자료구조 heap

ㆍ우선순위 큐 : 우선순위의 개념을 큐에 도입한 자료구조

- 데이터들이 우선순위를 가지고 있고 우선순위가 높은 데이터 먼저 나간다.

|  |  |
| --- | --- |
| 자료구조 | 삭제되는 요소 |
| 스택(Stack) | LIFO |
| 큐(Queue) | FIFO |
| 우선순위 큐(priority Queue) | 가장 우선순위가 높은 데이터 |

- 우선순위 큐 이용 사례

1. 시뮬레이션 시스템

2. 네트워크 트래픽 제어

3. 운영체제에서의 작업 스케쥴링

4. 수치 해석적인 계산

- 우선순위 큐는 배열 , 연결리스트, 힙으로 구성 가능하다 (heap 이 가장 효율적)

|  |  |  |
| --- | --- | --- |
| 우선순위 큐를 구현하는 표현 방법 | 삽입 | 삭제 |
| 순서없는 배열 | O(1) | O(n) |
| 순서없는 연결 리스트 | O(1) | O(n) |
| 정렬된 배열 | O(n) | O(1) |
| 정렬된 연결 리스트 | O(n) | O(1) |
| 힙(heap) | O(logn) | O(logn) |

ㆍ heap 이란?

- 완전 이진트리의 일종 (우선순위 큐를 위하여 사용)

- 여러개의 최대 , 최소 값을 빠르게 찾아내기 위함

- 반 정렬상태 유지

=>큰 값이 상위레벨에 있고, 작은 값이 하위 레벨에 있음 (부모 노드의 키 값이 자식 노드의 키 값보다 항상 크거나 작은 이진트리)

- heap tree에서는 중복된 값 허용

ㆍ heap 종류

- 최대 힙(max heap)

부모노드의 키 값이 자식노드의 키 값보다 크거나 같은 완전 이진트리

key(부모노드) >= key(자식 노드)

- 최소 힙(min heap)

부모노드의 키 값이 자식노드의 키 값보다 작거나 같은 완전 이진트리

key(부모노드) <= key(자식노드)

ㆍ heap에서의 부노노드와 자식노드의 관계

- 왼쪽 자신의 인덱스 = (부모의 인덱스) \* 2

- 오른쪽 자식의 인덱스 = (부모의 인덱스) \* 2 + 1

- 부모의 인덱스 = (자식의 인덱스) / 2

ㆍ heap 삭제

1. max heap에서 최대 값은 root노드 이므로 root노드가 삭제된다.

- max heap에서 삭제 연산은 최대 값을 가진 요소를 삭제하는 것

2. 삭제된 루트 노드에는 heap의 마지막 노드를 가져온다.

3. 힙 재구성

ㆍ heap 구현

1. 완전 이진트리는 배열로 구현

2. 구현을 쉽게하기 위해 배열 사용시 인덱스는 1부터 사용

3. 특정 노드 배열 인덱스가 current 라면 , 부모노드 = current / 2 , 자식 노드 = current \* 2(좌측 자식 노드) ,

자식 노드 = current \* 2 + 1 (우측 자식 노드)