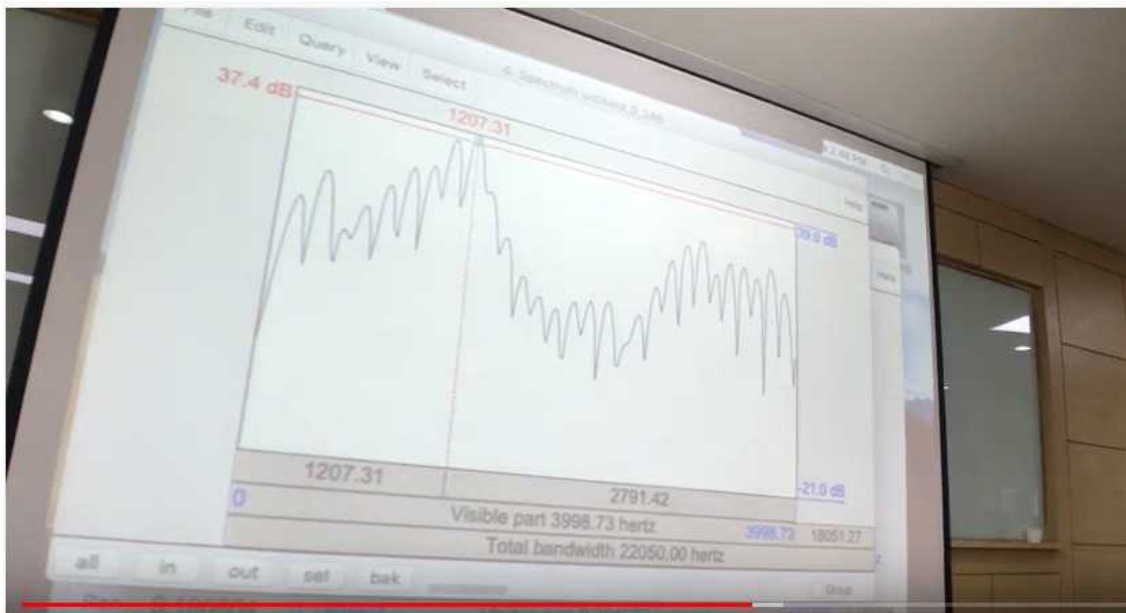
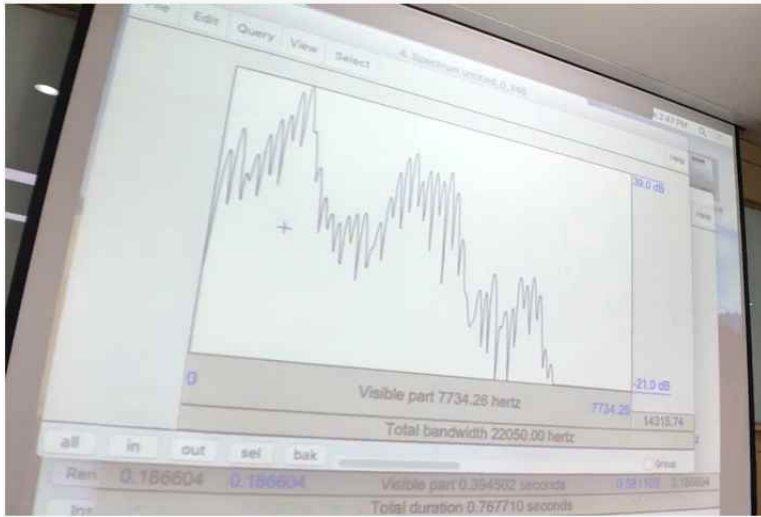
spectrogram : 스펙트럼을 time으로 visualize한거 /시간축으로 계속 늘어놓은거

조금 select해서 spectrum - view spectral slice - 확대

440에 피크가 나옴. x축 frequency축에서 440이 되는 포인트.

440hz를 가진 sinewave를 만들어서 spectral analysis를 해봤더니 440을 가진게 있더라.

아를 녹음해서 spectral analysis 해보면 (x축 frequency y축 amplitude)



아까는 440 하나만 있었는데 많음. / 저주파에 성분들이 많음

-확대해보면 / 점 사이의 간격이 같음.

1206 나누기 9 해보면 133. /

이콜라이즈

- 등간격으로 되어있음. (130, 260, 390...)

133HZ 와 일치.

첫 번째 sine wave의 130은 나의 pitch인 133hz와 일치.

> simplez tone의 합을 여러개 해봤을 때 반복되는 패턴이 여러개. 젤 낮은 주파수의 그것과 일치. /아의 pitch는 젤 작은 simplez톤의 frequency와 일치하는.

*마지막 패턴은 첫 번째 반복되는 패턴과 같음. 젤 작은 simplex frequency 133 와 우리말

의 pitch와 일치. vocal folds가 1초에 몇 번 떨리느냐와도 일치.

+아를 녹음하고 analysis를 해봤더니 단순한 simplex sine 들의 합임.

pitch - 1초에 (반복되는 주기가) 몇 번 반복되는가. vocal fold가 떨리는 주기와 같음(1초에 몇 번 떨리느냐)

모음을 어떻게 만드는가. - pitch에 해당되는 frequency를 안다면 S을 만들고 배수로 넘어가면 됨. (100,200, 300...hz인 sine wave를 다 합하면 아라는 소리가 나오는)

진동수 - frequency . 반복되는게 1초에 몇 번 자리잡고 있는가. 단위는 Hz.

음의 높낮이(pitch) 는 켈 작은 sine wave의 진동수와 일치한다 !

human voice source

모음중에서도 아 이 가 다른 것은 입모양으로 달라지는 것.

성대에서만 녹음을 하면 똑같은 소리.

성대에서 나는 소리를 그대로 캡처했을때의 소리를 source. - larynx에서 나는 소리

filter - 튜브 가 어떻게 달라지느냐 . / source에서 filter를 어떻게 바꾸느냐 에 따라 아 이 소리를 만드는 것.

source를 가지고 spectral analysis를 할 수 있음. > 오른쪽 그림.

아를 녹음했을때보다 훨씬 깨끗. smoothly decreasing 모든사람들의 source의 패턴.

첫 번째 주파수F0 와 피치 일치.

sine wave들의 합

Fo - fundamental frequency. amplitude가 크고 점점 작아짐.

곱하기 2, 3되는게 harmonics라고 부름.

115, 230.. 거리가 여자라고 치면 첫시작이 더 높아 더 음성음성.

/만 hz까지 남자가 갖는 배음의 숫자가 여자보다 많음. (이런식으로 시험)

Filtered by vocal tract

배음의 구조가 안깨졌음. (114, 곱하기 2 3 은) 그대로 유지.

amplitude의 패턴이 깨짐. 지멋대로 왔다갔다 A가 smoothly decrease 하는게 아니라

human voice source 슬라이드

spectrogram 읽는 방법 :왼쪽그림) wave와 쌍을 이룸. x축이 time.(sine wave도) y축은 frequency / 까만게 크기가 센거. 밑에는 까맣고 위로 갈수록 열어짐. low F 저주파 로 갈수록 에너지가 크고 high 고주파 로 갈수록 에너지가 약함 (오른쪽 그림에 low F에서 에너지가 크고 high로 갈수록 작아짐)

아 로 녹음한거 뒤죽박죽인게 왼쪽그림에서 나타남. (filtered by vocal tract)

0926

Part 5

모든 소리는 complex sine (pure tone)

spectrogram 의 x축은 시간 y축은 frequency (시험 !!)

