Dijkstra Algorithm

2020 Winter 초급

서강대학교 강효규

Dijkstra Algorithm

✓ $Graph$ 내, 시작 정점에서 목표 정점까지의 최단경로를 구해주는 알고리즘

✓ O((V+E)logE)

✓가중치가 음수인 간선이 존재할 때는 사용 할 수 없다.

priority_queue

✓다익스트라 알고리즘에선 우선순위 큐라는 자료구조가 사용된다.

✓일반적인 큐는 들어간 순서에 상관없이 먼저 들어온 데이터가 먼저 나간다.

✓우선순위 큐는 들어간 순서에 상관없이 우리가 정한 우선순위가 높은 원소를 추출한다.

✓여러 방법으로 구현할수 있으나 일반적으로 *heap* 구조를 이용한다.

priority_queue

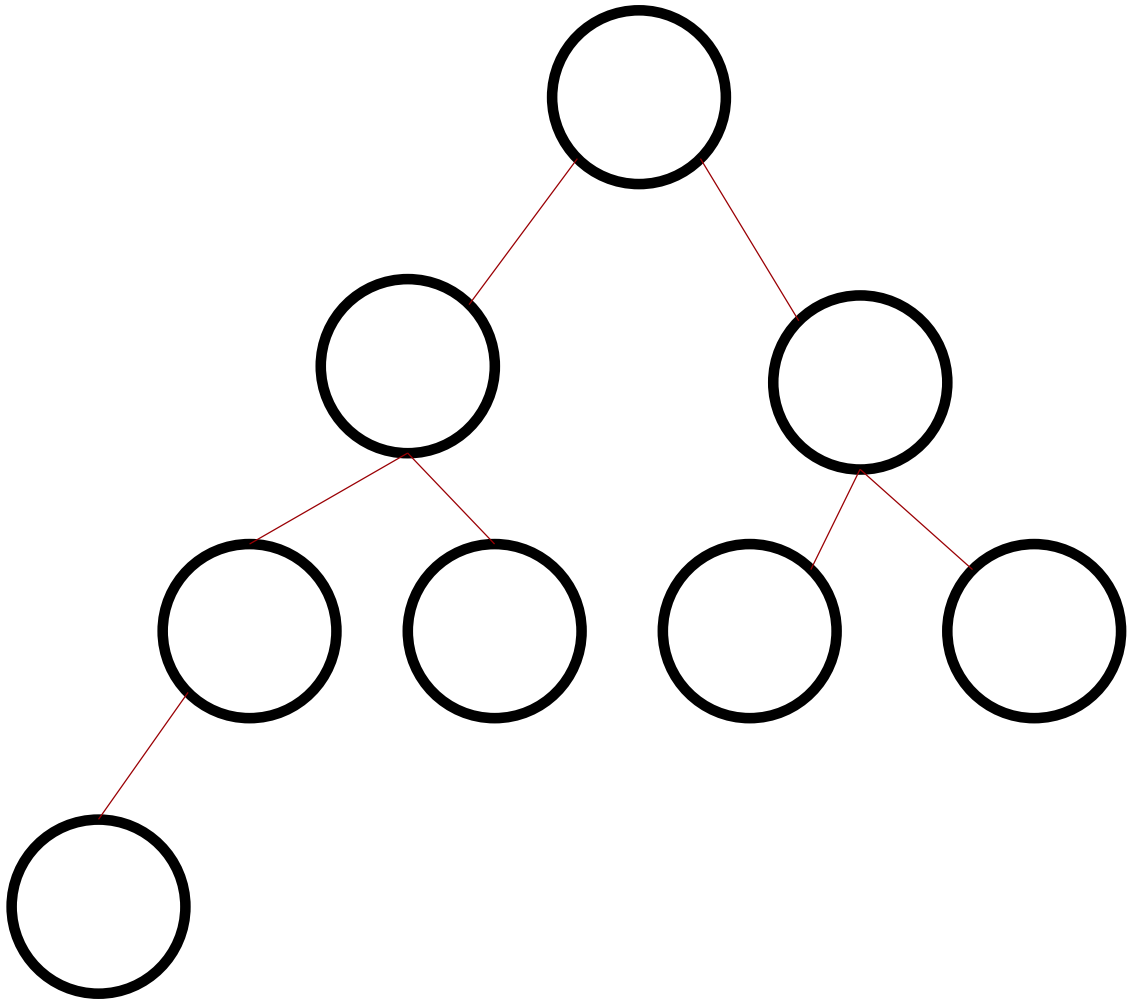
✓*heap* 이라는 자료구조를 이용하면 우선순위 큐의 삽입과 삭제를 O(logN)에 가능하다

✓*heap*은 완전 이진 트리이다.

