순서가 있는 데이터 구조

작성일시	@2022년 7월 25일 오후 3:44
내용	문법 자료구조 제어문 파이썬
주차	2주차
자료	

문자열(String)

- 문자들의 나열
  - 모든 문자는 str타입(immutable)
- 문자열은 작은따옴표(')나 큰 따옴표(")를 활용하여 표기
  - 문자열을 묶을 때 동일한 문장부호 활용
  - PEP8에서는 소스코드 내에서 하나의 문장부호를 선택하여 유지하도록 함

문자열 조회/탐색 및 검증 메서드

문법	설명
s.find(x)	x의 첫 번째 위치를 반환. 없으면 **-1을 반환**
s.index(x)	x의 첫 번째 위치를 반환. 없으면 **오류 발생**
s.isalpha( )	알파벳 문자 여부 *단순 알파벳이 아닌 유니코드상 letter(한국어 포함)
s.isuuper( )	대문자 여부
s.islower( )	소문자 여부
s.istitle( )	타이틀 형식 여부

- .find(x)
  - x의 첫 번째 위치를 반환. 없으면 **-1을 반환(오류 발생x)**

```
word = 'apple'

print(word.find('p')) # 1
print(word.find('k')) # -1
```

- .index(x)
  - x의 첫 번째 위치를 반환. 없으면 **오류 발생**

```
word = 'apple'

print(word.index('p')) # 1
print(word.index('k')) # ValueError: substring not found
```

- 문자열 관련 검증 메서드 사용 예시

```
print('abc'.isalpha()) # True
print('ㄱㄴㄷ'.isalpha()) # True
print('Ab'.isupper()) # False
print('ab'.islower()) # True
print('Title Title!.istitls()) # True
```

- 문자열 관련 검증 메서드
  - .isdecimal( ) ≤ .isdigit( ) ≤ is.numeric( )

문자열 변경 메서드(s는 문자열)

문법	설명
s.replace(old, new[,count]	바꿀 대상이나 글자를 새로운 글자로 바꿔서 반환
s.strip([chars])	공백이나 특정 문자를 제거
s.split(sep=None, maxsplit = -1)	공백이나 특정 문자를 기준으로 분리
‘seperator’.join([iterable])	구분자로 iterable을 합침
s.caplitalize( )	가장 첫 번째 글자를 대문자로 변경
s.title( )	문자열 내 띄어쓰기 기준으로 각 단어의 첫 글자는 대문자로 ,나머지는 모두 소문자로 변경
s.upper( )	모두 대문자로 변경
s.lower( )	모두 소문자로 변경
s.swapcase( )	대↔소문자 서로 변경

```
print('coone'replace('o', 'a')) # caane
print('wooooooowoo'.replace('o', 'e', 2)) # weeooooowoo

print('     와우!\n'.strip()) # '와우!'
print('     와우!\n.lstrip()) # '와우!'
print('     와우!\n.rstrip()) # '     와우!'
```

```
print('안녕하세요???'.rstrip('?')) # '안녕하세요'

print('a, b, c'.split(',')) # ['a', ' b', ' c']
print('a b c'.split()) # ['a', 'b', 'c']
print('서울시 강남구 00동'.split()[0]) # '서울시'

print('!'.join('hello')) # 'h!e!l!l!o'
print(' '.join(['3', '5']) #  '3 5'
```

리스트(List)

리스트 메서드

문법	설명
L.append(x)	리스트 마지막 항목에 x를 추가
L.insert(i, x)	리스트 인덱스 i에 항목 x를 삽입
L.remove(x)	리스트 가장 왼쪽에 있는 항목(첫 번째) x를 제거 항목이 존재하지 않을 경우, ValueError
L.pop( )	리스트 가장 오른쪽에 있는 항목(마지막)을 반환 후 제거
L.pop(i)	리스트의 인덱스 i에 있는 항목을 반환 후 제거
L.extend(m)	순회형 m의 모든 항목들을 리스트 끝에 추가(+=와 같은 기능)
L.index(x, strat, end)	리스트에 있는 항목 중 가장 왼쪽에 있는 항목 x의 인덱스를 반환
L.reverse( )	리스트를 거꾸로 정렬
L.sort( )	리스트를 정렬(매개변수 이용 가능)
L.count(x)	리스트에서 항목 x가 몇 개 존재하는지 갯수를 반환

```
cafe = ['starbucks', 'tomntoms', 'hollys']

# insert

cafe.insert(0, start')
print(cafe) # ['start', 'starbucks', tomntoms', 'hollys']

# 리스트 길이를 넘어가는 경우에는 맨 끝으로
cafe.insert(10000, 'end')
print(cafe) #['starbucks', 'tomntoms', 'hollys', 'end']

# extend(iterable)

cafe.extend(['coffee'])
print(cafe) # ['starbucks', 'tomntoms', 'hollys', 'coffee']
```

```
cafe.extend('cup')
print(cafe) # #['starbucks', 'tomntoms', 'hollys', 'c', 'u', 'p']
```

```
# remove

numbers = [1, 2, 3, 'hi']

numbers.remove('hi')
print(numbers) #[1, 2, 3]

# pop

word = numbers.pop()
print(word) # hi
print(numbers) # [1, 2, 3]

# clear

numbers.clear()
print(numbers) # []
```

- 리스트 정렬 관련 메서드
  - .sort( ) / .sorted( )
    - sort(): **원본** 함수를 정렬
    - sorted(): **원본의 복사본**을 정렬 (원본은 그대로!)

```
# sort()
numbers = [3, 2, 5, 7]
result = numbers.sort()

print(numbers, result) # [2, 3, 5, 7], None

# sorted()
numbers = [3, 2, 5, 7]
result = sorted(numbers)
print(numbers, result) # [3, 2, 5, 7], [2, 3, 5, 7]
```

  - .reverse( )

```
numbers = [3, 2, 5, 1]
result = numbers.reverse()
print(numbers, result) #[1, 2, 3, 5], None
```

튜플(Tuple)

- 튜플은 변경할 수 없기 때문에 값에 영향을 미치지 않는 메서드만을 지원
- 리스트 메서드 중 항목을 변경하는 메서드들을 제외하고는 대부분 동일

멤버십 연산자

- in / not in

```
print('a' in 'apple') # True
print('b' not in 'apple') # True
print('hi' in 'hi I am Sam') # True
print('서순' in ['서순', '요까일엇무', '기러기']) # True
```