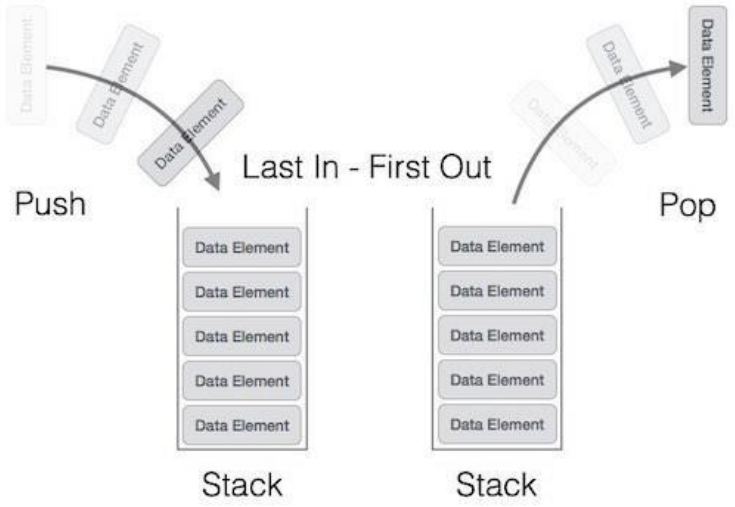
자구&알고 스터디

스택, 큐, 덱, 2차원 배열

스택 (Stack)

- 후입 선출 : 가장 마지막으로 들어온 자료가 가장 먼저 나감
- 간단하게 생각하면 종이를 쌓아 두는 것을 생각하면 편함
  - 스택을 활용하면 "롤 백" 할 수 있다는 상상을 하면 좋음
  - 브라우저의 "뒤로 가기"의 대표적인 구현 방법이 스택 활용방법.



스택 (Stack)

- 구현 방법도 간단함
  - 배열 (또는 리스트)의 끝자리를 확인, 끝자리에 삽입, 끝자리 원소를 삭제가 가능하기만 하면 됨.
- 예시 1 (파이썬)

```
arr = []
arr.append(1) # 마지막에 삽입
arr[-1] # 마지막에 삽입 된 원소 확인
arr.pop() # 마지막 원소 삭제
```

- 예시 2 (파이썬)

```
arr = []
endIndex = 0 # 마지막 원소의 인덱스 + 1
arr[endIndex]; endIndex += 1 # 마지막에 삽입
arr[endIndex-1] # 마지막에 삽입 된 원소 확인
endIndex -= 1 # 마지막 원소 삭제
```

스택 (Stack)

- 구현 방법도 간단함
  - 배열 (또는 리스트)의 끝자리를 확인, 끝자리에 삽입, 끝자리 원소를 삭제가 가능하기만 하면 됨.
- 예시 (C++)

```
#include<stack>
stack<int> stk; // 스택 선언
stk.push(1); // 마지막에 삽입
stk.top(); // 마지막에 삽입 된 원소 확인
stk.pop(); // 마지막 원소 삭제
stk.size(); // 스택의 크기
```

스택

- 기본 문제 : https://www.acmicpc.net/problem/10773
  - 예시 답(파이썬) : http://boj.kr/c90f75446f554f4d8344099a9338b0d4
  - 예시 답(C++):

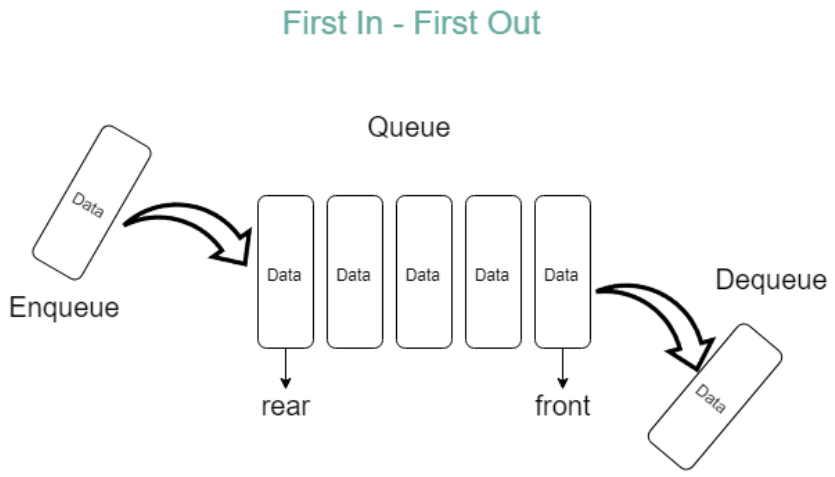
덱 (Deque)

