

*Phonetics는 물리적으로 어떻게 소리가 나는지와 관련된 것이다. 소리가 어떻게 인지되는지에 관심을 두는 phonology와는 차이가 있다. 영어에는 44개의 다른 소리가 존재한다.

Articulatory phonetics(mouth), Acoustic Phonetics(through air), Auditory Phonetics(to ear)

*Articulation(조음)

vocal tract(nose, pharynx, larynx, ear로 구성)

upper vocal tract[lip, teeth, alveolar ridge, hard palate, softpalate(velum), uvula, pharynx wall로 구성]

lower vocal tract[lip, tip, blade, front/center/back tongue, root, epiglottis로 구성]

한국어는 음절이 반복되고, 영어에서는 stress가 반복되기 때문에 혀의 위치가 중요

폐 -> 성대(수문과 비슷한 역할); 떨림이 느껴진다는 것은 성대가 떨력인다는 것

여자 250번 정도, 남자는 100번 정도

1) articulatory process [lips, tongue body, tongue tip]

constriction location : where exactly -(앞뒤)

constriction degree how much exactly(상하)

2) phonation process[larynx]

voiced(진동을 느낄 수 있음) - 모든 모음과 일부 유성 자음

voiceless(진동x) - p, h, k, s, f

3) oro-nasal process[velum]

velum이 내려가면서 소리가 지나가는 통로가 열림

모든 모음과 비음(m,n,ng)을 제외한 자음도 velum이 열리면서 소리남

*constriction location : where exactly -(앞뒤)

lips(2개의 소리: 앞 - [ㅂ][ㅍ] / 뒤 - [ㄴ][ㄹ])

tongue body(2개의 소리 - 앞 : [ʃ],[y] [ear], [yarn] / 뒤 : 그)

tongue tip(4개의 소리 - [th(윗니)],..)

constrictor 관점에서는 같지만, 로케이션 관점에서는 다른 소리

*constriction degree how much exactly(상하): 조금 막을 건지, 많이 막을 건지, upper part를 치는 정도에 따라

stops: [p], [b], [t], [d], [k], [g]

fricatives 마찰음 [f], [v], [θ], [ð], [s], [z], [ʃ], [ʒ], [h]

approximants [r], [l], [w], [j]

vowels 모음의 정의는 막힘이 없는 것 -> upper part를 별로 안 칩

tongue body, tongue tip, lip, velum, larynx의 위치로 어떤 소리가 나는지 알 수 있다.

*praat: 소리의 duration, intensity(loudness), pitch를 확인 가능

빨간 띠: formant(F1, F2에 따라 모음 구별 가능)

praat에 나타나는 큰 파동의 횟수는 larynx의 떨림 횟수와 일치

헤르츠: 1초에 성대가 떨리는 횟수

2주차

Pratt

intensity(db), pitch(hz), Formants(hz) 분석 가능

HZ(= frequency의 단위) : 1초 당 반복되는 구간이 몇 번 생겼는지

여기서, 반복되는 구간은 결국 sine wave이며, Larynx의 떨림과 일치한다.

그리고 이 세상의 모든 signal은 sine wave로 이루어졌다.

사람의 목소리도 마찬가지다. 목에다 장치를 설치해, 그 소리를 녹음하면, sine wave가 나타

나지만, 마이크를 통해 사람의 말소리를 녹음하면 sine wave의 합(=complex sound)이 나오

즉 입모양에 따라 합쳐지는 sine wave의 종류가 달라진다.

sine wave는 frequency와 magnitude 두 가지 특성에 의해 구분된다.

합쳐진 sine wave는 complex tone이라 하는데, F0과 반복되는 주기가 일치한다. 그리고 각 sinewave들의 frequency는 f0의 배수이다.

voice source(목에서 직접 측정한 소리): 스펙트럼이 점진적으로 감소하는 모양

filtered by voice tract(마이크로 측정한 소리): 점진적으로 감소하는 모양이 유지되지 않음

하지만, 둘 다 f0 배수의 frequency를 가진다는 공통점이 있다.

소리를 표현하는 방법

1)스펙트로그램

x축: 시간, y축: frequency

*위쪽으로 갈수록 열어짐

*frequency와 amplitude는 반비례

2)스펙트럼

x축: frequency, y축: amplitude

*주파수 별로 어떤 성분이 있는지 확인 가능하다.

목소리가 정해지는 방법 = f0과 이에 대응되는 합

f0은 pitch와 일치

filter: 스펙트럼에 나타나는 mountain과 valley의 모양을 결정지음

입모양에 따라 비슷한 모양을 띠

f1은 formant와 일치

f1은 혀의 높낮이와 일치

f2는 혀의 front / back과 일치

따라서 일정한 주파수와 amplitude 수치에 따라 특정 모음과 자음의 소리를 구성할 수 있고 pure 톤들의 합으로 사람 목소리를 나타내는 것이 가능하다.