특정구간을 선택해서 spectral analysis / 어떤 성분이 많은 지 보는게 spectrum

vocal tract (입모양) filter 역할을 해 다양한 모음소리를 낼 수 있음.

등간격으로 harmonics가 유지

egg- voice source에서 직접 녹음한거

까만게 블록티어나온. (mountain 과 valley)

모든 사람들의 '아' 의 패턴은 똑같이 나옴.

첫 번째 산맥에 해당되는 주파수 : 첫 번째 formant / 두 번째는 두 번째 formant....

어떤 특정한 음의 높이를 특정한 입모양은 좋아하게 됨. - formant와 일치.

어떤 산맥은 '아'라는 입모양이 좋아하는 산맥들이 있고, 어떤 소리의 높이들은 싫어하는 - valley

harmonics 가 되어 나는 소리 - 기타 소리 .

'라'를 치고 analysis를 해보면 배음으로 됨. - 사람의 성대에서 나는 소리 source 랑 똑같은 음.

기타소리는 - complex tone /

pure tone이랑 높이는 같음.

wave form 은 x축이 시간. y축은 value.

Practice with pure tone & spectrum

y축은 frequency 하는거 시험에 나옴

입 모양. 필터 역할을 함으로써 다른 모음을 소리 낼 수 있다.

Synthesizing Source

- voice source 직접 만들기 - 여러개 만들어서 합하기 - 10개

- 100hz, 200hz ... 1000hz / amplitude를 0.05만큼 낮춰.

- spectrogram 이 가장 낮은 부분에서 에너지가 까맣.

100, 200, 300hz 다 똑같은 음임.

- 다 선택해서 combine to stereo - 하나의 object로 만드는.

- view&edit 하면 10개가 다보임. sine wave 가 달라지는게 보임.

- 들어보면 pure tone 보다는 사람 목소리와 비슷.

>여러가지 sine wave를 + 안한상태. 독립적으로 stereo로 존재. 다 합하면 complex wave.

여러 개가 동시에 존재 - stereo / 그 반대는 mono (더하기로 합하는)

-convert to mono - view&edit - wave 는 complex tone

-반복되는 패턴은 F0와 일치. / 소리가 100hz와 일치. 높이가 같다고 인식.

-이걸 무한대로 1100 1200 계속 합하면 peak 하나 0000 peak 하나 000.... - pulse(하나 뻗어 나온거) train?

- mono wave 조금 선택 (반복되는거 3개정도) - plot spectrum - spectrum slice - 그 순간에서의 spectrum (10 개 gradually decreasing)

*tube. vocal tract 가 들어가면 specturm 은 산맥이 존재 (블록 튀어나오고 들어가는)

도장 찍는 역할을 filtering / 첫 번째 블록 튀어나온게 first formant 두 번째게 second fromant / f0 은 켈 첨에 나오는거의 frequency / formant 는 피크의 frequency

f12 만 있으면 모든 모음 구분 가능. / 서로 다른 모음은 서로 다른 입모양. / 각각의 f12를 그래프로 나타낼 수 있음. - f2 가 x 축 1를 y축.

위에서 밑으로 Front back / f 12 가 입의 위치와 같음.

f1은 모음의 높낮이 결정. 혀의 높낮이. f2는 front back을 결정 backness

new - sound - create sound from vowel editor

영어 아 와 한국어 아 - 영어가 더 back 하고 더 low.

클릭하는 만큼 소리의 길이가 남. 클릭을 하고 drag를 하면 이중모음이 나옴.

“영어뿐 아니라 모음이 어떻게 만들어지는가의 원리. ”

<10/1>

코딩 : 자동화 - 똑같은게 반복될 때

컴퓨터 안에 쓰는 모든 것은 코딩으로 이루어짐

컴퓨터 language 와 사람의 언어는 같음.

language들의 공통점 : 문법, 단어.

모든 언어는 단어가 있고, 이를 어떻게 combine 하나.

단어 - 의미. 정보가 포함. 정보를 담는 그릇. (그릇 하나에 정보로서의 사과를 담으면 이 단어는 사과가 됨)

컴퓨터 단어에 해당되는 것이 변수./ 정보를 담는 그릇이 변수로서 필요하고 기계한테 communication을 하는 문법이 필요.

컴퓨터 문법

1) 변수라고 하는 그릇에 정보를 넣는 것. variable assignment

2) 자동화 기계화라고 생각할 때 , 조건이 필요. conditioning에 대한 문법. if conditioning

3) 여러 번 반복하는 것. for? 몇 번 동안 반복하라

4) 쯤 중요 ! 함수를 배워야. - 어떤 입력을 넣으면 출력이 나오는 것.

많은 명령들(1), 2), 3). 을 한꺼번에 입력과 출력으로 패키징 하는 것

수치적 함수의 예 - 두 개의 숫자를 넣으면 그 시작과 끝까지 합하라 하는 함수.

자동차 - 핸들을 틀면 자동차가 나아가는

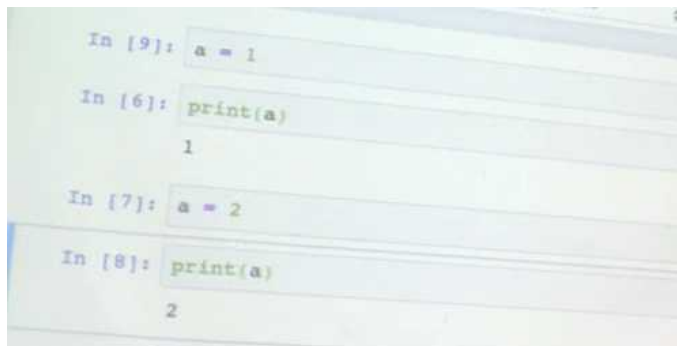
함수 속에 들어가는게 1)2)3) 이 될 수 있고 함수 속에 함수가 들어갈 수도.

anaconda prompt - jupyter notebook

컴퓨터 디렉토리 - 어디에 위치해 있다. / new - python3

변수가 정보를 담는 역할.

정보의 종류 - 숫자, 문자



=은 같다는게 아니라 오른쪽의 정보를 왼쪽에 있는 variable 로 assign 한다. - 순서가 바뀌

면 안됨. / 1이라는 정보를 a 라는 변수에 넣는다. / run을 하고 1이 변수에 들어갔는지 확인 - print(a)

print는 누가 만들어 놓은 함수 - 어떤 변수를 그 안에 넣으면 스크린에다가 그걸 print뿌려주는. / python의 모든 함수는 누가만들어놔야하고 , 내가 만들 수도 있고

함수이름을 치고 () 안에다가 입력을 해주면됨.

print라는 함수의 입력은 a. - 입력이라고 표시해주는건 괄호. > 1을 print 해줌.

