

Phonetics – 물리적 소리 그 자체

Phonology – 물리적 소리를 머리로 인지하는 과정 (sound system)

Phonetics

articulatory : 소리를 만드는 원리에 관한 것 (from mouth)

acoustic : 소리가 공기를 타고 가는 것 (through air)

auditory : 소리를 귀로 듣는 것 (to ear)

articulatory

1. phonation process

Larynx open -> vocal cord vibration (x) -> voiceless

Larynx closed -> vocal cord vibration (o) -> voiced

2. oro-nasal process

Velum lowered (nasal tract open) -> 비음 (m, n, ŋ), 코로 숨쉴 때

Velum raised (nasal tract closed) -> 모든 모음, 비 음을 뺀 모든 자음

3. articulatory process

3-1. constrictor

(※ 모든 모음의 constrictor -> tongue body)

lip / tongue tip / tongue body

3-2. constrictor location(CL – 앞뒤)

Lip – bilabial / labiodental

Tongue tip – dental / alveolar / retroflex / palato- alveolar

Tongue body – palatal / velar

3-3. constrictor degree(CD – 상하, 공기흐름이 막히 는 정도)

Stop / fricative / approximant / vowel

Acoustic

Pitch – 소리의 높낮이

Intensity – 소리의 크기(볼륨 / dB)

Formant – 가로줄 4줄(점찍힌 부분) -> 모음을 구별 하는 지표

Phonemes: Individual sounds that form words Lips: p,b,m,f,v,w Tongue tip: θ ,ð, t,d,s,z, ʃ,ʒ ,l,r

Tongue body: k,g,ŋ,j,vowels Velum: m,n,ŋ Larynx: p,f, θ , t,s, ʃ ,k,h

Source -Sound at larynx -Complex tone, not a simplex tone

Sine wave -Frequency, Amplitude에 의해 결정 -가장 기본적인 형태 -이 세상 모든 것들: sine wave
들의 결합 -x축은 시간, y축은 value

Spectrum -Sine wave를 frequency, amplitude 그래프로 변환한 것 -x축:frequency, y축:amplitude

Spectrogram -에너지 분포를 보여줌 -x축:시간, y축:frequency

Complex tone -주기가 제일 작은 single tone 주기와 같음 -> 즉 가장 slow한 single tone의 frequency가 complex tone의 frequency

Human voice source -spectrum을 보면 gradually decreasing -F0(fundamental frequency)=pitch - amplitude 점점 줄어듦 주파수가 배음의 harmonix구조. Cf)frequency를 10000까지 자른다고 가정했을 때 남자의 배음 수가 더 많음.

Filtered by VT -harmonix 구조는 유지 그러나 gradually decreasing이 아님

Guitar Plucking -Voice source와 비슷. -F0 존재, harmonix 구조

Synthesizing source -여러 개가 동시에 존재: stereo -> simplex tone들 여러 개 존재 -하나만:mono -> complex tone -cf)pure tone들을 무한대로 합치면 pulse train 구조

Source filter theory -Wave form에서 보면 voice source는 pulse train 형태 -spectrum을 보면 harmonix의 형태를 이루며 gradually decreased. -이가 vt를 통과하면 나오는 소리의 스펙트럼을 보면 산맥(peak)이 존재. 스펙트럼에서 가운데 거를 도장이라고 생각하면 이게 우리의 filter인데 이걸 source에다가 찍으면 마지막 그림이 나온다. F0는 제일 첫번째 나오는 것의 frequency고 formant는 첫 번째 peak의 frequency?

Vowel space -F2를 x축 f1을 y축으로 점을 찍어보면 우리들의 입의 위치와 같다. F1은 그 모음의 높낮이(height)를 결정 f2는 앞뒤(frontness/backness)결정.

코딩: 자동화

자동화를 하는 이유: 똑 같은 것이 계속 반복되기 때문에.

프로그래밍 언어

-Java, Python, C 등이 존재.

-이들은 모두 문법, 단어가 존재한다는 공통점.

변수(Variable)

-단어란? 정보를 담는 그릇.

-컴퓨터 언어에서 단어에 해당하는 것이 변수.

-이 변수에 정보를 담고 기계와 커뮤니케이션할 때 필요한 것이 문법.

문법

1. 변수에 정보를 넣는 것이 **Variable assignment**

2. **If conditioning.** 자동화, 기계화에 있어서 조건화에 대한 문법이 필요함.

3. **For/Loop**를 통한 반복

4. 함수를 배우는 것. 어떤 입력값을 넣었을 때 내가 원하는 출력값이 나오는 것이 함수. 반복적, 재사용 가능.

