파이썬 프로그래밍

17차시

리스트의 부분 참조와 항목의 삽입과 삭제

⚠ 학습개요

… 리스트 부분 참조(슬라이싱)

… 리스트 중간에 삽입

… 리스트 항목 삭제

… 문장 del

… 리스트 마지막에 리스트를 이어 붙이기

⚠ 학습목표

… 리스트의 부분 참조를 할 수 있고 대입할 수 있다.

… 리스트의 중간, 마지막에 항목을 삽입할 수 있다.

… 리스트나 항목을 삭제할 수 있다.

… 리스트를 이어 붙일 수 있다.

Chapter 1.

리스트
부분 참조

PYTHON PROGRAMMING

⚠ 리스트의 부분 참조인 슬라이싱

+ 리스트[start:stop:step]

- 첨자 start에서 첨자 stop-1까지 step 간격으로 부분 리스트 반환
- step이 **양수: 오른쪽**으로 구성
  - start에 시작하는 순방향(오름차순, 수가 차례로 늘어가는 것)
- step이 **음수: 왼쪽**으로 구성
  - start에 시작하는 역방향(내림차순, 수가 차례로 줄어가는 것)

+ 리스트 슬라이싱의 결과

- 첨자에 해당하는 부분 리스트가 없으면 결과는 빈 리스트: []

⚠ 리스트의 부분 참조인 슬라이싱

```
>>> alp = list('abcdefghij')
>>> print(alp[1:-1])
['b', 'c, 'd', 'e', 'f', 'g', 'h', 'i']
>>> print(alp[-1:1:-1])
['j', 'i', 'h, 'g', 'f', 'e', 'd', 'c']
>>> print(alp[-2:2:-2])
['i', 'g', 'e']
```

⚠ 일상 코딩: 한 글자의 단어로 슬라이싱 이해하기

[코딩실습] 한 글자의 한국 단어로 이해하는 리스트 슬라이싱 난이도 기본

```
1. wlist = ['밥', '삶', '길', '죽', '꿈', '차', '떡', '복', '말']
2. print('wlist[:] = ', wlist[:])
3. print('wlist[::] = ', wlist[::])
4. print('wlist[::-1] = ', wlist[::-1])
5. # 순방향
6. print(wlist[::3])
7. print(wlist[1::3])
8. print(wlist[2::3])
9. # 역방향
10.print(wlist[::-2])
11.print(wlist[-1:-8:-3])
12.# 순방향과 역방향, 다양한 첨자
13.print(wlist[1:-1:])
14.print(wlist[-2:-9:-3])
```

⚠ 일상 코딩: 한 글자의 단어로 슬라이싱 이해하기

주의 5, 7번 줄은 들여쓰기로 입력한다.

결과

```
wlist[:] = ['밥', '삶', '길', '죽', '꿈', '차', '떡', '복', '말']
Wlist[::] = ['밥', '삶', '길', '죽', '꿈', '차', '떡', '복', '말']
Wlist[::-1] = ['말', '복', '떡', '차', '꿈', '죽', '길', '삶', '밥']
['밥', '죽', '떡']
['삶', '꿈', '복']
['길', '차', '말']
['말', '떡', '꿈', '길', '밥']
['밀', '차', '길']
['삶', '길', '죽', '꿈', '차', '떡', '복']
['복', '꿈', '삶']
```

Upgrade coding

첫 줄의 한 글자 단어로 구성되는 wlist를 만드는 코드가 번거로울 수 있다.
이 문장을 보다 간소화 시킬 수 있는 코드로 수정해 보자.