1이 떠있는거는 저 라인은 cell이 아님.

a=1에서 다음셀이 또 a=2 를 치면 1은 없어짐 .

파란색으로 바뀔 때 b를 치면 below에 cell 하나 더 만듦.

a를 치면 above 에 cell을 하나 더 만듦. 지우고 싶으면 x

문자를 넣는 방법)

b= love 하면 안됨.

문자는 반드시 quote 해줘야.

실행의 단축키는 shift enter

b = 'love'

print(b)

love

love라는 변수에 숫자 2를 넣는

love = 2

```
In [7]: a = 1
In [4]: print(a)
1
In [5]: a = 2
In [8]: print(a)
1
In [11]: b = 'love'
In [12]: print(b)
love
In [15]: love = 2
In [16]: b = love
In [17]: print(b)
```

b= love

b에다가 love가 가진 변수의 내용(2) 을 넣는 것.

print c 라고 안하고 c만 쳐도 보여줌.

a=1; b= 2; c= 3

```
In [19]: a = 1  
b = 2  
c = 3  
c
```

```
Out[19]: 3
```

```
In [20]: c
```

```
Out[20]: 3
```

`print (a) ; print(b) ; print (c`

1

2

3

세미 콜론을 하면 한줄에 다 넣을 수 있음

```
In [23]: a=[1,2,3,5]
```

```
In [24]: type(a)
```

```
Out[24]: list
```

```
In [25]: a=1
```

```
In [26]: type(a)
```

```
Out[26]: int
```

```
In [27]: a=1.2
```

```
In [28]: type(a)
```

```
Out[28]: float
```

```
In [29]: a='love'
```

```
In [30]: type(a)
```

```
Out[30]: str
```

[] 하면 여러 숫자를 한번에 넣을 수 있음: list.

a에는 list 로서 1235가 들어가 있음.

a 에 들어간 data의 유형이 list 인지 그냥 숫anzi 알려면

> list형태의 타입인지 알려면 함수 type

정보로 들어갈 수 있는게 숫자-int/float 문자

문자를 넣을때는 string

list 에 반드시 숫자만 들어갈 필요는 없음. int, str, float 이든 집합으로 한번에 넣은게 list.

list속에 list도 들어갈수있을까/

- int, str, list(int, str) 가 들어가있는.

[]대신에 () 넣으면 tuple= list 와 같음.

tuple이 더 보완에 강함.

dictionary에 쉼표에 따라 두 개가 들어있는. { } 를 씀.

{ }를 써야 dict이고 쉼표로 하고. : 콜론으로 되있음.

표제어와 설명의 쌍, 콜론으로 엮어줌.

```
In [31]: a=[1,2,3,5, 'love']
```

```
In [32]: type(a)
```

```
Out[32]: list
```

```
In [37]: a=(1, 'love',[1,'bye'])
```

```
In [38]: type(a)
```

```
Out[38]: tuple
```

```
In [39]: a=[1, 'love',[1,'bye']]
```

```
In [40]: type(a)
```

```
Out[40]: list
```

```
In [41]: a={'a': 'apple','b': 'banana'}
```

```
In [42]: type(a)
```

```
Out[42]: dict
```

str 말고도 들어갈 수 있음.

<10/8>

암기 x

variables > string> syntax> numpy

variable - 숫자 , 문자

어떤 변수속에 정보가 들어있을 때,

a = 1

b = 1

c= a + b 하고 실행하면

c

> 2 나옴

-a 와 b 정보를 가져와 c에 담은 것.

a = [1,2]

b = [3, 4]

c = [a{0} + b{0}]

c

>4

여러개가 있을 때 취합해야될 때, a 의 0 번째, b 의 0 번째.

list 에 acces를 할 때 a 중에서 어떤 값 만 가져올 때 대괄호 해서 가져온다.

세미콜론은 두줄에 적을거를 한줄에

a 에 1을 넣었을 때 a의 type 은 'int'

float 이라는 함수는 어떤 변수가 들어오면 float 타입으로 바꾸기. a를 float로 바꾸기. 소수 점이 있는 숫자로 바뀜.

in [19]: float 인데 int로. print (a) 라고 한다면 1.

in [20] 대괄호에 들어가는 것은 index - 내부적 정보를 부분적으로 가져온다

```

In [17]: a = 1; print(type(a))
<class 'int'>

In [1]: a = 1; a = float(a); print(type(a))
<class 'float'>

In [19]: a = 1.2; a = int(a); print(type(a))
<class 'int'>

In [20]: a = '123'; print(type(a)); print(a[1])
<class 'str'>
2

In [21]: a = '123'; a = list(a); print(type(a)); print(a); print(a[2])
<class 'list'>
['1', '2', '3']
3

In [22]: a = [1, '2', [3, '4']]; print(type(a)); print(a[0]); print(a[1]); print(a[2])
<class 'list'>
1
2
[3, '4']

In [23]: a = (1, '2', [3, '4']); print(type(a)); print(a[0]); print(a[1]); print(a[2])
<class 'tuple'>
1
2
[3, '4']

```

2라는 문자를 가져온 것.

a = 123 (' '를 빼고 print(a{1}) 하면, error 이 나옴)

>하나밖에 없으면 0번째 꺼지 1번째게 없음.

list 함수가 1,2,3 쪼개서 list로 만들어줌.

list 화 한후 2번째 걸 가져와라 - 3이 문자로서 나옴.

dict (표제어 : 내용) 의 정보를 access 할때는 abc를 index의 수단으로

“a” 에 해당되는걸 가져와라

in [22] : print(a{0}) 숫자로서 1을 print 한 것./ a 의 첫 번째것은 문자로서의 2. /

```

In [24]: a = {"a": "apple", "b": "orange", "c": 2014}
print(type(a))
print(a["a"])
<class 'dict'>
apple

```

```

In [27]: a=[(1,2,3), (3,8,0)]
print(type(a))
a
<class 'list'>

```

```

Out[27]: [(1, 2, 3), (3, 8, 0)]

```

in [24] : 정보가 pair 로 들어가. 앞부분을 index로 쓴다 (그냥 list 에서는 0 123을

index 로 켜는데 dict에 있는 정보를 access할 때 앞부분 a b c를 index의 수단으로)

표제어가 str 타입으로 되었음 - int가 표제어가 될 수 있을까?

