

배열 리스트 (Array List)

- 01 배열 리스트 이론
- 02 배열/리스트 문제 풀이

신 제 용

01 배열 리스트 이론

리스트 추상 자료형과, 내부적으로 배열로 구현된 배열 리스트 (Array List)에 대해서 알아봅니다.

학습 키워드 - 배열, 리스트, 추상자료형, 자료구조

Chapter 01
배열 리스트 이론

리스트 (List)

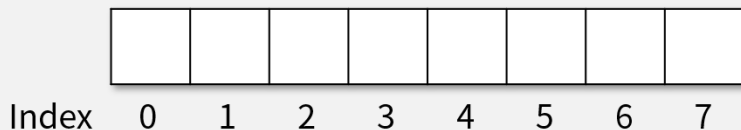
- 순서가 있는 자료를 다루는 **추상 자료형**
- 추상 자료형이기 때문에, 구현 방법이 명시되어 있지 않음
- 대표적인 리스트를 구현한 자료구조 – 배열 리스트, 연결 리스트

Chapter 01

배열 리스트 이론

리스트의 연산자

- 비어있는 리스트를 생성하는 연산자 (`__init__()`)



1. Constructor

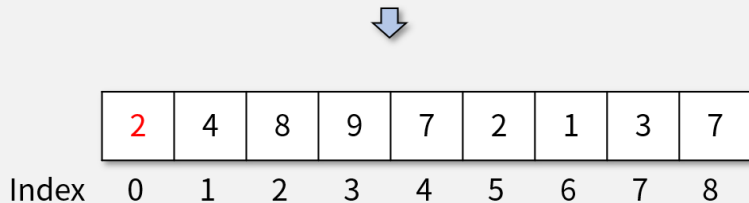
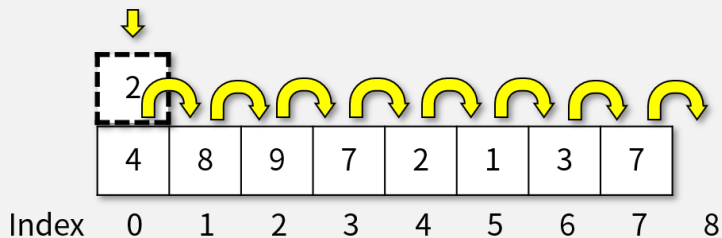
- 리스트가 비어있는지 확인하는 연산자 (`is_empty()`)

Chapter 01

배열 리스트 이론

리스트의 연산자

- 리스트의 앞에 개체를 삽입하는 연산자 (prepend())

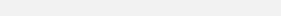


3. Prepend

Chapter 01
배열 리스트 이론

리스트의 연산자

- 리스트의 뒤에 개체를 삽입하는 연산자 (append())



4	8	9	7	2	1	3	7	2	
Index	0	1	2	3	4	5	6	7	8



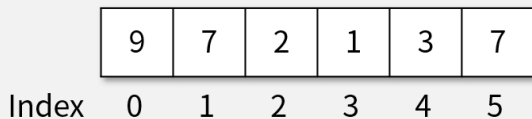
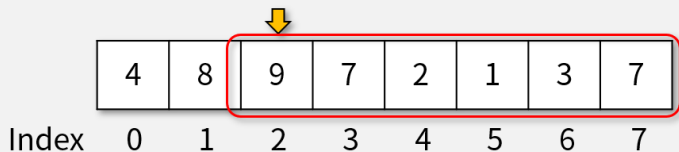
	4	8	9	7	2	1	3	7	2
Index	0	1	2	3	4	5	6	7	8

4. Append

Chapter 01
배열 리스트 이론

리스트의 연산자

- 리스트의 첫 머리(head)를 결정하는 연산자 (set_head())

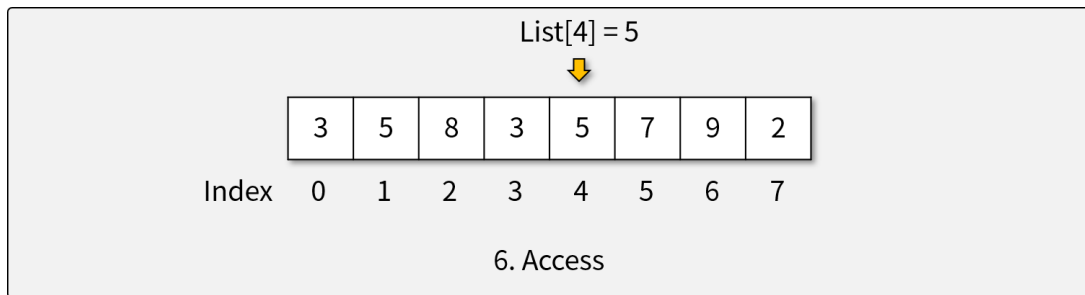


5. setHead

Chapter 01
배열 리스트 이론

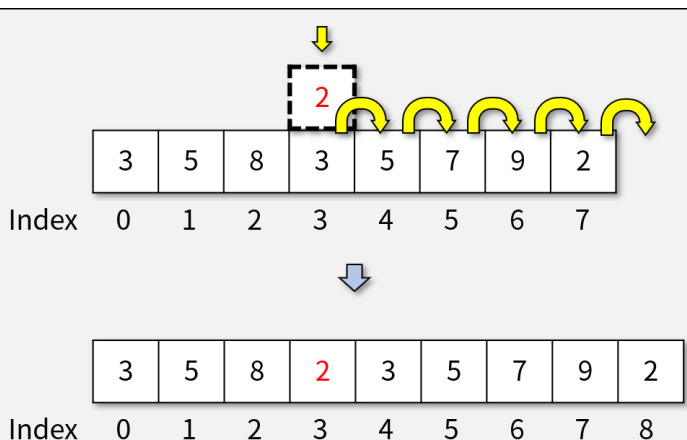
리스트의 연산자

- 주어진 인덱스에 접근하는 연산자 (`access()`, `get()`)