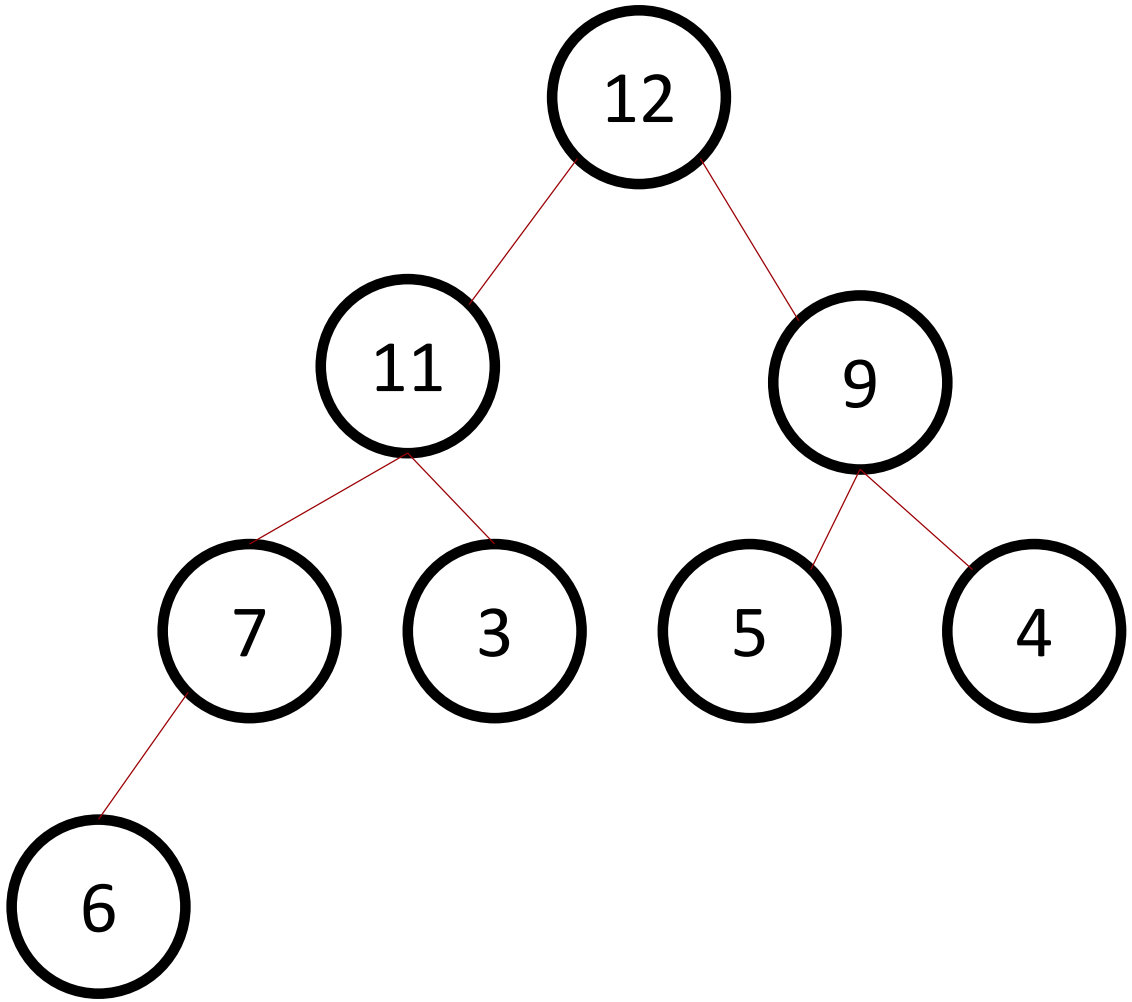
✓이진 트리란, 각각의 노드가 최대 두 개의 자식 노드를 가지는 트리 자료 구조

✓완전 이진 트리란, 이진 트리 중에서 마지막 레벨을 제외한 모든 레벨의 node가 완전히 채워져 있으며 마지막 레벨의 node들은 가능한 한 왼쪽부터 채워져 있는구조.