힌트 함수 lsit()와 문자열 활용

⚠ 리스트의 슬라이스로 리스트의 일부분을 수정

+ 리스트의 일부분을 다른 리스트로 수정하려면

  슬라이스 방식에 대입

```
>>> sports = ['풋살', '족구', '비치사커', '야구', '농구', '배구']
>>> sports[0:3] = ['축구']
>>> print(sports)
['축구', '야구', '농구', '배구']
```

```
          첨자  0      1        2       3      4      5
>>> sports = ['풋살', '족구', '비치사커', '야구', '농구', '배구']
>>> sports[1:3] = sports[4:6]
>>> print(sports)
['풋살', '농구', '배구', '야구', '농구', '배구']
```

⚠ 리스트의 슬라이스로 리스트의 일부분을 수정

+ 리스트의 일부분을 다른 리스트로 수정하려면

  슬라이스 방식에 대입

```
>>> sports = ['풋살', '족구', '비치사커', '야구', '농구', '배구']
>>> others = ['축구', '미식축구', '골프']
>>> sports[0:4] = others[:2]
>>> print(sports)
['풋살', '축구', '미식축구', '농구', '배구']
```

Chapter 2.

리스트 중간에 삽입

PYTHON PROGRAMMING

⚠ 리스트 메소드 insert(첨자, 항목)으로 삽입

+ 리스트의 첨자 위치에 항목을 삽입

- 리스트.**insert(첨자, 항목)**
- 삽입되는 항목은 무엇이든 가능
- **빈 리스트에도 삽입 가능**

```
>>> kpop = []
>>> kpop.insert(0, '블랙핑크')
>>> kpop.insert(0, 'BTS')
>>> kpop
['BTS', '블랙핑크']
>>> kpop.insert(1, '장범준')
>>> kpop
['BTS', '장범준', '블랙핑크']
```

Chapter 3.

리스트 항목 삭제

PYTHON PROGRAMMING

⚠ 리스트 항목 삭제

+ 메소드 remove(항목)

```
>>> kpop = ['BTS', '블랙핑크', '장범준', '잔나비']
>>> kpop.remove('장범준')
>>> print(kpop)
['BTS', '블랙핑크', '잔나비']
```

```
>>> kpop.remove('장준')
Traceback (most recent call last):
 File "<stdin>",line 1 , in <module>
ValueError: list.remove(x): x not in list
```

```
>>> if '장준' in kpop:
...     kpop.remove('장준')
...
>>> print(kpop)
['BTS', '블랙핑크', '잔나비']]
```

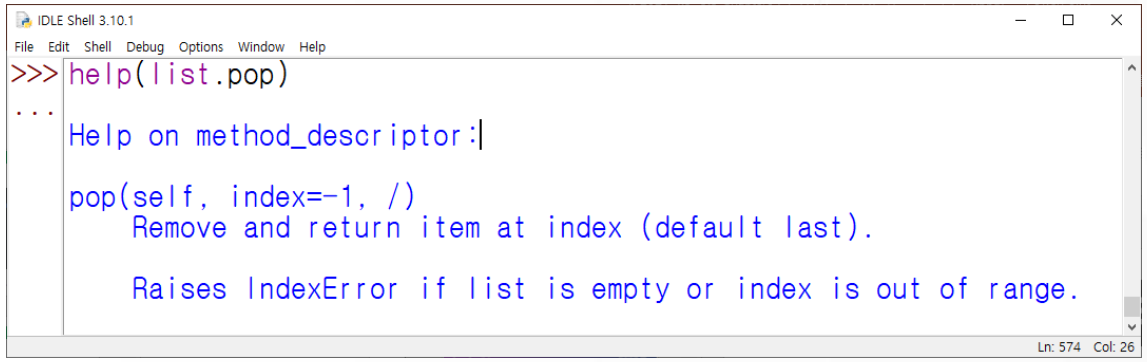
⚠ 리스트 항목 삭제

+ pop(첨자)

+ pop( )

- Remove and return item at index (default last).

```
>>> kpop = ['BTS', '장범준', '블랙핑크', '잔나비']
>>> print(kpop.pop(1))
장범준
>>> print(kpop.pop())
잔나비
>>> print(kpop)
['BTS', '블랙핑크']
```



끝

Chapter 4.

