Phonology는사람이 인식하는 차원이고 Phonetics는 사람이 인식하지 못하는 하위의 차원이다. 발화 시의 물리적인 현상을 그 자체로 바라보는 것이 Phonetics이다.

Vocal tract에는 Nasal과 Oral tract가 있다. Nasal tract로 가면 nasal sound가 되고 oral tract로 가면 oral sound가 된다. Larynx는 후두이고, larynx 사이의 틈을 glottis라고 한다. 사람이 꿀꺽할 때 기도를 막는 역할을 하는게 epiglottis이다. Nasal/Oral은 Velum에 의해 결정되는데, Velum이 raised되면 oral이고 lowered되면 nasal이다. 코로 숨 쉴 때 velum은 lowered된다. Voicing을 결정하는 것은 larynx에서 이루어진다. Glottis가 열려 있으면 성대가 떨리지 않아 voiceless sound가 나오는 반면에 glottis가 닫혀 있으면 성대가 떨려 voiced sound가 나온다.

영어에서 소리는 5가지 요소로 인해 결정이 되는데, Constrictor, Constriction Location(CL), Constriction Degree(CD), Velum, Larynx가 그것이다. Lips, tongue tip, tongue body는 articulatory process를 담당하는 constrictor이다. 이곳에서 혀와 입술을 이용해 소리에 영향을 미친다. Velum은 oro-nasal process를 담당한다. 이곳에서는 소리의 oral/nasal을 결정한다. Larynx에서는 phonation process가 발생하는데, voicing이 이곳에서 결정된다.

Constrictor이 정해졌다면 Constriction location, Constriction degree로 더욱 specify 해야 한다. Lips가 constrictor이라면 bilabial, labiodental이라는 CL로 나뉜다. Tongue tip이 constrictor이라면 dental, alveolar, palato-alveolar, retroflex로 CL이 나뉜다. Tongue body가 constrictor이라면 palatal, velar로 CL이 나뉜다. 'J(y)'는 palatal에 속하고 'sh'는 palate-alveolar에 속하고 'r'은 retroflex에 속한다.

Constriction Degree는 stops, fricatives, approximants, vowels로 나뉜다. Approximants에는 'r', 'l', 'w', 'j' 4개만 있다. 'm', 'n', 'ng'도 nasal sound이지만 stop이다. 모든 모음의 constrictor은 tongue body이다. 모든 phoneme은 이러한 요소들의 combination으로 설명이 된다.

Spectrogram에서 띠로 나타나는 것을 formant라고 한다. 각 띠마다 아래에서부터 first formant, second formant 등으로 부르며 f1, f2 등으로 줄여서 부른다. 아래에서부터 위로 formant는 무한대로 나타난다. Formant값이 모음을 결정한다. 같은 모음이면 비슷한 formant 형태로 나타난다. 즉 formant는 모음을 구별하는 수치적인 지표이다.

Pitch setting- pitch range에서 설정을 해줘야 남자 목소리, 여자 목소리 구별이 가능해진다.

소리를 녹음하고 확대하면 하나의 큰 파도, 패턴이 보이는데, 이 큰 파도 하나에 성대의 떨림 한 번이 대응된다. 큰 파도가 1초 동안 나타나는 횟수가 바로 pitch(Hz)이다. 이 큰 파도의 duration을 재고 이 값을 통해 (1 나누기 duration값)을 하면 큰 파도가 1초 동안 나타나는 횟수를 계산할 수 있으며 이것이 pitch값이다. Pure tone으로 내 목소리의 pitch를 입력하면, 내 목소리와 같은 높이의 소리를 들을 수 있다. 이때, pure tone의 곡선은 사인 곡선과 같다.

이 세상 모든 소리는 사인 웨이브의 조합으로 표현된다. 사인 웨이브는 frequency와 amplitude로 결정된다. 사인 웨이브의 x축은 시간이며 y축은 단순한 값(value)이다. 우리 주위의 모든 소리는 pure tone이 아닌 복잡한 소리라고 할 수 있다. 이 때 이 복잡한 소리의 spectral analysis에서 나오는 첫번째 사인 웨이브를 f0, fundamental frequency, the number of vocal cords vibration of a second 등으로 부른다. F0의 frequency가 전체 소리의 pitch이다.