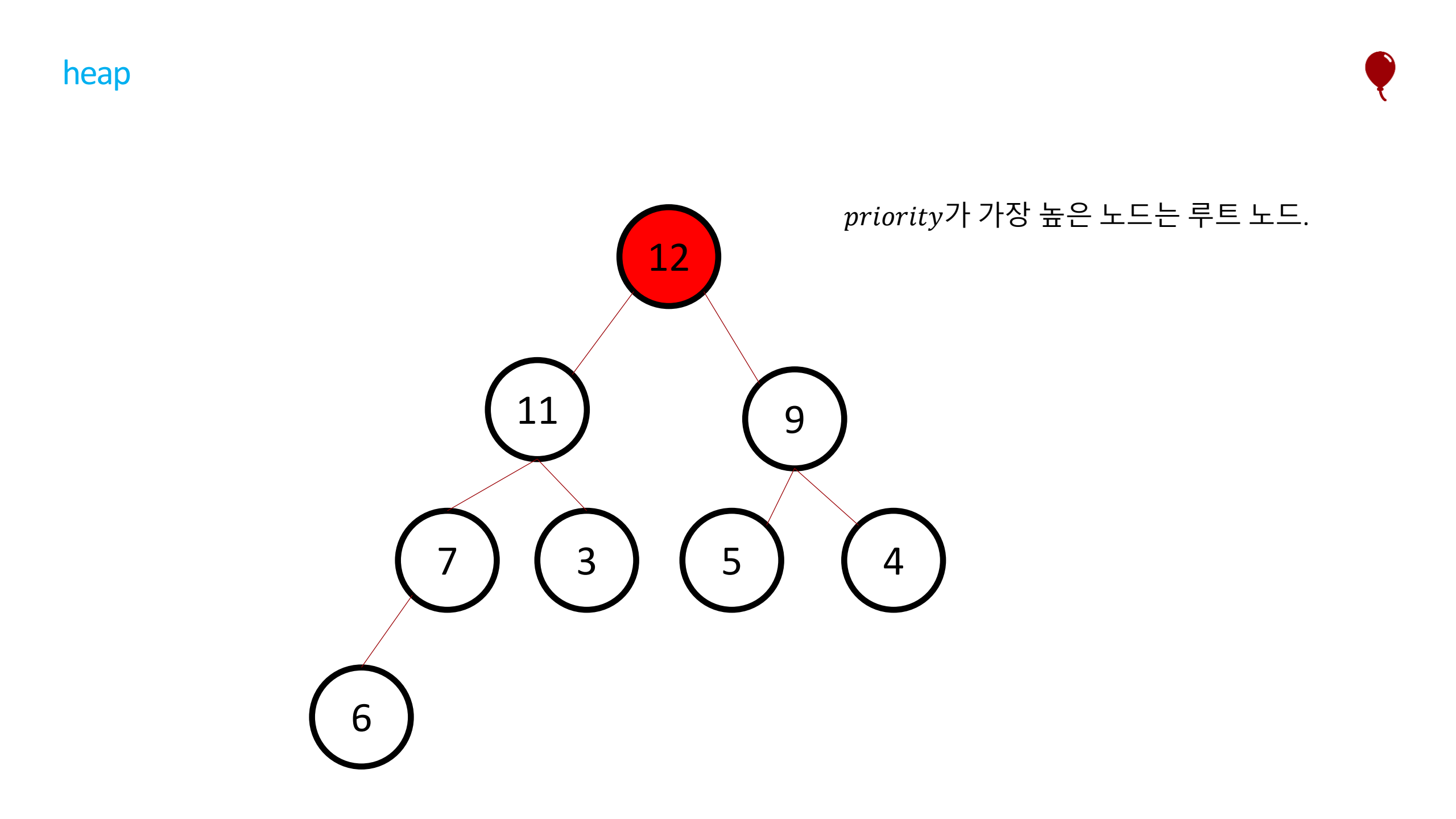
heap



heap

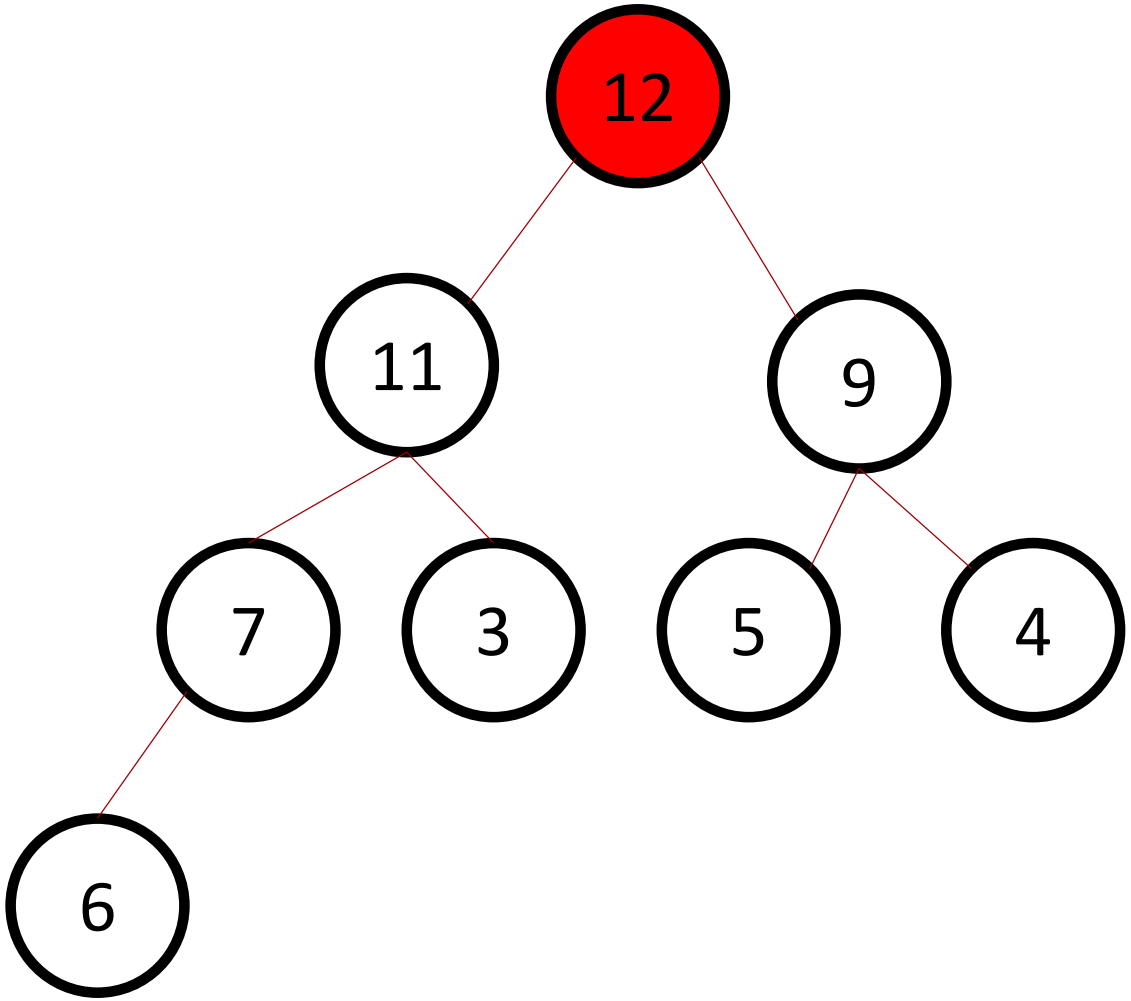