>네. (“a” 대신 “1”을 넣어도 똑같이 apple 이 나옴)

In [27]

a=[(1,2,3), (3,8,0)]

print(type (a))

print(a[0])

```
>(1,2,3)
type ( a[0])
> tuple
```

string -,abcdef

list

다시 돌아가서 f는 -1

젤 첫 번째건 늘 0 /아무리 길어도 젤 마지막건 -1 / count할 필요없음 / 끝에서 몇 번째할 때는 -

range를 해서 여러개의 정보를 가져오고 | 싶을때는 :

첫 번째에서 3번째 직전까지 [1:3]

[1:] 첫 번째에서 끝까지

[:3] 처음에서 3번째 직전까지

list 랑 str이랑 index를 하는 방식. 정보를 가져오는 방법이 같음 = 같은 접근 방법

```
In [6]: s = 'abcdef'
        print(s[0], s[5], s[-1], s[-6])
        print(s[1:3], s[1:], s[:3], s[:])

<class 'str'>
a f f a
bc bcdef abc abcdef

In [3]: n = [100, 200, 300]
        print(n[0], n[2], n[-1], n[-3])
        print(n[1:2], n[1:], n[:2], n[:])

100 300 300 100
[200] [200, 300] [100, 200] [100, 200, 300]

In [4]: len(s)
Out[4]: 6

In [5]: s[1]+s[3]+s[4:]*10
Out[5]: 'bdefefefefefefefefef'

In [7]: s.upper()
Out[7]: 'ABCDEF'

In [10]: s = ' this is a house built this year.ㄴ'
         s
Out[10]: ' this is a house built this year.ㄴ'

In [11]: result = s.find('house')           # index of first instance of string t inside s (-1 if not found)
         result
Out[11]: 11

In [12]: result = s.find('this')            # index of last instance of string t inside s (-1 if not found)
         result
Out[12]: 1
```

len 은 변수내에 있는 정보의 길이. s 의 길이는 6 (abcdef)

len(n) 은 총 3개

s[1] + s[3] +s[4:]

ef를 10번 곱해.

s. upper ()

>대문자로 바뀜

.점 을 하면 실행이됨.
변수를 만들고 점 함수

s. find : s 에 담긴 string 속에서 찾아라.
11번째에서 house 가 시작 (띄어쓰기 포함)
this 는 1번째에서 시작. / 쥬 처음 나오는 index를 찾아라 (this 가 2개니)

```
In [13]: result = s.rindex('this')      # like s.find(t) except it raises ValueError if not found
result

Out[13]: 23

In [14]: s = s.strip()                 # a copy of s without leading or trailing whitespace
s

Out[14]: 'this is a house built this year.'

In [15]: tokens = s.split(' ')        # split s into a list wherever a t is found (whitespace by default)
tokens

Out[15]: ['this', 'is', 'a', 'house', 'built', 'this', 'year.']

In [16]: s = ' '.join(tokens)         # combine the words of the text into a string using s as the glue
s

Out[16]: 'this is a house built this year.'

In [17]: s = s.replace('this', 'that') # replace instances of t with u inside s
s

Out[17]: 'that is a house built that year.'
```

rindex . last index 쥬 마지막 꺼의 index를 해라. 왼쪽부터 count . 여러개중 쥬 마지막 index

s.strip 잡스러운걸 지워주는. / space를 없애고, n도 없애고 순수한 텍스트만 남겨주는

s.split : s에 긴 string이 있는데, 단어 수준에서 작업하고 싶을 때. (' ')을 이용해서 잘라라. / s를 split 함수에 있는 입력을 이용해서 잘라라.

긴 string 을 ' '을 이용해서 잘라라. / 스페이스로 구분 .
> ' ' 는 꼭 적고 , 로 구분 되었으면 ' , ' ??

list 로 되있는걸 복구. 스페이스를 이용해서 token에 들어있는 list들을 붙여라.
자른걸 다시 문장으로 복구하고 싶을 때 join.
' '을 이용해서 token에 들어있는 list들을 붙여라.
' ' 하고 스페이스 대신 , 넣어보기.

replace / string속에 모든 this를 that 으로 바꿔라.

반복해야될 때 loop.(여러개 반복) / conditioning /

#을 붙이고 뭘 적어도 실행이 안됨/ 하나의 cell을 markdown 하면 실행이 안됨. 주석처럼 사용

```
In [2]: a = [1, 2, 3, 4]
        for i in a:
            print(i)

1
2
3
4

In [3]: a = [1, 2, 3, 4]
        for i in range(len(a)):
            print(a[i])

1
2
3
4

In [4]: a = ['red', 'green', 'blue', 'purple']
        for i in a:
            print(i)

red
green
blue
purple

In [5]: a = ['red', 'green', 'blue', 'purple']
        for i in range(len(a)):
            print(a[i])

red
green
blue
purple
```

for loop : 여러개를 여러번 해야할 때

for_ in_ : - 이게 문법

in뒤에 있는 것을 하나씩 돌려서 i 가 받아서 뭘갈 하라

a에 1, 2, 3, 4 / a속에 있는 걸 하나하나씩 해서 i 에 넣고 계속 다 돌아라.

첫 번째 loop가 돌아가서 i가 1을 받아 print했을 때 1이 나옴. 두 번째 loop, a의 두 번째인 2가 i 로 들어감.

in 뒤에 range라는 함수를 쓸 수도 : range뒤에 어떤 숫자가 나오면 list를 만듦. / 4가 나오면 0~3까지 - 4개의 index(0부터 시작) 를 만듦. range가 0,1,2,3 을 만들어줬으니 loop를 돌면서 i 가 쥔 첨엔 0을 받고 , a의 1번째. a의 0 번째걸 print 해라.

그냥 print(i)를 적으면 0,1,2,3 이 나옴. - range 4라고 하면 0~3까지의 index를 만들어.

a의 길이를 적으면 / len (a) 이라고 적으면 a의 길이를. /

각각의 element를 in 앞에있는 변수에 담아둬.

range 함수를 써서 index를 만듦 (0, 1, 2, 3,) 그걸 i가 받아서 a의 몇 번째가 되는.

print(i)를 하면 0,1,2,3 : range 4라고 하면 0~4

len(a) 하면 a의 길이를 적어주는 것.

첫 번째 loop에서 i는 0. range 가 0~6 . 7번의 루프를 돔.

a의 string.

for loop를 쓰지않고 print 하는 방법은

print (a[0]) ...

print (a[1])

...

for s in a :

 print(s)

a 에 있는 list를 개수만큼 for loop돌려라 - 4번. s의 variable 는 계속 바뀔.

for loop 의 내용은 indent를 반드시 해야.

a의 list를 개수만큼 for loop 에 돌려라.

range (4) or range(len(a))

print(a[s]) : a 의 몇 번째

```
In [6]: a = ["red", "green", "blue", "purple"]
b = [0.2, 0.3, 0.1, 0.4]
for i, a_ in enumerate(a):
    print("{}: {}".format(a_, b[i]*100))

red: 20.0%
green: 30.0%
blue: 10.0%
purple: 40.0%
```

```
In [7]: a = ["red", "green", "blue", "purple"]
b = [0.2, 0.3, 0.1, 0.4]
for a_, b_ in zip(a, b):
    print("{}: {}".format(a_, b_*100))

red: 20.0%
green: 30.0%
blue: 10.0%
purple: 40.0%
```

```
In [8]: a = 0
if a == 0:
    print(a)
else:
    print(a+1)

0
```

```
In [8]: for i in range(1, 3):
        for j in range(3, 5):
            print(i*j)

3
4
6
8
```

```
In [11]: for i in range(1, 3):
        for j in range(3, 5):
            if j >=4:
                print(i*j)

4
8
```

길이가 똑같은 list 두 개.

a_ 대신 s를 씀 수업때

enumerate:

a의 list를 번호를 매긴다.

첫 번째 루프에 i 에 0. s에 red. /

그 output 값이 자기자신도 되지만 그것에 대한 번호도 매겨준다.