Python

-정보의 종류: 숫자 / 글자

-컴퓨터 언어에서 =는 같다가 아니라 오른쪽의 정보를 왼쪽의 variable에 assign한다는 뜻. 즉, 오른쪽의 것이 정보이고 왼쪽의 것이 variable이어야 함. 순서 바뀌면 안됨!!

-문자를 assign 할 때는 반드시 quote를 해주어야 함.

But, 문자가 변수일 때는 quote를 안 해도 됨.

Ex) love=3; a=love; print(a) 하면 3이 나옴.

함수

1. Print()

-어떤 변수를 그 안에 넣으면 그 안에 있는 것을 screen에 print out 해주는 함수.

-ex) a=1 ; print(a) 를 했을 시 1이 출력됨.

Cf) print(a) 했을 때 나오는 1은 결과값이지 별개의 cell이 아님. 즉 뒤에 a=2를 run 하고 다시 print를 하면 a=2가 나옴.

Cf) print를 안 해도 한 셀 젤 밑에 변수명만 쳐도 제일 밑의 변수값이 나옴.

2. Type()

-변수의 유형이 무엇인지 보여주는 함수

-ex) a=[1,2,3,5] 이면 type(a)의 결과는 list

a=(1,2,3,5) 이면 type(a)의 결과는 tuple.

➔ 여기서 list와 tuple은 이름만 다르지 실질적 기능은 똑같음.

Cf) List 속에 List 가능할까? 가능함.

3. Dictionary{}

-표제어, 표제어에 대한 설명으로 구성됨.

-,에 따라서 구분.

-{}를 사용

-정보를 pair로 넣는다. pair는 colon을 통해 setting이 된다.

앞 부분을 정보의 index로 사용한다.

Retrieve

-어떤 variable 속에 정보가 들어 있을 때, 그 정보를 가지고 오는 것.

-variable 내부 정보에 들어갈 때는 반드시 []를 쓰고, []안에 들어가는 것은 index를 사용함.

List에서 정보에 access하는 법.

-[]를 통해서 access.

ex) print(a[-1],a[2])

-range를 정해서 access하려면? : 를 사용

ex) a[1:3] -> 1번째, 2번째

a[0:4] -> 0번째~3번째

a[:] -> 전체

-cf)list와 str은 정보를 가져올 때 같은 방식으로 함.

-cf)str에서 quote 빼고 정보를 빼내려하면 error뜸 ->정보가 하나이기 때문.

ex) a=123; print(a[1]) 하면 error임.

Dict에서 정보 access

ex) a={"a":"apple", "b":"banana", "c":2014}

에서 dict는 pair 중 앞의 것을 index로 사용하기 때문에

-print(a["a"]) 와 같은 식으로 정보를 쓸 수 있음. 이 때, index에 quote가 들어가 있기 때문에 정보에 access할 때도 quote를 사용해주어야 함.

-물론, 여기서 표제어로서 str이 아닌 int를 사용할 수도 있다. 즉, "a"를 1로 대체하고

print(a[1])을 해도 똑같이 apple이 나온다는 것이다.

덧셈 및 곱셈

-예를 들어, a='피자가좋아' 라고 하자.

-a[0]+a[1]*3을 하면 결과로서 '피자자자'가 나온다.

cf) print(a[0])+print(a[2]) 같은 식은 error이 뜬다.

이를 활용하기 위해선 print(a[0]+a[2])와 같은 식으로 사용해야 한다.

함수

1) float

-어떤 variable이 들어오면 그것을 float으로 만들어 줌.

2) int

-어떤 variable이 들어오면 반올림을 하던가 해서 그것을 int로 만들어 줌.

3) Len

-variable 내의 정보의 길이를 준다.(정보의 개수)

-ex) a=['1','2','3']; len(a) 하면 3이 나옴.

b='1234' ; len(b) 하면 4 나옴.

-주의할 점은 Len함수는 int와 같은 숫자에는 사용할 수가 없음.

-ex)a=[1,'2',[2,3,4]] 와 같은 복잡한 리스트여도 그냥 크게 정보 3개 들어있어서 len하면 3개 나오더라.

4) .upper()

-대문자로 바꿔주는 함수

-ex) b='bus타고 학교가야지'

a=b.upper() ;a 하면 'BUS타고 학교가야지' 가 나옴.

5) .find()

-무언가를 찾아주는 함수. 띄어쓰기 포함해서 왼쪽부터 몇 번째에 나오는 지 알려주는 함수

-ex)abc=' this is a house built this year. /n'

abc.find('is') 하면 3이 나옴. (주의: 'is' 라고해도 this안에 is가 포함되어 있어서 3 나옴.