Source는 성대에서 나는 소리를 그대로 녹음한 소리이다. 성대에서 소리를 바로 녹음하는 것을 EGG라고 한다. Source의 소리와 기타의 소리가 유사하다. Source의 spectrum은 pulse train의 형태를 띈다. 여기서 Pulse는 waveform에서 툭 튀어나오는 부분들을 가리킨다. Pulse train의 spectral analysis를 해보면 gradually decreasing하는 형태를 띈다. Source는 harmonics를 이룬다. Harmonics-> F0의 배음들(f0의 frequency의 배수들)이 이루는 사인 웨이브로 소리가 표현된다. 모든 소리가 harmonics를 이루는 것은 아니다. Source와 기타소리 등이 harmonics를 이루며 gradually decreasing을 한다.

Source는 vocal tract를 거치며 filtered된다. Source가 vocal tract를 지나 articulate되면 peak/mountain과 valley가 있는 산맥 형태의 spectrum이 만들어진다. 즉, 산맥을 만들어 주는 것이 filter의 역할이다. 각 모음을 발음할 때마다의 입, 혀 모양이 다른데, 각 모음마다 좋아하는 산맥형태가 있다고 생각하면 된다. Source가 vocal tract를 지날 때, 그 입모양이 도장 역할을 해서 source의 spectrum을 도장 찍는다고 생각하면 된다. 그렇게 해서 도장 찍힌 spectrum이 실제 발화된 소리의 spectrum이 된다. 이 때 이 spectrum은 산맥을 형성하게 된다. 그 때, 첫 mountain 부분이 첫 번째 formant로 f1이며 다음 formant가 f2이고, f1과 f2만 있으면 모음이 결정된다. F1을 y축, f2를 x축으로 했을 때 입 모양과 똑같이 된다. 이 때 f1은 혀의 높낮이를 결정하고, f2는 혀의 front, back를 결정한다. F2가 클수록 front vowel, f1이 작을수록 high vowel.

Spectrogram은 spectrum을 시간축으로 늘어놓은 것이다. Spectrum의 입체형이 spectrogram이라고 생각하면 된다. Spectrogram의 x축은 시간이고 y축은 frequency이고 짙은 정도는 amplitude이다. Spectrum들을 연달아 연속적으로 연결한 것이 spectrogram이다.

Spectral slice를 통해 spectral analysis를 정확히 보기 위해서는 최소 여러 개의 큰 파도(큰 파도 한 번에 성대 한 번 진동)를 선택해야 spectral analysis를 제대로 알 수 있다.

Stereo는 병렬적으로 일일이 다 더해서 나오는 소리이다. Pure tone이 남아있는 소리이다. Mono는 pure tone이 남지 않고 완전히 합쳐지는 소리이다. 그래서 단 하나의 wave form으로 표현되며, spectral analysis를 해보면 mono 소리를 구성하는 각 사인웨이브를 알 수 있다. Mono로 convert를 하면 mono 소리가 그 f0 소리와 pitch가 동일하게 된다. Stereo는 그렇지 않다. Gradually decreasing 하고 harmonics를 이루는 pure tone을 여럿 mono로 만들면 source 소리가 나와 pulse가 만들어지는데, pure tone이 많을수록 pulse가 더욱 두드러지게 튀어나온다.

코딩: 자동화하는 것
똑같은 것이 반복이 되기 때문에 자동화를 한다.
Ex) 폰에 있는 모든 것들은 코딩으로 이루어짐.