(변수가 두 개로 받아줌)

i 가 번호고 s가 자기자신인 element

for loop 가 첫 번째 돌 때 i는 0. s는 red. / 두 번째는 i는 1 두 번째는 green.

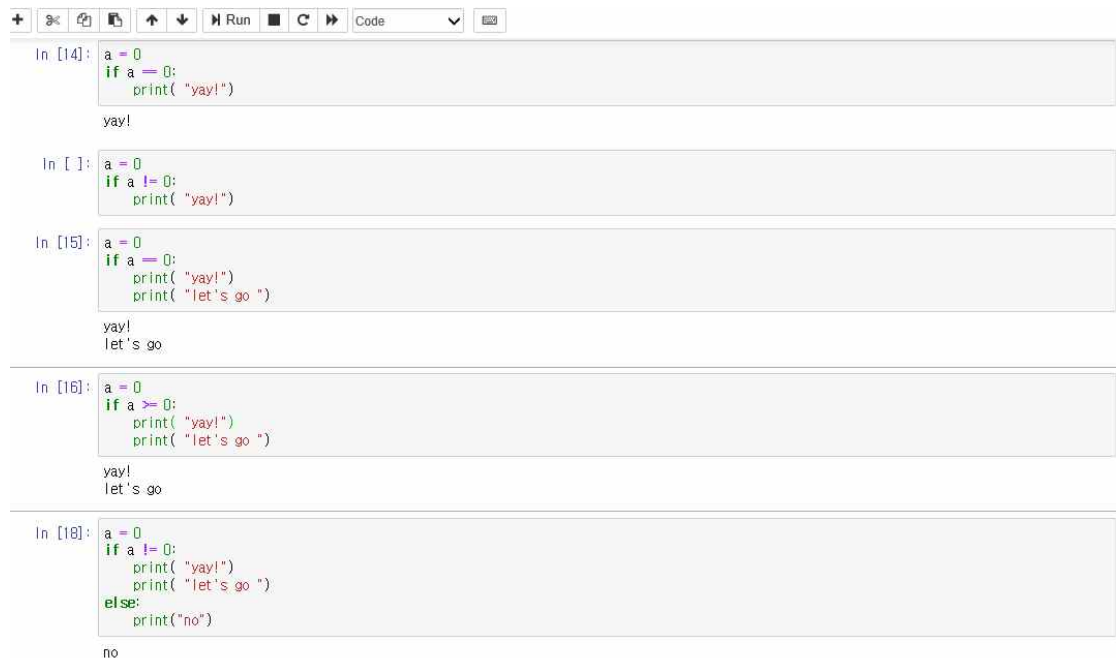
i 는 첫 번째 돌 때 0. 두 번째는 1. green

저런 형태의 format으로 적고 싶을 때 "{i} : {s} " . format에 있는 두 개가 for loop을 돌면서 중괄호속에 각각 꽂힘. s는 red. i 는 0 - b의 0 번째 : 0.2

for loop 이 4번 도는데 첫 번째 for loop 에 i,s 는 각각 0 과 red를 물고 그 다음줄로. / print 하고자 하는 format 대로. /

zip

zip을 함으로써 독립적인 4개가 pair 로 됨. loop를 돌고 첫 번째 루프에서 red, 0.2 가 s, i 로. (사진)



```
In [14]: a = 0
         if a == 0:
             print("yay!")

yay!

In [ ]: a = 0
         if a != 0:
             print("yay!")

In [15]: a = 0
         if a == 0:
             print("yay!")
             print("let's go ")

yay!
let's go

In [16]: a = 0
         if a >= 0:
             print("yay!")
             print("let's go ")

yay!
let's go

In [18]: a = 0
         if a != 0:
             print("yay!")
             print("let's go ")
         else:
             print("no")

no
```

if ~면

equal 사인을 두 개 쓰면 우리가 아는 진짜 equal sign.

a가 0 이면, colon, / ~해라 하는 부분은 indent

a가 0이 아니면 : ! / print가 안됨. / print를 여러개 해도 됨 (indent 필수)

> = : 0이상이면 ,

= > : 순서가 틀리면 안됨.

~하면 이거 하고, ~안하면 이거 한다 - else :

```
In [20]: for i in range(1,3) :
          print(i)
          for j in range(3,5):
              print(i*j)

3
4
6
8

In [22]: for i in range(1,3) :
          for j in range(3,5):
              if j >=4:
                  print(i*j)

4
8

In [25]: for i in range(1,3) :
          if i >=3:
              for j in range(3,5):
                  print(i*j)
```

/ 100번의 루프가 있는데 각각의 루프에 대해 50번의 루프가 돈다면 - 총 5000번이 실행.

range (1,3) : 1부터 2. (3직전까지)

젤 큰 for loop 는 2번 돈다. range 가 1,2 / 1할 때 , 2번돌고. (3,4) / 총 4번 실행

I 가 1일 때 , j loop 가 돌아감. j의 첫 번째 값은 3. 1*3 이 print. 그 다음에 j 는 4. 원래 I는 1이었으니 1*4. I가 2가 되면서 또 2번 되어서 총 4개의 숫자.

중간에 print(i)를 빼도됨. - 이 print는 2번 실행.

print(i*j) 는 총 4번 실행됨.

각각의 I에 대해 j가 두 번 돈다.

3들어올때는 안됨. 총 2번 실행됨.

indent 가 안됐을때 error.

두 번 for loop 되는거 시험 !

part 9

numpy - 라이브러리 .import해서 불러와야함.

중요한 함수들은 그 함수들이 모여있는 라이브러리를 불러와야함.

함수가 들어있는 어떤 라이브러리를 . 패키지.

numpy 라는 패키지 라이브러리 속에 또 작은게 들어있을수 있음, 그 밑에 패키지 이름 A, B 해서 만들고 또 그안에 만들수도.

(점을 함수라고 생각)

numpy 안에 내부적으로 함수가 존재할수 있음. A안에 패키지가 또있다면. 패키지. 내부적으로 안에 있는걸 부를때는 " numby.A.D.어떤함수" . numpy가 젤 위에 있는 상위개념이고 그 안에 또 패키지... 점이 그런 용도로 쓰임.

1) import numpy (np안에 있는 건 쓸수있음)
 numpy-A-D-f를 쓰고 싶으면 import np를 했으니까
 numpy.A.D.f라고 쓰면됨> (너무 복잡)
 >불러올 때 쥔 큰 깍뎡기를 불러오는 방법.
 2) from Numpy Import A (> numpy 에 있는 A를 불러오자) (from을 쓸수도)
 A.D.f (그러면 A를 그때부터 쓸수있음_-)
 from Numpy import A.D (이렇게도 가능)
 >from 으로 import할 수도 있음

사진

In [2]

np라는 쥔 큰거를 불러옴.

import해서 들어오고 numpy를 줄여서 쓰고 싶을 때 as.

np 라고 줄여서 쓰고 싶을 때 as

mat를 불러옴.

matplotlib가 쥔 위에 있고(깍뎡기) 그안에 포함되있는게 - pyplot가 속해있음. 포함되어 있는 subset을 plt로 받아온다.

or

쓸 수 있는 다른 방법중 틀린 것-시험

from mat import pyplot as plt

-그 다음부터 plt쓸수 있음

(점은 포함관계로 되었다 라이브러리에)

numpy는 왜 필요? list 하고 아주 비슷한데 쓰는 이유는 수학적으로 계산도 할 수 있고 해서 처리할 모든 data는 다 np 처리를 해야

[3]

np.empty

empty는 함수 (괄호로 input을 받음)

np라는 쥔 큰 라이브러리 안에 empty 라는 함수.-(그 아래에 subset 은 포함되지 않고)

list가 입력으로 들어감.