heap

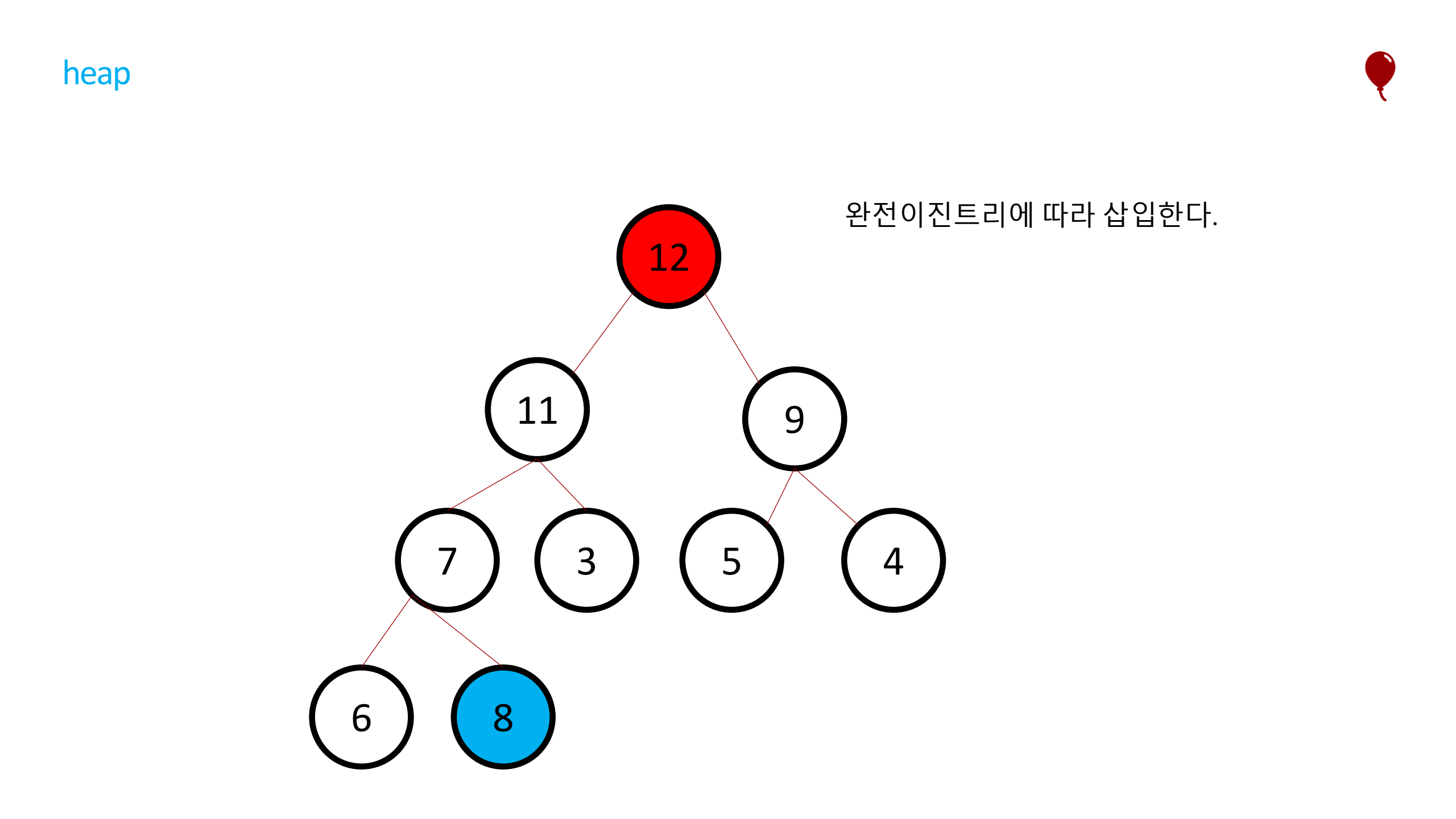


$priority$가 가장 높은 노드는 루트 노드.