3주차

코딩이란?

자동화 - 계속 같은 것이 반복되기 때문에

언어 - 단어를 어떻게 결합하는지(단어와 문법)

단어 - 의미/정보를 담는 그릇

어떤 단어를 이용해서 어떻게 합칠 것인가

컴퓨터 언어에서 단어의 역할을 하는 것은 변수(variable), 변수에 담기는 것은 문자와 숫자가 전부

1. Variable assignment 변수에다 정보를 지정하는 것
2. if Conditioning 자동화, 기계화 - 조건문 문법 if
3. 자동화에 중요한 것은 여러 번 반복하는 것 for
4. 함수: 어떤 것을 입력했을 때 원하는 값이 나오는 것이 함수

1,2,3,4가 함수 안에 들어가는 것

Ex> 모든 사람이 함수다 : 먹고 싸는 거, 기름으로 자동차가 달리는 것

재사용과 반복을 위해 코딩을 한다.

= : 같다는 뜻이 아니라, 오른쪽에 있는 정보를 왼쪽에 있는 변수로 assign한다.

함수(입력)

각각의 셀이 따로 실행 가능

Variable은 정보를 override 함

한 변수에 여러 개를 정보를 넣고 싶을 때 - list[], tuple(), dictionary={'key':'value'} 표제어와 설명의 쌍

차이 - 튜플이 더 보안에 강하다(바꾸기 힘들다)

Type() : 변수가 어떤 특성을 가지고 있는지 알려줌

4주차

하나의 variable에 여러 정보를 담으면, 최신 정보로 업데이트 됨

어떤 variable 속에 정보가 들어 있을 때,

retrieve 정보를 가지고 온다.

변수 간의 덧셈, data access

a=[0번째, 1번째,...]

리스트에서 일부분의 정보를 가져올 때, 대괄호를 이용함

타입종류()어떤 변수의 type을 다른 타입으로 바꿔주는 함수

어떤 변수의 내부 정보를 가져오고 싶으면, 대괄호를 사용하여, 인덱스를 적으면 됨

dict의 access

pair set에서의 key 부분을 인덱스로 사용

string과 list는 정보의 접근 차원에서 비슷

```
In [47]: s='abcdef'
```

```
In [48]: print(s[0],s[5],s[-1],s[-6])
```

a f f a

왼쪽으로 가면 음수

print(s[1:3]) - 1에서부터 3까지 가져와라

print(s[1:]) - 첫 번째부터 끝까지

print(s[:3]) - 처음부터 2번째까지

리스트도 같은 방식

len() 함수 - 정보의 길이를 알려줌

s.upper() - 대문자로 변화

for i in a:

in 뒤에 있는 것을 하나씩 돌려서 I가 그 하나하나를 받고 어떤 일을 해라

if conditionint

function 입력이 들어가고 출력이 나오는 것들(이미 있는 것도 있고, 내가 만드는 것도 있음)

ipynb 확장자: 코드와 필기를 한번에

```
In [18]: a=['red', 'green', 'blue', 'purple']
b=[0.2, 0.3, 0.1, 0.4]

# enumerate도 함수명 - 번호 매기는 함수, 두개의 변수가 들어갈 수 있음
for i,s in enumerate(a):
    #i에다가는 번호(index 값), s에다가는 red 이런 애들
    print("{}:{}".format(s, b[i]*100))

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

```
In [19]: a=['red', 'green', 'blue', 'purple']
b=[0.2, 0.3, 0.1, 0.4]

# enumerate도 함수명 - 번호 매기는 함수, 두개의 변수가 들어갈 수 있음
for i,s in enumerate(b):
    #i에다가는 번호(index 값), s에다가는 red 이런 애들
    print("{}:{}".format(a[i], s*100))

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

```
In [21]: a=['red', 'green', 'blue', 'purple']
b=[0.2, 0.3, 0.1, 0.4]

# enumerate도 함수명 - 번호 매기는 함수, 두개의 변수가 들어갈 수 있음
for s,i in zip(a, b):
    #a와 b는 반드시 길이가 같은 리스트가 되어야 함, zip - 둘을 합친다는 뜻, 덩어리처럼 바뀌는 것
    #s에는 a의 값, i에는 b의 값이 들어가는 것, index 값은 들어가지 않음
    print("{}:{}".format(s, i*100))

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

```
In [28]: a = 3
#(크거나 같으면)
if a >= 0:
    print('yy')
else:
    print('nn')
```

yy

```
In [31]: for i in range(1,3):
        for j in range(3,5):
            if j >=4:
                print(i+j)
            #!!!인덴테이션!!!
            #계층적인 개념을 잘 알기!!
```

```
File "<tokenize>", line 4
    print(i+j)
    ^
```