2 by 3 로 리스트로 .

가로 세로 행렬 : 직사각형의 숫자의 array

2행 3열 - 똥똥 옆으로

내부적으로 들어가는 것은 datatype을 .dtype을 int로 해서 하나 만들어라

empty 라는 함수는 빈. (비었따 하지만 빈 건 아님)

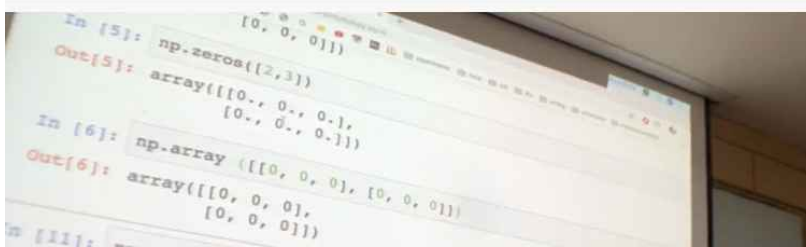
out[2]

numpy속에 들어있는 함수를 이용했기 때문에 숫자가 만들어지는데 계산가능한 숫자

2by 3의 array로 되었음. - 3개짜리 list가 하나가 있고 그게 두줄이 있음.

int : dtype을 int로 해서. 안에 든 숫자는 랜덤한 숫자. 소수점 숫자 아님.

[4]



np라이브러리 속에 있는 zeros라는 함수를 이용. 거기 입력에 list로 2곱하기 3을 넣음.
이 함수는 2by 3의 행렬을 만드는데 0으로 채워진.

```
** [0,0,0]
```

>>그냥 0이 들어가있는 벡터

이걸 2by 3로 만들고 싶을때는

```
** [[0,0,0] , [0,0,0]]
```

2행이 있고 3열이있는 리스트.

>>쓸수가 없음 계산안됨.

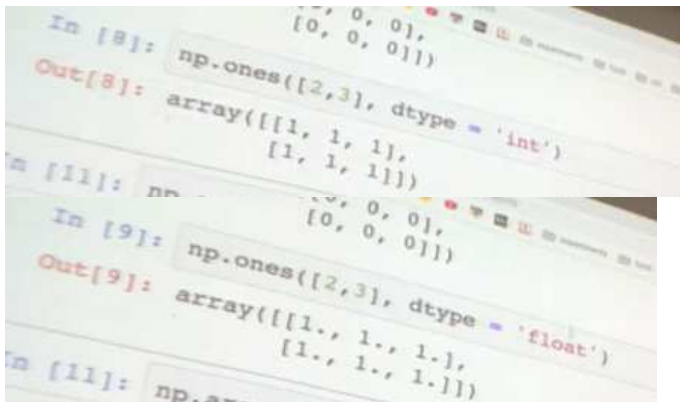
계산이 되는 array로 만드는 방법이

```
np.array ([[0,0,0] , [0,0,0]])
```

리스트에서 array로 바꿔줌

[4]와 같음. zeros함수와 같음.

np.array - list를 array 로 convert해줘라 >위에거랑 똑같은 결과



zeros대신 ones하면 1이 만들어짐. 1 다음에 점 점이 붙은건 int가 아니라 float. (dtype = int 안했을 때)

ones라는 함수가 디폴트로 datatype을 float로.

dtype= 'int' 적어주면 점이 사라짐

float64 로해도 변화는 없음. 소수점 몇째짜리 까지 할까에 대한 정의. precision을 얼마나 밑에까지 할까에 대한 정의. - 오차를 허용하고 싶지 않을 때

64- 정교하고 싶을 때 - 숫자를 길게 하면 메모리를 많이 차지함. 1.00000...

정확도와 데이터 양 반비례.

np에 있는 arange라는 함수는 계산이 될 수 있는 array를 만들어줌. 숫자가 다섯 개. 5라고 쓰는순간 index 5개가 만들어짐.(0부터)

```

In [5]: np.arange(5)
Out[5]: array([0, 1, 2, 3, 4])

In [6]: np.arange(0,10)
Out[6]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

In [7]: np.arange(0,10,2)
Out[7]: array([0, 2, 4, 6, 8])

In [8]: np.arange(0,10,2, dtype='float')
Out[8]: array([0., 2., 4., 6., 8.])

```

arange (0,10) : 0부터 10까지 하면 10은 포함안되고 10 밑으로까지.

(0,10,2) : 2만큼 증가increment하면서 10미만까지 increment- 증가분
data type설정도 할수있음- float, float32, float64도 쓸수있음. - 얼마나 정확성을 높이느냐
와 관련.

```

In [9]: np.linspace(0,10,6)
Out[9]: array([ 0.,  2.,  4.,  6.,  8., 10.])

In [10]: np.linspace(0,10,7)
Out[10]: array([ 0.,  1.66666667,  3.33333333,  5.,  6.66666667,  8.33333333, 10.])

In [15]: X= np.array([4,5,6])
X
Out[15]: array([4, 5, 6])

In [14]: X= np.array([[1,2,3],[4,5,6]])
X
Out[14]: array([[1, 2, 3],
                [4, 5, 6]])

In [17]: X= np.array([[1,2],[3,4],[5,6]])
X
Out[17]: array([[1, 2],
                [3, 4],
                [5, 6]])

In [18]: X= np.array([[[1,2],[3,4],[5,6]],[[1,2],[3,4],[5,6]]])
X
Out[18]: array([[[1, 2],
                 [3, 4],
                 [5, 6]],
                [[1, 2],
                 [3, 4],
                 [5, 6]]])

In [20]: X. ndim
Out[20]: 3

```

X. ndim - 3차원.

[9]

linspace - linear의 준말

linspace(0,10,6) : 0부터 10까지 0, 10포함(arange와 달리). 그걸 총 6개로 똑같이 나누어
준다.

linear space - 차이가 다 똑같음. 첫째에서 둘째로 가는

[11]

np. array : list에서 array 로 바꾸는거

벡터를 만들고 싶을 때,

2by3를 만들고 싶을 때.(행렬을 만들고 싶을 때)

3by 2 만들고 싶을 때.

*벡터면 1차원. 직사각형 2차원. 직육면체로 되면 3차원. 2차원 행렬 두 개 있으면 3차원 대괄호 두 개- 2차원. 대괄호 하나 더 쓰고 콤마하고 같은거 쓰고 대괄호닫기. >3차원 -대괄호 3개.