heap

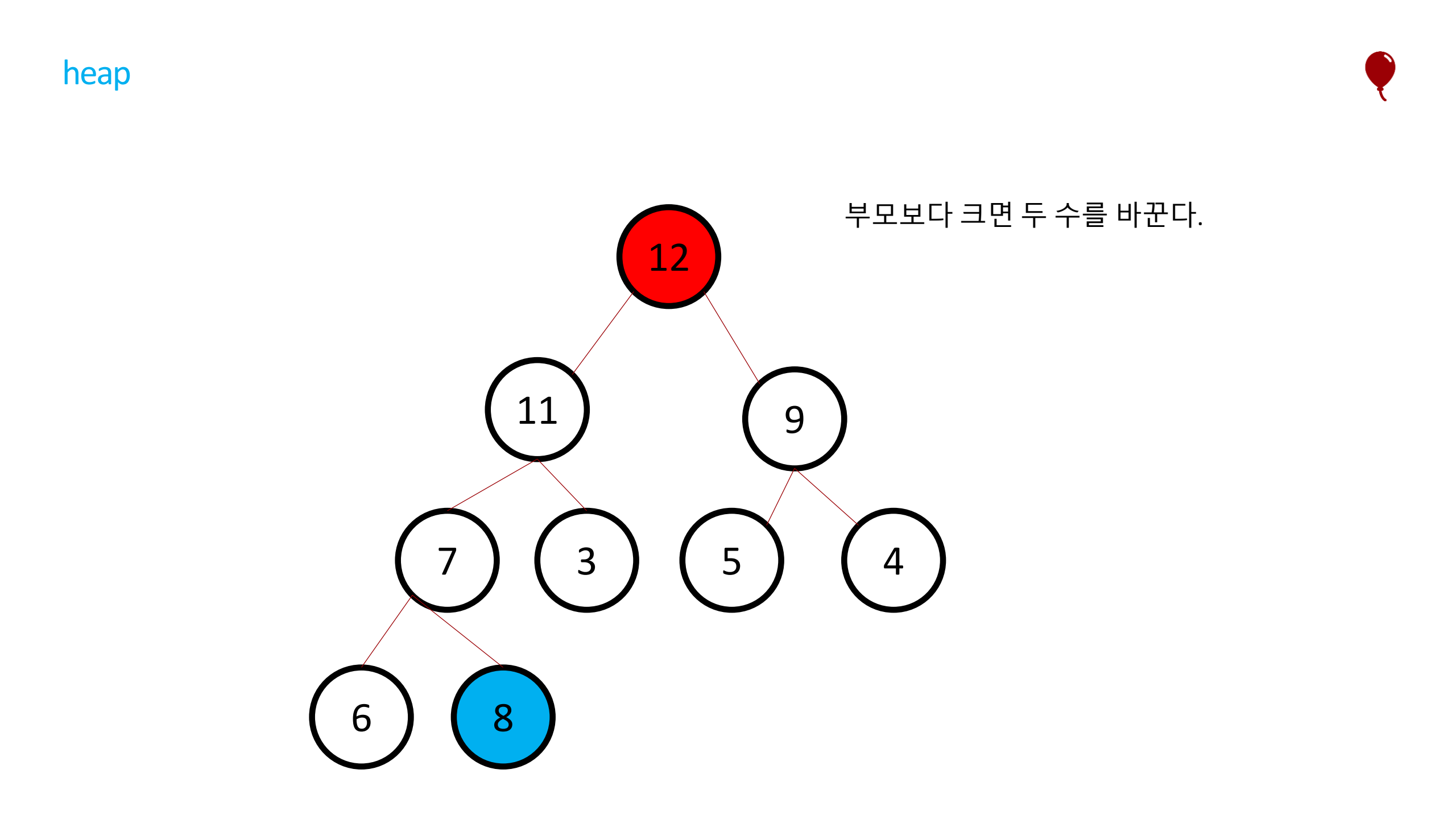


How to push value 8?