사람 language처럼 컴퓨터 language가 있다.
모든 language의 공통점-> 단어, 문법. 단어가 있고, 단어를 어떻게 combine 하느냐 하는 것. 단어는 그 속에 의미(정보) 포함. 단어: 정보를 담는 그릇.
컴퓨터 language에서 단어에 해당하는 것이 변수(variable). 변수에 정보를 담고 기계에 전달을 할 때에 그 방식이 필요함. 이때 필요한 것이 문법. 이때 문법은 다음과 같다. 첫번째, 변수에 정보를 넣는 것 variable assignment. 두번째, if문법을 씀. ~할 때는 ~하라 이런 것 if conditioning. 세번째, 여러 번 반복하는 것. For문법 사용해 여러 번 반복하게 함 for loop. 네번째, 함수를 배우는 것. 내가 무엇을 입력하면 어떤 출력이 나오는 것. 입력과 출력의 packaging을 하는 것이 함수.
정보의 종류에는 숫자, 글자. 두 가지만 존재.
'=' sign 뜻: 오른쪽에 있는 정보를 왼쪽에 있는 variable로 assign한다.
a = 1 하면 1이라는 정보를 a라는 variable에 넣는 것임.
print라는 것도 함수임. 어떤 변수를 그 안에 넣으면 변수에 assign된 것이 뭔지를 알려주는 함수임.
여기서 a가 입력이고 1이 출력.
a = 1 한 후 a = 2라고 또 하면 overwrite됨. 덮어씌워짐.
그냥 알파벳을 쓰면 무조건 변수로 취급함. 그러니 문자를 쓰고 싶으면 ''를 써야함.
Run 단축키는 시프트+엔터.
하나의 셀에서 마지막 줄에 변수를 하나만 쓰면 print 안 써도 그 변수에 assign된 것 보여줌.
여러 개를 한 변수에 담을려면 list 써야함. 이를 대괄호로 표현할 수 있음. a = [1, 2, 3] 이렇게 함. a = [1, 2, 3, 5, 'love']라고 해도 됨. List 안에 list가 들어갈 수 있음. a = [1, 'love', [1, 'bye']] 이래도 됨.
이때 대괄호 말고 소괄호 쓰면 type이 tuple이라고 나온다. Tuple이 보안에 더 강함. 사실상 list와 tuple은 같다고 보면 됨.
이것이 어떤 종류의 변수인지를 알기 위한 함수는 type임. type로는 list, int(integer), float, str(string) 등이 있음. Float는 실수(?)임. String은 문자배열.
a = {'a': 'apple', 'b': 'banana'} 이건 dict(dictionary)임. 이 때 'a'가 표제어, 'apple'이 설명어임. 'b'가 표제어, 'banana'는 설명어. Dictionary의 괄호는 중괄호.

a라는 리스트에서 부분만을 가져오고 싶다면 a[0]의 방식으로 가져올 수 있음. . a list에서 0번째, b에서 0번째 꺼를 가져와서 더한 것임. list에서 첫번째는 0번째임. 정보를 통째로 가져오려면 a를 적고, 그 중에서 선별해서 가져오려면 a[몇] 쓰면됨.
float라는 함수는 어떤 variable이 들어오면 그걸 float로 바꿔줌. int함수는 variable을 int로 바꿔줌. 정보에서 어떤 것을 가져올 때는 반드시 대괄호를 적고 그 안에 index를 적으면 내부적인 정보를 부분적으로 가져온다.
''로 되어있으면 문자 취급, 그게 아니라 그냥 숫자만 되어있으면 숫자 취급. '1'이면 문자, 1이면 숫자. Dict에 있는 정보를 access 할 때는 페어에서 앞부분을 index로 쓴다. 0,1,2 그런 숫자로 하지 않는다. String과 list는 정보를 접근하는 방식이 매우 비슷함. Parallel함.
s[1:3]이렇게 나온 것은 range로 정보를 가져오는 것인데, 첫번째부터 3번째 직전까지 정보를 가져오는 것임(3번째는 가져오지 않음.)
len함수는 정보의 개수, 길이를 알려줌(length).
더하기는 숫자만이 아니라 문자에도 똑같이 적용됨.
Upper은 업 시켜주는 함수임. 소문자는 대문자로 만들어줌.
find함수는 첫번째 나오는 것을 찾아주는 함수. 전부다 찾아주지는 않음. 띄어쓰기도 순서에 고려됨.
rindex함수는 가장 나중에 나온 것을 찾아주는 함수.
strip함수는 남은 것들 걸러서 온전한 텍스트만 남기는 함수.
s.split(' ') 뜻: s라는 string을 ' '(빈칸)을 이용해서 잘라라. 중요!
여러 개로 쪼개진 것을 다시 합치고 싶을 때 join을 씀
' '.join(tokens) 뜻: 빈칸을 이용해서 token에 들어 있는 list를 붙여라
s.replace('this', 'that') 뜻: 이 스트링에 있는 모든 'this'를 'that'으로 바꿔라.
For loop- 여러 번 반복 될 때 쓰임.
for i in a: in 뒤에 있는 것(a)을 하나하나씩 돌려서 i가 그 하나하나씩을 받아서 뭔가를 하라. a에 있는 것들을 하나하씩 돌려서 i에 할당을 하고 그 아래에 있는 것을 하라.
Range 함수: range 뒤에 어떤 숫자가 나오면 list를 만들어준다. 4라고 나오면 0부터 3까지의 list를 만든다.
Enumerate 함수: 번호를 매긴다. 그 list 값도 나오지만 추가로 번호를 매겨진다.
"{}: {}%".format(s, b[i]*100) 뜻: format괄호 안에 있는 것들이 왼쪽의 형태로 박혀라. 즉, 왼쪽의 형태로 format으로 하고 싶으면 쓰는 것.
Zip은 따로 독립된 함수를 합치는 것. 페어로 합침.
==하면 진짜 같다는 의미. =가 어떤 값을 변수에 assign하는 기능 이라면 ==는 진짜 같다는 equal sign임.
>= 이건 부등호. 순서 중요. a >= 0 이면 a가 0보다 크거나 같다는 뜻.
!= 하면 같지 않다는 의미(==의 반대)
For, if 함수에서는 끝에 클론(:)을 붙이고, 인덴트를 하는 것이 매우 중요.