차원

```
In [21]: X.shape
Out [21]: (2, 3, 2)

In [22]: X.dtype
Out [22]: dtype('int32')

In [23]: X.astype(np.float64)
Out [23]: array([[[1., 2.],
                  [3., 4.],
                  [5., 6.]],
                [[1., 2.],
                  [3., 4.],
                  [5., 6.]])
```

```
In [25]: np.zeros_like(X)
Out [25]: array([[[0, 0],
                  [0, 0],
                  [0, 0]],
                [[0, 0],
                  [0, 0],
                  [0, 0]]])
```

```
In [24]: X*0
Out [24]: array([[[0, 0],
                  [0, 0],
                  [0, 0]],
                [[0, 0],
                  [0, 0],
                  [0, 0]]])
```

x. shape

(2,3,2)

2곱하기 3곱하기 2 의 차원이다. (3개니까 3차원인데

-3개니까 3차원인데 3곱하기 2가 2차원짜리 직사각형. 이게 두 개가 있다. 쉘 큰 괄호속에있는게 두 개. (가 첫 번째 나오는 2) 그다음괄호속에 있는 3개의 리스트 안에 2개의 숫자.

X. dtype

: x속에 나오는 숫자가 어떤 type인가 >int /32 precision과 관련

타입을 바꾸고 싶을 때 astype 이라는 function .

-np.float64로 바꿔줘라

np. zeros_like / x를 만들었는데

모든 형태를 유지한채 숫자를 0으로 바꿔주라. : zeros like

X*0 해도/ 이제 계산이 가능하니 똑같이 만들어짐.

#data = np. random ~~

np 라는 큰 라이브러리 속에 random 이라는 sub패키지 속에 있는 normal이라는 함수를 쓴다.

from np import random. normal 도 가능

normal함수 - normal distribution을 만들어주는. 정규분포. 종모양 데이터를 만들어주는.

종 모양을 만들기위해 필요한게 두가지 정보 - mean값(평균) 과 얼마나 뚱뚱한지./ 0 -

mean 1- 뚱뚱 100개의 데이터 / random한 데이터. 값(나올때마다 다 $\frac{1}{100}$) / 총 백개의 데이터가 나옴.

data. ndim 은 1차원. data의 디멘션은 일차원- 대괄호 하나

data. shape- 총 100개의 숫자가 나옴.

100개의 데이터를 가지고 정규분포를 가지는지 플레팅을 할 수 있음.

mat라는 라이브러리 속에있는 plt라는 서브패키지를 import해올 것. plt - subpackage 속 세 속에 hist라는 함수를 씀.- 벡터 array 계산할수있는 data를 입력으로 하고

히스토그램: 늘 옵션에 bins 바구니를 총몇개로 할건가 정함. 바구니를 10개로 만들어 0기준 한쪽으론 -. 한쪽으론 + 값. 값들이 점점 쌓이는.

x축에다가 데이터값을 공으로 생각하고 던지면 바구니 속에 쑥 들어감. 위에 공들이 쌓임.

그렇게 그래프 그리는데 히스토그램.

그림 - 바구니 총 10개. range 속에 들어가는 값들 . y값에 해당되는 건 무조건 0을 포함한 정수값 .값들을 다 합하면 100개- 시험

<11/5>

Phasor

1초동안 얼마나 숫자가 뽁뽁하게 값들을 담을건가 - sampling rate

SR을 만으로 한다고 규정하면 1초동안 총 만개의 숫자를 담을거다.

1초에 얼마나 할 때 hz라는 단위.

list상에서는 어떠한 계산도 안되기 때문에 numpy 에 담는다.

np.array []

다차원의 array를 만들 수 있다. 1차원이면 벡터 2차원이면 직사각형 3차원이면 volume

numpy - list에 숫자 담는것보다

np. array [] 하면 np형태의 데이터로 바뀐다.

Sound

다양한 pure tone의 합으로 복잡한 사인.

sinusoidal-사인 코사인처럼 곡선으로 생긴 phasor- sinusoidal function을 만들어내는 것 싸인하고 코사인에 들어가는 입력 - sin(입력)입력값은 degrees 가 들어가면 안되고 radians 가 들어가야함.

0부터 100파이 까지 사인 코사인 그래프를 그리면 총 몇 번의 반복? - 50번. 2파이가 반복되니까 .

파이는 무리수

$\sin(\pi/4)$

오일러 공식

e도 무리수

sin - 입력변하면 출력을 해주는 함수

sin cos처럼 e 저거도 함수 - e, I 는 fix 되었음. radian값이 변함으로써 어떤 값이 뺄어지

는.

세타만 변하면 어떤 값이 나오 e, I는 숫자. 세타에 파이 넣으면 다 숫자 니까 숫자값이 나오.
오일러 평선의 값들은 어떻게 그럴까.

복소수 평면

각도값을 그려서 만들고 몇 콤마 몇 하면 $0.3 + 0.8i$ 이렇게 나오

벡터의 정의는 숫자열 . 벡터값으로 표현이 됨.

t세타가 바뀔때 따라 원을 따라 .. $0 \sim \pi \sim 2\pi \sim$. 계속 도는.

projection : x축 a에 project를 하면 위에서 보는. > ---- 가로로 왔다갔다

y축에 project를 해서 b 보면 세로로 왔다갔다.

실수의 관점에서만 보겠다 하면 위에서 보면 됨. 1에서 시작해서 왔다갔다

허수만 보겠다 하면 0에서부터 위아래로 왔다갔다

실수 - \cos 허수 - \sin

\cos 그래프는 1부터 시작 \sin 은 0부터 시작

실수 - 1부터 시작 . 허수 - 0부터 시작해서 위 > 아래 0부터 올라갔다가 내려가는.

인풋이 세타 레디언 각도값.

pure tone에 넣는 입력값중에 frequency (1초에 몇 번 왔다갔다 하는가)

사인 세타를 쓰는 각도값이 변한다고 할 때 시간의 개념이 안들어감 -각도값이기 때문에.

시간의 개념이 들어가야 1초에 몇 번 왔다갔다냐해서 소리의 높이.

각도개념, 초개념을 같이 넣어줘야 진정한 소리가 나옴. 소리는 반드시 시간의 개념이 들어있어야.

sampling rate : 1초에 총 만개의 숫자로 표현.

#

<11/7>

크게 matplotlib 가 있고

import matplotlib.pyplot

해도 똑같은거.

젤큰 라이브러리 밑에 sub라이브러리.

#parameter setting

time 은

0부터 2 파이까지 만들어짐.

part 11

numpy 라이브러리를 import.

첫 번째 줄 . 플래팅 할 때 쓰는 라이브러릴. 큰 라이브러리 이름은 matplotlib . 큰 라이브러

리 밑에 sub라이브러리. - 이것과 똑같은 역할) import matplotlib.pyplot

from을 쓰며 그 속에 pyplot을 부름.

Phasor에서 변수들을 parameter setting

변수에다 값을 담아 놓음.

amp 나 sr dur을 바꿀 때 이것만 고치면 결과값이 쉽게 바뀜 .이런게 parameter time을 왜 만들까 - 사인 코사인 오일러 평선 등은 입력값이 각도값임 (degree 안돼 radian만) . time이 필요한이유는 각도값 만으로는 실체의 소리를 만들수없음. 사인 코사인으로 플레팅을 했을 때 어떻게 플랫 되는지 해볼거임.

sr은 1초에 몇 개가 들어나는지 - 1초라는 말이 들어가면 time과 관련있는거.