- Double-ended queue 의 줄임말 -- > 양쪽 끝에서 삽입과 삭제가 가능한 큐 (뒤에서 설명)!

- 하지만 대부분 그냥 이미 구현 되어 있는 deque 씀
- 파이썬 : from collections import deque
  - 선언 : arr = deque([])
  - 앞에서 삭제 : arr.popleft()
  - 뒤에서 삭제 : arr.pop()
  - 앞에서 삽입 : arr.appendleft()
  - 뒤에서 삽입 : arr.append()
- C++ : #include<deque>
  - 선언 : deque<int> deq;
  - 앞에서 삭제 : deq.pop_front();
  - 뒤에서 삭제 : deq.pop_back(1);
  - 앞에 있는 원소 보기: deq.front();
  - 뒤에 있는 원소 보기 : deq.back();
  - 앞에서 삽입 : deq.push_front();
  - 뒤에서 삽입 : deq.push_back(1);

큐 (Queue)

- 선입 선출 : 먼저 들어온 자료형이 먼저 나간다
- 사람들이 줄 서는 것을 생각하면 됨

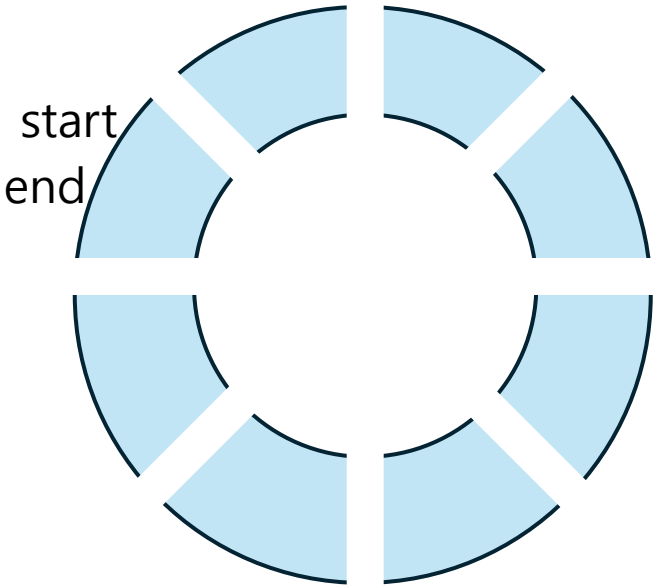


큐 (Queue)

