알고리즘 e러닝

최적의 코딩을 결정하는 기본 알고리즘

이것이 코딩테스트다 강의와 순서만 다름

1. 가장 기본이 되는 자료구조: 스택과 큐

1. 스택

- 먼저 들어온 데이터가 나중에 나가는 형식. 선입후출. LIFO
- 입구와 출구가 동일한 형태로 스택을 시각화할 수 있다.
- 박스쌓기
- 동작 방법과 사용예시 알아둬야 할 필요가 있음. B?D?FS 등에서 자주 쓰임

-구현 by python

리스트 자료형 사용하면 됨

```
stack = []
#파이썬에서는 스택 구현에 리스트 사용

stack.append(5)
stack.append(2)
stack.append(3)
stack.append(7)
stack.pop()
stack.append(1)
stack.append(4)
stack.pop()

print(stack[::-1]) # 최상단 원소부터 출력
print(stack) # 최하단 원소부터 출력

# 파이썬에서는 별도로 표준라이브러리 이용할 필요 없이 리스트를 이용하면 된다.
```

파이썬에서는 별도로 표준라이브러리 이용할 필요 없이 리스트를 이용하면 된다.

자바에서는 push, pop으로 삽입삭제, 최상단부터 출력할 때는 peek사용!

2. 큐

- 먼저 들어온 데이터가 먼저 나가는 형식의 자료구조. 선입선출
- 큐는 입구와 출구가 모두 뚫려있는 터널과 같은 형태로 시각화 가능.
- 줄서있는 모습. 대기열. 먼저온사람 차례대로
- 데이터가 들어가는 쪽이 rear, 나오는 쪽이 front

구현 by python

```
from collections import deque

#큐 구현할 때는 덱 라이브러리 사용
queue = deque()

queue.append(5)
queue.append(2)
```

```
queue.append(3)
queue.append(7)
queue.popleft()
queue.append(1)
queue.append(4)
queue.popleft()
#삽입은 append, 삭제는 popleft

print(queue) # 먼저 들어온 순으로 출력
queue.reverse() # 역순으로 바꾸기
print(queue) # 나중에 들어온 순으로 출력

# 리스트를 이용해도 기능적으로는 큐를 구현할 수도 있지만
# 시간 복잡도가 높아서 비효율적. pop할시 꺼낸 다음 나머지를 앞으로 땡겨야해서 시간복잡도가 O(n)
# 엄밀히 말하면 덱 라이브러리는 스택과 큐의 장점을 합친 구조의 자료구조라고 볼 수 있음. list의 append와 동일하게 작동함. 오른쪽으로 삽입. 시간복잡도 상수시간. O(1)
# popleft도 가장 왼쪽에 있는 데이터 삭제니 O(1)
#출력결과
#deque([3, 7, 1, 4])
#deque([4, 1, 7, 3])
```

자바에서는 삽입offer, 삭제 poll 사용.

먼저 들어온 원소부터 추출하려면

System.out.print(q.poll() + " ");

2. 우선순위에 따라 데이터를 꺼내는 자료구조

3. 활용도가 높은 자료구조: 트리 자료구조

4. 특수한 목적의 자료구조: 바이너리 인덱스 트리