파이썬 프로그래밍

리스트의 기초

한국기술교육대학교
온라인평생교육원

▣ 리스트 메소드

1. 리스트가 지원하는 메소드

```
s = [1, 2, 3]

s.append(5) # 리스트 맨 마지막에 정수 값 5 추가
print s

s.insert(3, 4) # 3 인덱스 위치에 정수 값 4 추가
print s
```

```
['begin', [1, 100, 3], 'end']
```

- s.appened(5) → s 리스트 값 뒤에 5 값을 추가
- s.appened 뒤에 문자열, 튜플, 사전 가능
- s.insert(3,4) → 첫번째 객체는 두번째 객체가 들어가야 할 위치
- 인덱스 3에 해당하는 5 자리가 밀리고 4가 들어감

▣ 리스트 메소드

1. 리스트가 지원하는 메소드

```
s = [1, 2, 3, 4, 5]

print s.index(3)         # 값 3의 인덱스 반환

print s.count(2)         # 값 2의 개수 반환

s = [1, 2, 2, 2, 2, 2, 3, 4, 5]
print s.count(2)
```

```
2
1
5
```

- s.index(3) → 3의 인덱스를 반환
- abc 인덱스가 5
- s.count(2) → s 안에 2가 몇 개 들어있는지 조사

▣ 리스트 메소드

1. 리스트가 지원하는 메소드

•python의 소팅 알고리즘: Timsort (변형된 merge sort)
  - 참고: http://orchistro.tistory.com/175

```
s = [1, 2, -10, -7, 100]
s.reverse() # 자료의 순서를 뒤집기 (반환값 없음)
print s

s.sort() # 정렬 (반환값 없음)
print s
```

```
[100, -7, -10, 2, 1]
[-10, -7, 1, 2, 100]
```

- s.reverse() → s의 순서를 뒤집는 것
- s.sort() → s 안 원소를 정렬
- 정수는 작은 순서대로 정렬

▣ 리스트 메소드

1. 리스트가 지원하는 메소드

```
s = [10, 20, 30, 40, 50]
s.remove(10) # 자료 값 10 삭제
print s

s = [10, 20, 30, 20, 40, 50] # 자료 값이 여러개 존재하면 첫번째 것만 삭제
s.remove(20)
print s

s.extend([60, 70]) # 새로운 리스트([60, 70]를 기존 리스트 s 뒤에 병합
print s

s.append([60, 70]) # 주의: append로 새로운 리스트를 추가하면 하나의 자료 요소로서
                        추가
print s
```

```
[20, 30, 40, 50]
[10, 30, 20, 40, 50]
[10, 30, 20, 40, 50, 60, 70]
[10, 30, 20, 40, 50, 60, 70, [60, 70]]
```

- s.remove(10) → s 안에 10 원소 삭제
- remove 뒤 값이 여러 번 나올 때는 첫번째 값만 삭제
- s.extend([60, 70]) → s에 60, 70을 그대로 추가
- extend 뒤에 리스트가 오면 값을 뽑아다가 넣어 리스트를 확장
- appened는 뒤의 인자를 element로 인식 → 중첩된 리스트가 됨

▣ 리스트 메소드

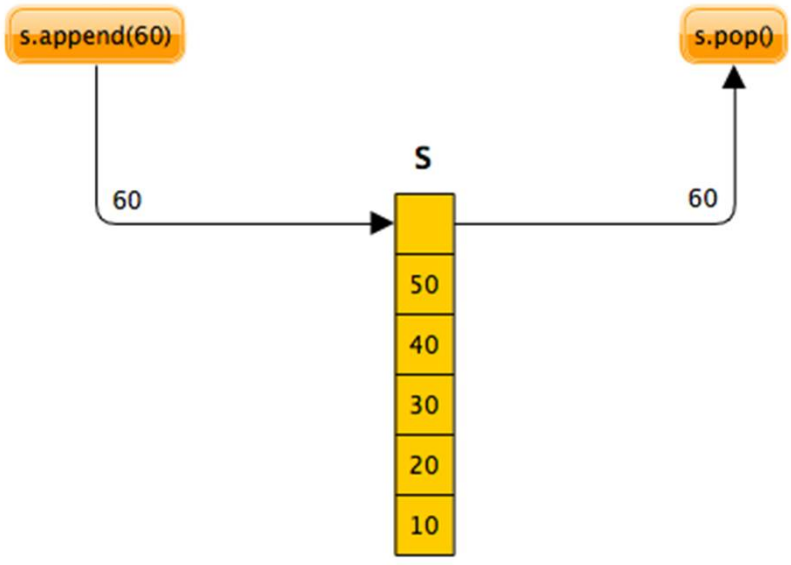
2. 리스트를 스택(Stack)으로 사용하기

```
s = [10, 20, 30, 40, 50]
s.append(60)
print s

print s.pop()

print s
```

```
[10, 20, 30, 40, 50, 60]
60
[10, 20, 30, 40, 50]
```



- s.pop() → s 리스트의 마지막 원소를 뺌

▣ 리스트 메소드

2. 리스트를 스택(Stack)으로 사용하기

```
s = [10, 20, 30, 40, 50]
print s.pop(0)   #0 번째 인덱스 값을 꺼낸다.

print s

print s.pop(1)   #1 번째 인덱스 값을 꺼낸다.

print s
```

```
10
[20, 30, 40, 50]
30
[20, 40, 50]
```

- s.pop(0) → 0번째 인덱스 값을 뺌

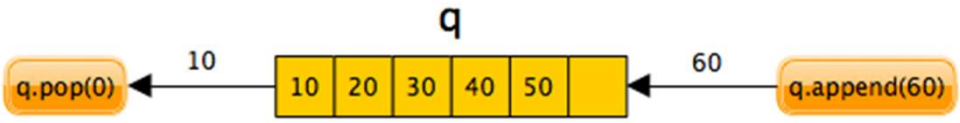
▣ 리스트 메소드

3. 리스트를 큐(Queue)로 사용하기

```
q = [10, 20, 30, 40, 50]
q.append(60)
print q.pop(0)

print q
```

```
10
[20, 30, 40, 50, 60]
```



- 큐: 먼저 들어오는 것은 먼저, 마지막에 들어온 것은 마지막에 나감