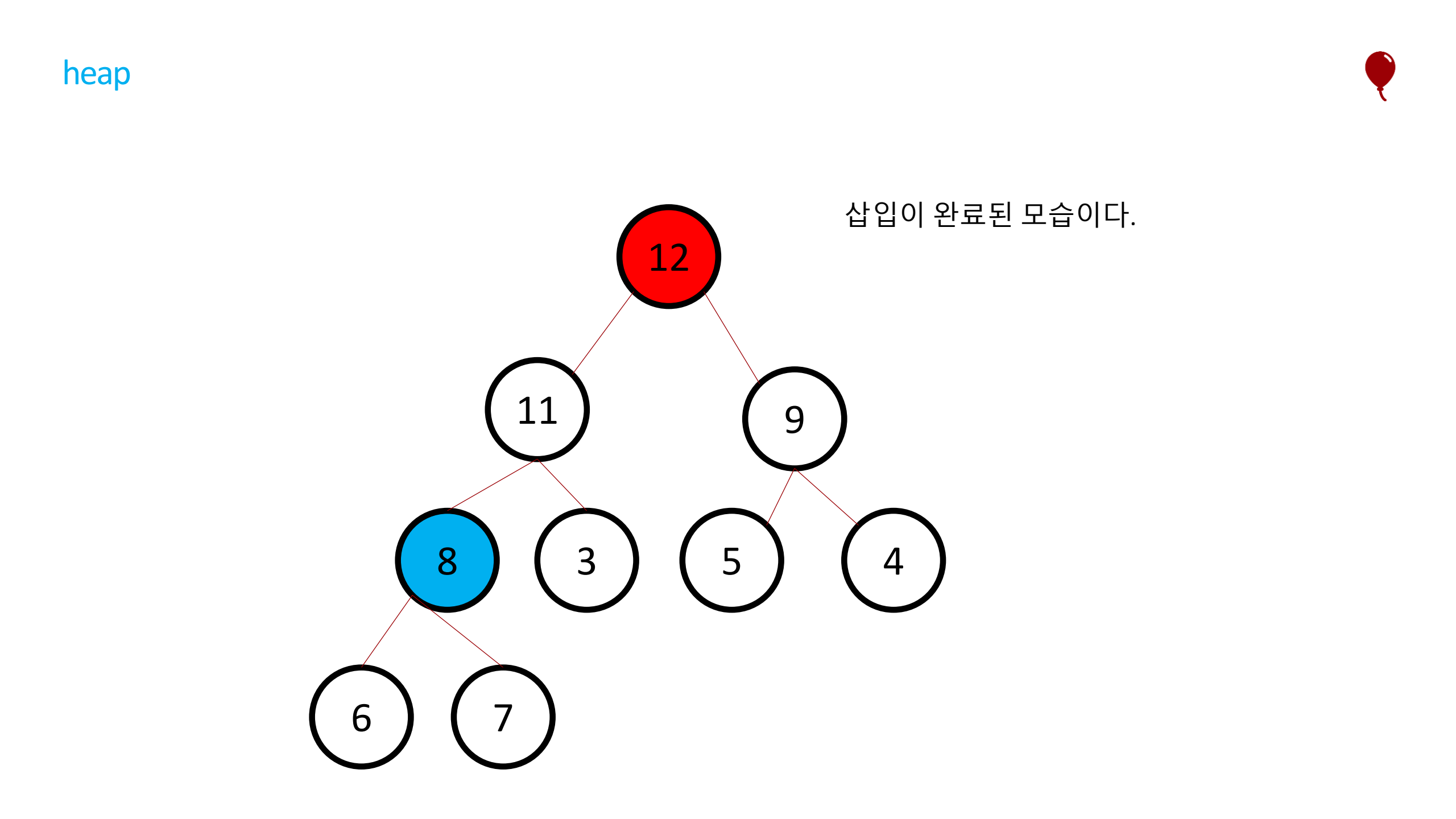


완전이진트리에 따라 삽입한다.

heap

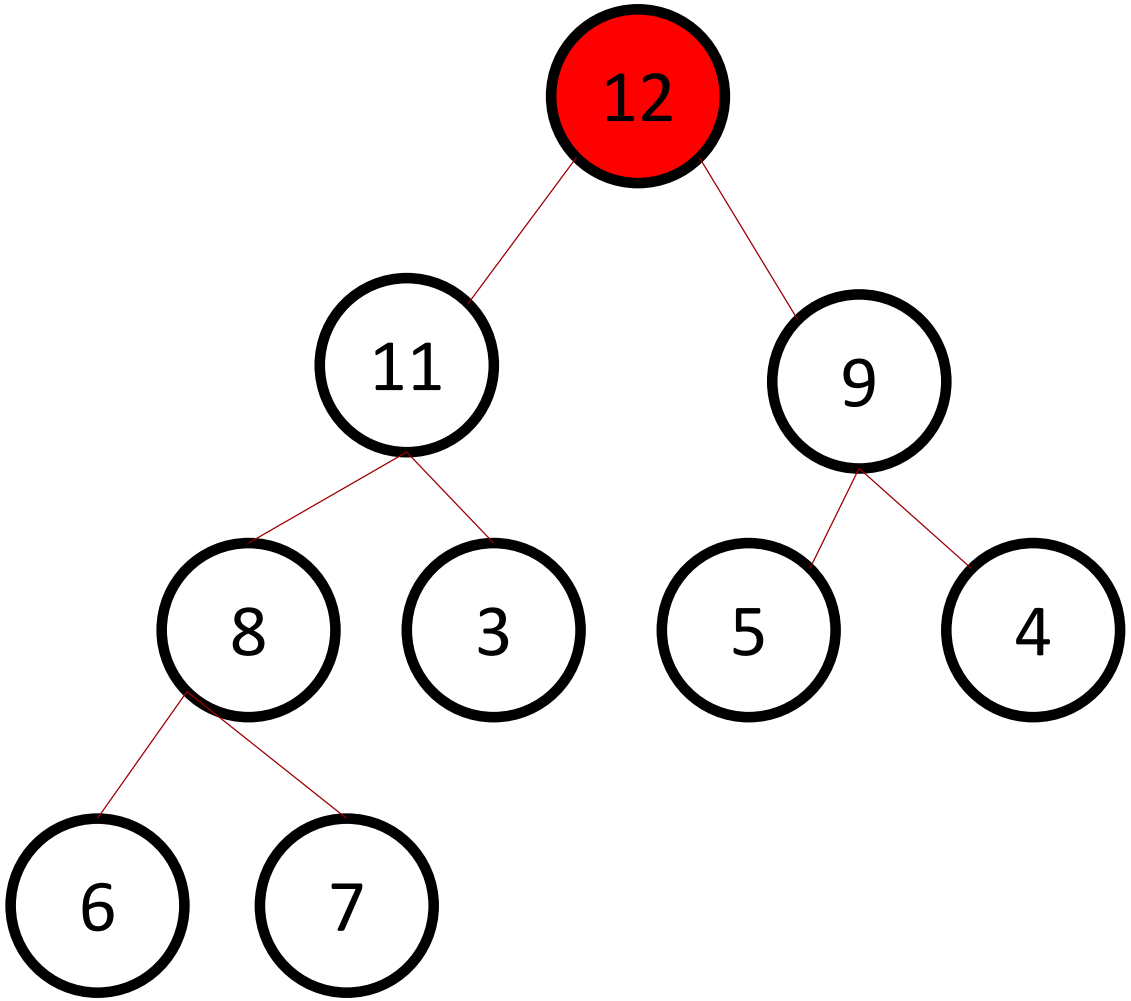


부모보다 크면 두 수를 바꾼다.