-cf)find 함수는 int에 사용이 불가능

6) .rindex()

-find함수와 비슷한 개념인데 첫 번째 말고 마지막 index를 왼쪽부터 counting하는 함수

-즉, 위에서 is를 하면 built this의 is를 몇 번째부터인지 찾아준다는 것.

7) .strip()

-잡스러운 것들을 정리해주는 함수. space도 깔끔하게 정리해 줌.

-ex) a=' this is a house built this year. \n'

일 때 `b=a.strip()` ; `a` 하면 'this is a house built this year.' 이 나옴.

8) `.split(' ')`

-긴 str이 들어간 것을 단어 별로 잘라서 list로 만들어 줌.

-ex) `a='this is the best place.'`

`b=a.split(' ');` b 하면 ['this', 'is', 'the', 'best', 'place.']가 나옴.

여기서 cf) '!'는 마지막 정보에 묶여서 나오더라

9) `' '.join()`

-list를 다시 string으로 만들고 싶을 때 사용.

-' ' 사이에 space말고 ,를 사용하면 , 넣은 상태로 붙여 줌.

ex) `jocker=['guy', 'who', 'is', 'bad']`

`' '.join(jocker)` 하면 'guy who is bad'가 나옴.

`','.join(jocker)` 하면 'guy, who, is, bad'가 나옴.

10) `.replace(,)`

-예를 들어 `.replace(this,that)` 하면 this를 that으로 바꿔주는 암튼 그런 함수임.

for, if

1. for

-for a in b: b에 있는 것을 한 번씩 돌려서 a로 받아서 무언가를 하라. 그 무언가는 for 밑에 적어 줌.

ex) for i in a:

`print(i)`

라고 한다면 a에 있는 정보들을 하나씩 불러서 i에 할당하고 이를 print해라 라는 뜻임.

`a=[1,2,3,4]` 일 경우 1,2,3,4,가 나옴.

-range: range뒤에 어떤 숫자가 나오면 list를 만들어 줌.

ex) `range(3)`이면 0~2까지의 리스트를 만들어 줌. 즉, 괄호 안의 숫자는 index의 개수를 말해 줌.

-for을 사용할 때, list를 그대로 주는 방법과 range함수를 쓰는 방법이 있다.

ex) `a=[1,2,3,4]`

`for i in range(3)`

`print(a[i])` 하면 1,2,3,4 나옴.

-여기서 cf) 리스트 내의 정보의 개수보다 range값이 크면 앞에 것들은 출력되고 넘어가는 범위의 것은 error가 뜬다.

-range 함수 효율적으로 이용하는 방법: len 함수와 함께 사용하는 것.


```
ex)a=['red', 'green', 'blue', 'purple']  
    for i in range(len(a))  
        print(a[i]) 하면 정보 다 출력 됨.
```

enumerate

-번호를 매김. 앞에 있는 변수가 번호, 뒤에 있는 것이 자기 자신의 element. 즉, 첫 번째 variable 에 index, 두 번째에 각각의 정보를 담음.

```
ex) )a=['red', 'green', 'blue', 'purple']  
    b=[0.2, 0.3, 0.4, 0.5]  
    for i, s enumerate(a)  
        print("{}:{}".format(s,b[i]*100))
```

하면 red:20% green:30% 이런식으로 다 나옴.

format함수는 "{}:{}".format(a,b)했을 때 a, b를 앞의 중괄호 안에 각각 넣어서 출력하는 함수임.

zip

-두 개의 길이가 같은 list가 있을 때 사용.

-zip를 함으로써 a, b가 pair가 된 느낌.

```
ex) )a=['red', 'green', 'blue', 'purple']  
    b=[0.2, 0.3, 0.4, 0.5]  
    for s, i in zip(a,b)  
        print("{}:{}".format(s,i*100))
```

하면 위와 같은 결과가 나옴.

2. if

-if도 indent 필요함

-a==0하면 0과 같은 것

a!=0하면 0이 아닌 것

a<=0하면 0보다 작거나 같은

a>=0하면 0보다 크거나 같은.

-else : 아닐 경우 이 것을 해라.

-ex)a==0

if a==0

print("yay!")

하면 yay!가 나옴.

-loop의 개념(중요)

```
for i in range(1,3):  
    for j in range(3,5):  
        print(i*j)
```

할 때, 위에 두 개 아래 두 개 해서 곱하면 총 4번의 루프가 돌 수 있음.

ex)

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

하면 4. 8이 나옴.

-error 뜨는 경우 ex

```
for i in range(1,3):  
    for j in range(3,5):  
        if j=6:  
            print(i*j)
```