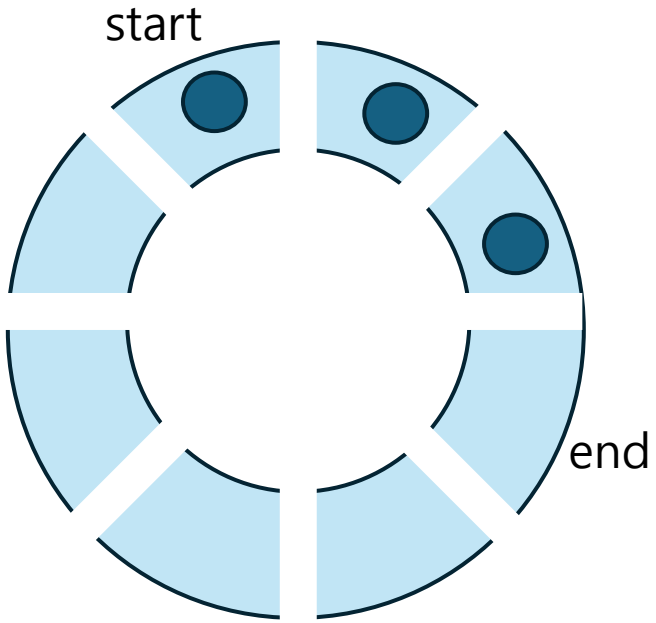
- 구현은 스택보다는 약간 어려움
- 보편적인 방법 – 원형 큐
  - 충분한 크기의 배열 선언
  - 입구 인덱스, 출구 인덱스 저장
  - 두 인덱스를 업데이트 하면서 큐 유지
- 덱을 활용하여 큐 구현
  - 덱의 앞 부분을 삭제로만, 덱의 뒷부분을 삽입으로만 사용하면 큐랑 다를 것이 없다.
- C++ : #include<queue>
  - 선언 : queue<int> q;
  - 앞에서 삭제 : q.pop();
  - 앞에 있는 원소 보기 : q.front();
  - 뒤에서 삽입 : q.push(1);

큐 (Queue)

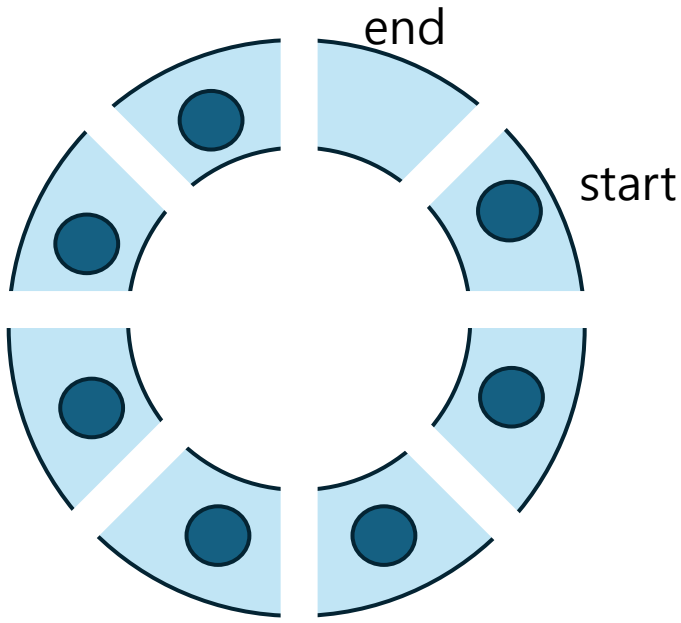
- 원형 큐



비어있는 상태



원소 3개가 들어가있는 상태



다 채운 상태

큐 (Queue)

- 구현 예제 문제: https://www.acmicpc.net/problem/10845
- 파이썬:
  - 원형 큐 : http://boj.kr/aa29261037d24999b22896ace6f3346b
  - 덱을 활용한 경우 : http://boj.kr/c59c9ea069ef41e092654eda80321629
  - 참고 : 시간 초과가 날 경우 Python3 대신 PyPy3로 제출
- C++:

2차원 배열

- 파이썬에서 N * M 배열을 선언하고 싶으면
  - arr = [[0] * M for _ in range(N)]
  - 크기가 M인 리스트를 N개 가지고 있는 리스트 선언
  - 만일 1행 3열에 있는 원소를 가져오고 싶다?
    - arr[1][3] → 1번째 리스트의 3번째 원소
- C++에서 N * M 배열을 선언하고 싶으면
  - arr = [N][M]; // 전역에서 선언 (지역변수로 선언하고 싶으면 동적할당이나 벡터 사용)
    - arr[1][3] → 1번째 리스트의 3번째 원소

2차원 배열

- 연습문제 : https://www.acmicpc.net/problem/2563
  - 예시 답(파이썬) : http://boj.kr/e14fa5066de44e6bbe4e86da69571b8c

복습 문제 모음

- 스택, 큐, 덱 : https://www.acmicpc.net/step/11
- 2차원 배열 : https://www.acmicpc.net/step/2