리스트의 연산자

- 주어진 인덱스에 삽입하는 연산자 (insert())

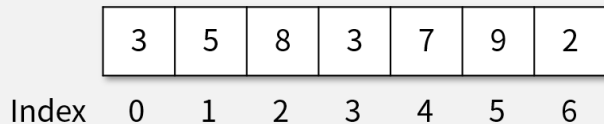
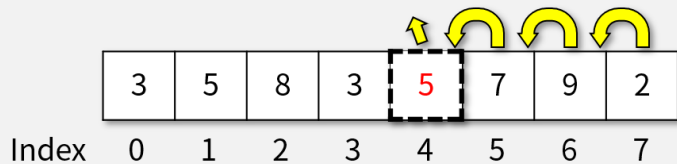


7. Insert

Chapter 01
배열 리스트 이론

리스트의 연산자

- 주어진 인덱스의 요소를 제거하는 연산자 (remove())



8. Remove

Chapter 01
배열 리스트 이론

배열 리스트 (Array list)

- 내부적으로 배열을 이용하여 구현하는 리스트
- 리스트 추상자료형에서 요구하는 연산자를 배열을 활용하여 구현
- 배열의 장점인 임의 접근(random access)가 핵심!

Chapter 01

배열 리스트 이론

Python과 리스트

- Python에는 잘 구현된 리스트 자료형이 있다!
- 모든 자료형을 허용하는, 매우 유연한 자료구조
- 지능형 리스트와 Pre-allocation 기법을 이용해 고성능으로 사용하자!

Chapter 01

배열 리스트 이론

02 배열/리스트 문제 풀이

다음 챕터에서는 배열과 리스트에 관한
기초 유형 문제를 풀이합니다.

Chapter 01

배열 리스트 이론

02 배열/리스트 문제 풀이

배열과 리스트를 활용하여 문제를 해결하는 방법을 배워봅시다.

학습 키워드 - 배열, 리스트

Chapter 02

배열/리스트 문제 풀이

```
# Prob1.  
# 리스트 arr의 모든 자료에 대해서,  
# 짝수 데이터들의 평균과 홀수 데이터들의 평균을 각각 계산하시오.  
# 출력은 2개의 출력을 리스트로 묶어 출력하시오.  
#  
# 입출력 예시)  
# arr = [1, 3, 5, 7, 9, 2, 5, 1, 4]  
# 결과: [3.25, 4.8]
```

```
def solution(arr):  
    pass
```

```
if __name__ == '__main__':  
    arr = [1, 3, 5, 7, 9, 2, 5, 1, 4]  
    sol = solution(arr)  
    print(sol)
```

Chapter 02

배열/리스트 문제 풀이

```
# Prob2.  
# 리스트의 자료 순서를 거꾸로 변경하시오.  
# 추가 리스트를 생성하지 말고, 주어진 리스트에서 in-place로 하시오.  
#  
# 입출력 예시)  
# arr = [1, 3, 5, 7, 9]  
# 결과: [9, 7, 5, 3, 1]
```

```
def solution(arr):  
    pass
```

```
if __name__ == '__main__':  
    arr = [1, 3, 5, 7, 9]  
    sol = solution(arr)  
    print(sol)
```

Chapter 02

배열/리스트 문제 풀이


```
# Prob3.  
# 리스트 arr에 존재하는 모든 피크 값을 출력하시오.  
# 단, 피크 인덱스란  
#  
# arr[i-1] < arr[i], arr[i+1] < arr[i]  
#  
# 를 동시에 만족하는 인덱스 i를 말하며,  
# 피크 값은 이 때 arr[i]를 말한다.  
#  
# 입출력 예시)  
# arr = [3, 1, 2, 6, 2, 2, 5, 1, 9, 10, 1, 11]  
# 결과: [6, 5, 10]
```

```
def solution(arr):  
    pass
```

```
if __name__ == '__main__':  
    arr = [3, 1, 2, 6, 2, 2, 5, 1, 9, 10, 1, 11]  
    sol = solution(arr)  
    print(sol)
```

Chapter 02

배열/리스트 문제 풀이

```
# Prob4.  
# 이차원 리스트 arr를 시계방향으로 90도 회전시킨 결과를 출력하시오.  
#  
# 입출력 예시)  
# arr = [[ 1,  2,  3,  4,  5],  
#        [ 6,  7,  8,  9, 10],  
#        [11, 12, 13, 14, 15]]  
# 결과: [[11, 6, 1],  
#        [12, 7, 2],  
#        [13, 8, 3],  
#        [14, 9, 4],  
#        [15, 10, 5]]
```

```
def solution(arr):  
    pass  
  
if __name__ == '__main__':  
    arr = [[ 1,  2,  3,  4,  5],  
           [ 6,  7,  8,  9, 10],  
           [11, 12, 13, 14, 15]]  
    sol = solution(arr)  
    print(sol)
```

Chapter 02

배열/리스트 문제 풀이