● Queue vs priority queue를 비교하여 설명해라

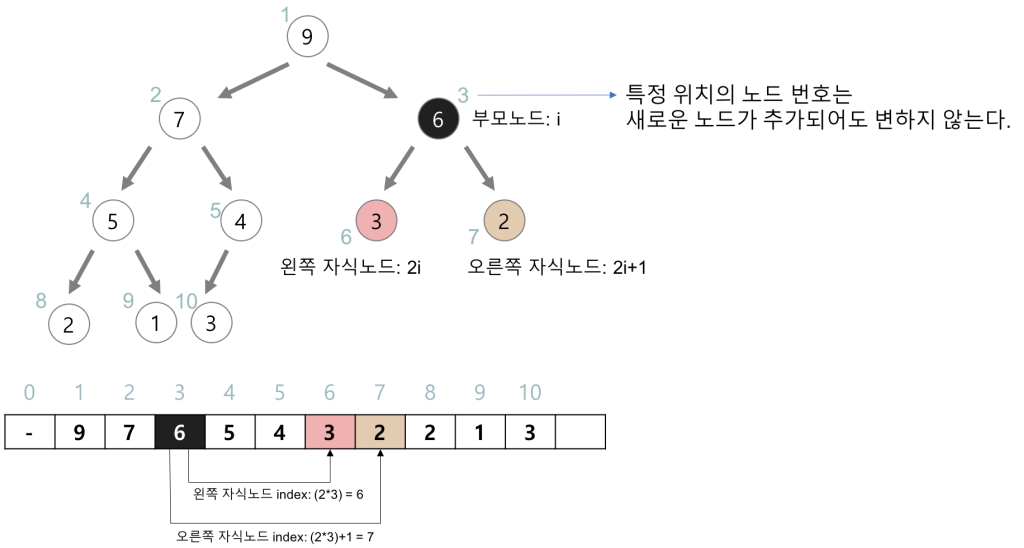
▶ Queue 자료구조는 시간 순서상 먼저 집어 넣은 데이터가 먼저 나오는 **선입선출 FIFO(First In First Out)** 구조로 저장하는 형식입니다. 이와 다르게 우선순위큐(priority queue)는, 들어간 순서에 상관없이 우선순위가 높은 데이터가 먼저 나옵니다. Queue의 operation 시간복잡도는 enqueue, dequeue 모두 O(1)이고,
Priority queue는 push, pop 모두 O(logn) 입니다.

● priority queue (우선순위큐) 구현

▶ 완전이진트리 구조를 가지고 있는 Heap은 그 자체로 우선순위 큐와 구현이 일치하기 때문에, 우선순위 큐는 Heap으로 구현할 수 있습니다. 트리는 보통 Linked-List로 구현해야 하지만 Heap은 트리임에도 불구하고 array를 기반으로 구현해야 합니다. 그 이유는 새로운 node, 새로운 데이터가 들어올 때 힙의 마지막 위치에 추가해야 하는데, 이때 array 기반으로 구현해야 이 과정이 수월해지기 때문입니다. 구현의 편의를 위해 array의 0번째 index는 사용하지 않고, 완전이진트리의 특성을 활용하여 배열의 index만으로 부모자식간의 관계를 정의합니다 (루트 노드의 인덱스 번호는 1입니다). n번째 노드의 왼쪽 자식 노드는 2n이고, n번째 노드의 오른쪽 자식 노드는 2n+1이고, n번째 노드의 부모 노드는 n/2입니다. 이렇게 Linked-List없이 배열 하나만으로도 Heap을 구현할 수 있습니다.

▶ Heap의 종류로는 최대 힙 max heap과 최소 힙 min heap이 있는데, 최소 힙은 각 node에 저장된 값이 child node들에 저장된 값보다 작거나 같고, root node에 저장된 값이 가장 작은 값이 됩니다. 반대로 최대 힙은 각 노드에 저장된 값이 자식 노드들에 저장된 값보다 크거나 같고, 루트 노드에 저장된 값이 가장 큰 값이 됩니다.

▶ 완전이진트리이기 때문에 반드시 노드가 왼쪽에서 오른쪽으로 채워져야 하고, 마지막 레벨을 제외하고는 모든 레벨이 완전히 채워져 있어야 합니다. max heap의 경우 가장 큰 루트 노드를 시작으로 자식 노드가 왼쪽에서 오른쪽으로 채워지지고, min heap의 경우 가장 작은 노드를 시작으로 자식 노드가 왼쪽에서 오른쪽으로 채워집니다.



Max Heap: https://gmlwjd9405.github.io/2018/05/10/data-structure-heap.html

● Heap 시간복잡도

▶ heap tree의 높이는 logN입니다. 그래서 push()를 했을 때, swap하는 과정이 최대 logN번 반복되기 때문에, 시간 복잡도는 O(logn)입니다.

▶ pop()을 했을 때도, swap하는 과정이 최대 logN번 반복되기 때문에 시간 복잡도는 O(logn)입니다.

● 노드 삽입과 삭제

▶ 최소힙에 노드를 삽입해 보겠습니다. 우선 완전트리의 맨 끝에, 즉, 힙의 맨 마지막 인덱스에 노드를 추가한 다음, 자신의 부모 노드와 비교해서 자기가 값이 작으면, 부모 노드와 자리를 바꿉니다. 그리고 또 부모 노드와 비교해서,

부모 노드의 값이 자기의 값보다 작을 때까지 반복하거나, 노드가 root에 도착할 때까지 반복합니다. heap tree의 높이는 logN입니다. push()를 했을 때, swap하는 과정이 최대 logN번 반복되기 때문에, 시간 복잡도는 O(logn)입니다.

▶ 최소힙에서 노드를 삭제해 보겠습니다. 최소힙에 노드를 요청할 때는 가장 작은 값을 요청하기 때문에, 루트 노드가 삭제됩니다. 그 이후 비어진 루트의 자리를 채우기 위해, 먼저 이진완전트리의 맨 마지막 노드, 즉 힙의 맨 마지막 인덱스에 있는 노드를 가져와서 루트에 채웁니다. 그런 다음 자신의 자식 노드들과 비교하여 자신보다 작은 노드, 둘 다 작다면 그 중 더 작은 노드와 자리를 바꿉니다. 그런 다음 또 자식 노드들과 비교해서 자신이 더 크다면 자리를 바꾸는 식으로 내려가다가, 자식 노드 둘 다 자신보다 크거나, 아니면 리프 노드(더이상 자식이 없는 노드 leaf node)에 도달하면 멈춥니다. pop()을 했을 때, swap하는 과정이 최대 logN번 반복되기 때문에 시간 복잡도는 O(logn)입니다.

Resources: inflearn 개발남 노씨, gmlwjd9405