2차원 - 머 콤마 머 해서 2차원의 숫자가 들어가는 숫자가 두 개있는 백타.

part 12

amp sr (얼마나 정보를 촘촘하게 할건가/숫자들이 1초동안 얼마나 나오는가) freq (얼마나 1초동안 왔다갔다 할까/shape 반복이 얼마나 왔다갔다 하는가)

어떻게 amp를 장착시킬까 - 증폭. / complex wave 소용돌이 치듯이 생김. 그게 얼마나 커지는가.

그림

sr=10hz. 주어진 숫자가 10개. 이걸로 100번 왔다갔다 하는 걸 표현 할수 있을까

아무리 동그라미를 아껴도 5번의 왔다갔다 까지밖에 안됨.

주어진 숫자에 개수의 표현할 수 있는 주파수는 반밖에 안됨 멕시머이.

1초에 웨이브를 두 개 만드는건 쉬움.

숫자가 주어져 있으면 무한대로 표현할 수 없음.

sr의 반에 해당되는것만 표현가능. Fre = 5hz 가 Maximum.

CD가 담는 음질은 sr = 44100Hz 1초에 주어진 숫자가 44100. 사람이 들을 수 있는 가청 주파수가 2만. CD 음질이 저기에 고정.

NF 는 22050 hz. - 피치로 쳤을 때 아주 높은 소리. /사람이 그 위로는 들을 수 없음.

+유선전화도 소리가 디지털로 선을 따라감. 숫자값들이 움직임. 얼마나 뻑뻑하게 전송을 해야 하는지가 중요. sr을 얼마까지 해야하는가가 중요. / 유선전화의 sr은 8000hz. 4000까지 표현가능.- 낮음. / 들을 수 있는 최대. /사람이 헛갈릴대 많음 엄만지 딸인지 말소리는 다 들리지만.

휴대폰은 16000hz. NF 8000 까지 .

sr의 half 에 해당되는 NF까지가 숫자상으로 표현할 수 있는 fre의 Max. - CD음질의 sr를 가지고 박쥐 소리를 녹음하면 저기 들어가지 않음. 고주파가 담기지 않음.

초음파는 fre에 해당하는 얘기 - 2만보다 훨씬 높은 소리.

pulse train :

그래프는 다 더했을 때 나오는 모양.

****frequency, sine wave, sr의 개념을 다시 보기 + fre 랑 sine wave의 관계.**

part 13

fre를 440hz로.

시간. 0.5초까지 만들었는데 0.01초까지 display.

parameter setting을 바꿔서 440hz(라 소리)를 880hz로 바꾸면 아까 소리와 높이가 같음.

- 1760hz로 하면 . 다 '라' 가 됨. -220,110 도 라. 배수로 하면 다 같은 음으로.

sin 대신 cos 을 쓰면 시작점이 달라짐. - 1부터 . 1초간 몇 번 왔다갔다 하는지는 같음

fre는 고정이기 때문에. / 두 소리는 둘다 '라' . 다르게 들리지 않음. / sin cos는 shape은

같지만 sin는 살짝 이동한거 $-\pi/2$ 90도 차이가 있음. / cos에서 90도 옆으로 이동하면 sin.

$\pi/8$ 를 이동하면 소리가 달라질까 ? 노. 얼마큼 이동하든 소리는 다 같음.

phase 에 대한 거는 인식을 못한다. frequency 변화는 느끼지만 phase shift는 아님.

complex phasor 실행 -

complex number자체는 플래팅이 안됨. a ,b를 각각해서 2차원으로 플래팅.

>라 에 해당하는 소리가 나옴.

real = cos

Pulse train

F0를 하나 정하고. 오천까지(NF)

spectrum - 한 타임 포인트에서의 어떤 주파수 성분이 많은지 보여주는거. 낮은 주파수부터 있는 그림.

decreasing 하게 만들어야.

pulse train 만든걸 prat.

<11/19>

데이터 기계(인공지능,함수) 데이터

함수

데이터 부분은 벡터-숫자열로 되었음. 숫자열이 앞뒤 같을 필요는 없음.

음성 text

>음성이 들어가서 text 형태로 나오는 >>음성인식

text 음성 - 음성합성

일본어text 한국어text - 기계 번역

-알파고 같은 경우 바둑상태가 입력으로 들어가서 몇 번째 수를 뽑아야 하는가가 나옴.

중간에 있는게 인공지능, 함수/어떤 형태를 지닐까 -행렬의 형태

입력벡터를 출력벡터로 만들어주는 함수의 역할. -인공지능

음성 텍스트 사람이름,, 뭐든 될수있음 벡터는

모든 데이터가 벡터로 되어야하는건 행렬 곱을 하기 위해서.

행렬의 형태를 거치면 다른형태가 될수있음- 이미지가 들어가면 누구의 얼굴

5 3 0 1 -1 0 2 -3 6

0 1 3

3 -5 7

2 3 4

4 x 1(세로가 몇 개있냐와 관련) 4x 3 가로가 몇 개 있냐와 관련.

젤 왼쪽 컬럼이 왼쪽에 들어가 곱하고 더해서 오른쪽벡터가 됨.

중간계 행렬함수. 왼쪽이 입력(음성) 이라 생각하고 함수를 곱하기해서 통과하고 , 행렬과 행렬의 곱. 입력이 함수를 통해서 출력이 됨.

기계가 많은 데이터로부터 학습을 해서 얻어내는 것이라고 생각 - 인공지능.

인공지능- 행렬의 곱. 입력벡터를

Linear algebra 선형대수 - 행렬

모든 인공지능이 선형대수.

<11/21>

A x = b

1 3

5 1 6 2 0 15 29 3

6 -1 3 9 1 33 19 1

33 3 -1

입력

출력

3 by 2. 2 by 3__ 3 by 3

-바깥에 있는게 결과값.

A =

[1 2

-1 0

3 5]

3 by 2

col - ws = 3차원

spanning 시키면 2차원의 plane이 나옴. = column space - cv를 표현하고 원점과 연결시키면

cs(2차원) 와 ws(3차원) 는 안똑같아. - 한차원 남는게 left null space

LC으로 표현가능한 모든걸 spanning . 벡터 두 개를 LC 하면 B가 나오는데 B가 될 수 있는 거, all possible Bs 가 column space. -

row space

ws 는 2차원. spanning해내는 공간은 2차원을 넘어가지 않음.

rv 가

column I 2 개 - cs와 같음. = rank

column 으로 하든 row로 하든 I 한 벡터들의 숫자는 늘 같음 - rank

3×2 (ws) df
 $2 = 2 = \text{cs}$. I 한것의 개수
 $1 \quad 0 \quad > \text{null space}$

$x_A = 0 \quad [0 \ 0 \ 0 \ 0]$

A라는 matrix 가 있는데 뭐를 곱해도 0이 되는 것.
x에는 1by 3가 와야됨. (3차원)