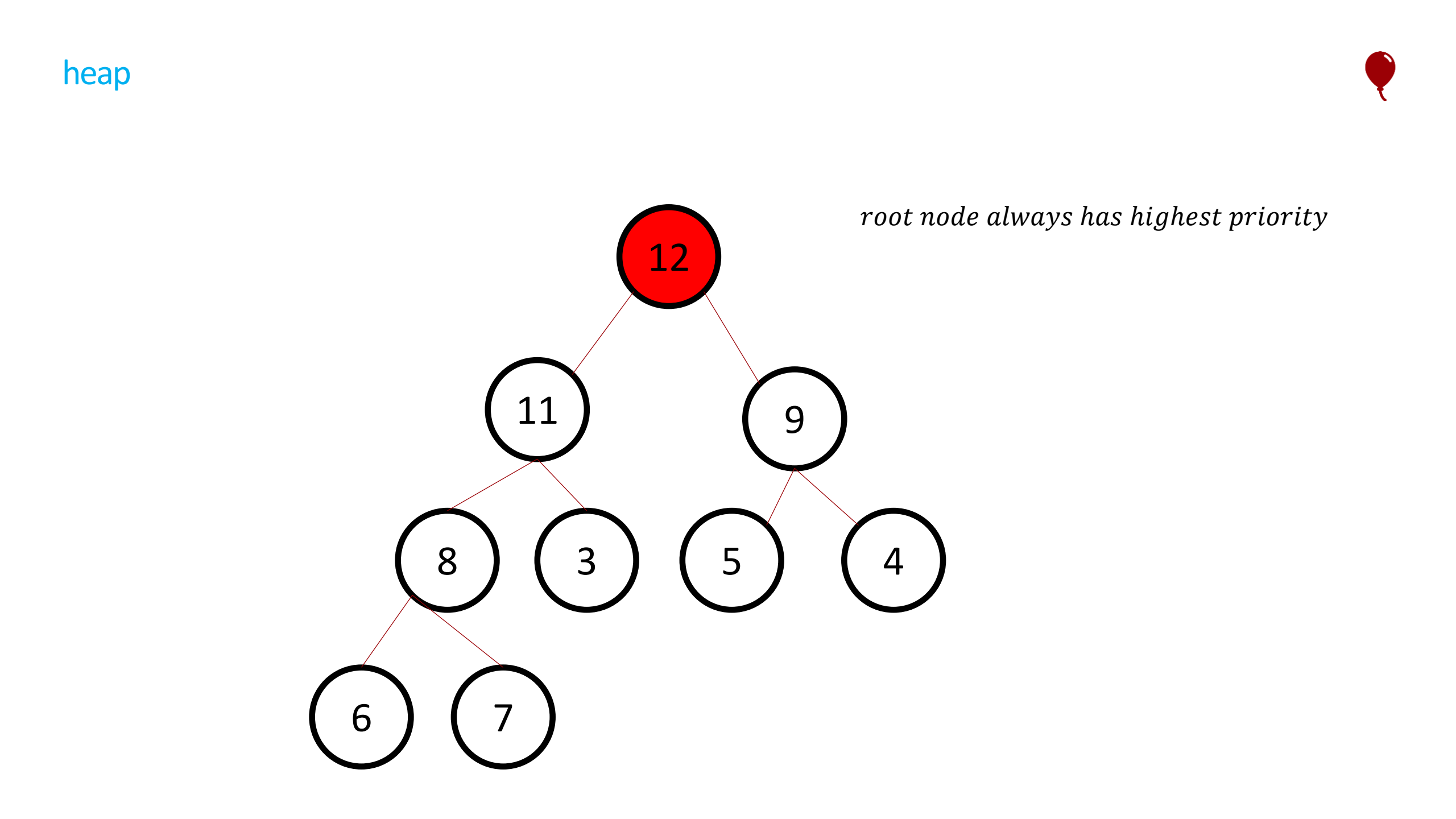
삽입이 완료된 모습이다.