문장 del

PYTHON PROGRAMMING

⚠ 함수 del()과 메소드 clear()

+ 문장 del

- **항목 삭제**와 **부분 삭제**
- 리스트 자체를 메모리에서 **완전 삭제 가능**

⚠ 함수 del()과 메소드 clear()

```
>>> kpop = ['BTS', '장범준', '블랙핑크', '잔나비']
>>> del kpop[0] # del(kpop[0])도 가능
>>> print(kpop)
['장범준', '블랙핑크', '잔나비']
```

```
>>> kpop = ['BTS', '장범준', '블랙핑크', '잔나비']
>>> del kpop[0:2]
>>> print(kpop)
['블랙핑크', '잔나비']
```

```
>>> kpop = ['BTS', '장범준', '블랙핑크', '잔나비']
>>> del kpop
>>> print(kpop)
Traceback (most recent call last):
 File "<stdin>",line 1 , in <module>
NameError: name 'kpop' is not defined
```

⚠ 함수 del()과 메소드 clear()

+ 메소드 clear( )

- 리스트의 **모든 항목이 삭제**
- 빈 리스트로 만듦

```
>>> kpop = ['BTS', '장범준', '블랙핑크', '잔나비']
>>> del.clear()
>>> print(kpop)
[]
```

PYTHON PROGRAMMING

⚠ 일상 코딩: 자신의 학용품 품목과 개수를 리스트에 삽입과 삭제

[코딩실습] 학용품 리스트의 항목 삽입과 삭제

난이도 기본

```
1. item = ['연필', '볼펜']
2. # 현재 학용품 품목 출력
3. print(item)
4.
5. # 연필 1개와 볼펜 세 자루 삽입
6. item.insert(1, 2)
7. item.insert(3, 3)
8. # 맨 뒤에 지우개, 1개 삽입
9. item.insert(4, '지우개')
10.item.insert(5, 1)
11.# 현재 학용품 품목 출력
12.print(item)
13.
```

⚠ 일상 코딩: 자신의 학용품 품목과 개수를 리스트에 삽입과 삭제

[코딩실습] 학용품 리스트의 항목 삽입과 삭제 난이도 기본

```
14.# 연필 두 자루 삭제
15.print(item.pop(1)) # 첨자 1항목 삭제
16.item.remove('연필') # 항목 연필 항목 삭제
17.del item[2:] # 지우개와 수 품목 삭제
18.
19.# 현재 학용품 품목 출력
20.print(item)
```

주의 15번 줄의 item.pop(1)은 삭제된 것을 반환하므로 print()로 삭제된 내용을 출력할 수 있다.

결과

```
['연필', '볼펜']
['연필', 2, '볼펜', 3, '지우개', 1]
2
['볼펜', 3]
```

끝

Chapter 5.

리스트 마지막에 리스트를 이어 붙이기

PYTHON PROGRAMMING

⚠ 리스트에 리스트를 추가하는 메소드 extend( )

+ 리스트.extend(list)

- 리스트에 인자인 **list를 가장 뒤에 추가**

```
>>> day = ['월', '화', '수']
>>> day2 = ['목', '금', '토', '일']
>>> day.extend(day2)
>>> print(day)
['월', '화', '수', '목', '금', '토', '일']
```

⚠ 리스트에 리스트를 추가하는 메소드 extend( )

+ + 연산자

- 리스트를 연결

```
>>> korean = ['불고기', '설렁탕']
>>> chinese = ['탕수육', '기스면']
>>> food = Korean + chinese
>>> print(food)
['불고기', '설렁탕', '탕수육', '기스면']
```

⚠ 리스트에 리스트를 추가하는 메소드 extend( )

+ * 연산자

- 항목이 지정된 정수만큼 반복된 리스트를 반환

```
>>> days = ['월', '화']
>>> print(days * 3)
['월', '화', '월', '화', '월', '화']
```

⚠리스트 부분 참조

… 리스트 슬라이싱
… 리스트의 슬라이싱으로 리스트의 일부분을 수정

⚠리스트 중간에 삽입

… 리스트 메소드 insert(첨자, 항목)으로 삽입

⚠리스트 항목 삭제

… 메소드 remove(항목)
… 메소드 pop(첨자), pop()
… 메소드 clear()

⚠문장 del

… del obj
… del obj[i]

⚠ 리스트 마지막에 리스트를 이어 붙이기

… 메소드 extend(list)

… 연산자 +

… 반복 붙이기: 연산자 *