데이터를 숫자로 변환할 때 숫자의 열로 표현한다. 이 숫자의 열이 벡터. 사진들도 모두 숫자로 표현. 모든 데이터는 벡터의 형태로 되어야 한다. 데이터는 벡터의 형태로 했을 때 다루기가 쉽다.
행렬- 직사각형의 형태에 숫자를 넣어 배열한 것. 가로가 행, 세로가 열이다. 행렬은 벡터는 아님. 길게 늘여놓아 진 것이 벡터.
컬러 사진은 행렬을 세겹 겹쳐 놓은 것(RGB). 흑백 사진은 행렬 하나. 동영상은 시간에 따라 계속 행렬이 변하는 것. 흑백 사진은 2차원, 컬러는 3차원, 동영상은 4차원
소리도 벡터화 시킬 수 있다. 소리의 웨이브는 결국 점으로 이루어져 있는 것인데, 이를 숫자로 처리하는 것이다. 텍스트도 벡터화 시킬 수 있다.
중요한 함수는 import를 해서 그 라이브러리를 불러와야함. 여기서 라이브러리는 일종의 함수 패키지라고 할 수 있다. Numpy라는 패키지 안에 여러 함수가 포함되어 있고, Numpy안에 또 다른 함수패키지도 포함되어 있다. Numpy.A.D.f 이렇게 점(.)을 통해 아래 패키지로 접근해 나가는 것. 또는 from Numpy import A.D 를 하면 Numpy안에 있는 A, 그 안의 D를 불러올 수 있게 됨.
Numpy는 일반적인 list에서는 수학적인 처리가 안되기 때문에 수학적인 처리를 해줘야 할 때 쓰인다. Numpy가 적용되면 list끼리의 계산이 가능해진다.
Numpy를 이용해 벡터(1차원) 행렬(2차원) 등을 만들어 계산을 할 수 있다. Matplotlib.pyplot라는 라이브러리는 plotting을 할 때 쓰인다. 히스토그램을 만들 때 이 plt가 적용된다.
Numpy에서 중요한 것은 shape, ndim, dtype 등이다. Shape와 ndim은 깊이 연관되어 있다.
Csv 파일은 데이터를 주고받을 때 쓰이는 파일 형식이다. 컴마를 이용해서 구분하여 두개씩 짝지어진 데이터를 담는 파일 형식이다.
Numpy에서 데이터 타입은 float64 등과 같이 뒤에 숫자가 표현되는데, 여기서 숫자는 소수점 몇째 자리까지 표현할 수 있는지, 그래서 얼마나의 precision을 둘 수 있는지의 값이다. 이 값이 크면 더욱 정확하고 세밀한 값을 이용할 수 있지만, 용량이 커진다는 문제점이 있다.
Reshape을 할 때 중요한 것은 같은 size로만 reshape을 할 수 있다는 것이다. 예를 들어 (2, 3, 4)의 shape을 가진 array가 있을 때, 이것을 (1, 7, 3)등과 같이 size가 다른 형태로는 reshape할 수 없고, (4, 3, 2)와 같이 같은 shape일 때에만 reshape 할 수 있다. 또한 주목할 점은, (-1, 3, 2)라는 형태를 써서 reshape을 할 수 있는데, 이는 3과 2가 정해지면 남는 값은 4로 자동으로 정해지기 때문에, 남는 값은 -1로만 써도 제대로 적용이 되어 reshape된다.
A라는 variable이 numpy의 형태이라면, a.sum()을 하면 자동적으로 모든 값들이 더해진다. Np.sum(a)를 써도 같은 값이 나온다. 이때, a.sum(axis=0) 등과 같이 차원을 적용해서 sum을 해야할 때는 조금 복잡해지는데, axis가 0이면 첫번째 차원의 관점에서, 1이면 두번째 차원의 관점에서 sum을 각각 해야한다. 이는 잘못하면 헷갈릴 수 있으니 충분한 연습이 필요하다.
행렬 사이에서 계산을 할 때에는 두 행렬이 반드시 같은 shape일 필요는 없다. 같은 shape이 아니라도 broadcasting을 통해 계산을 할 수 있다.