heap

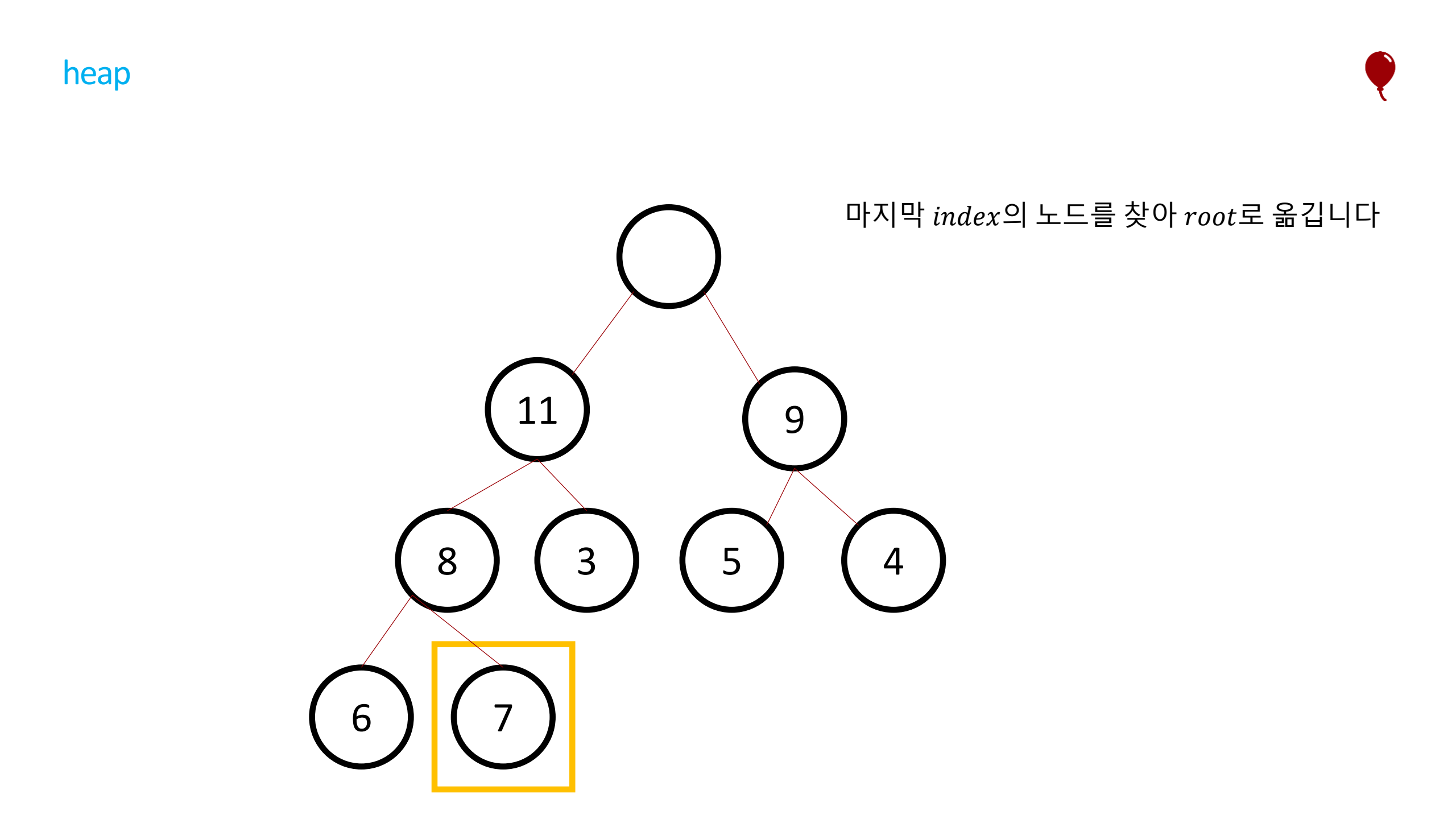


How to pop value 12?

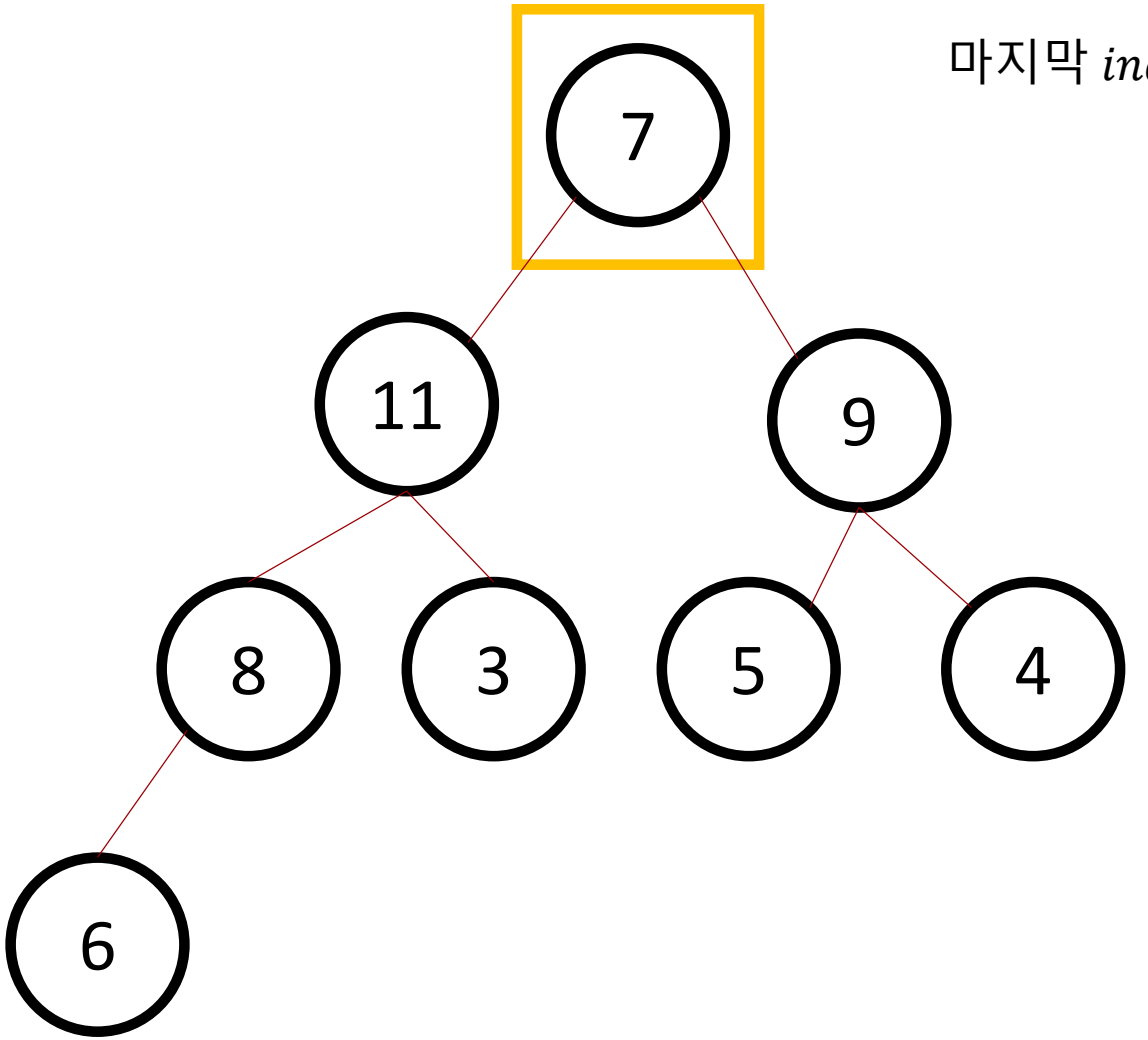
heap



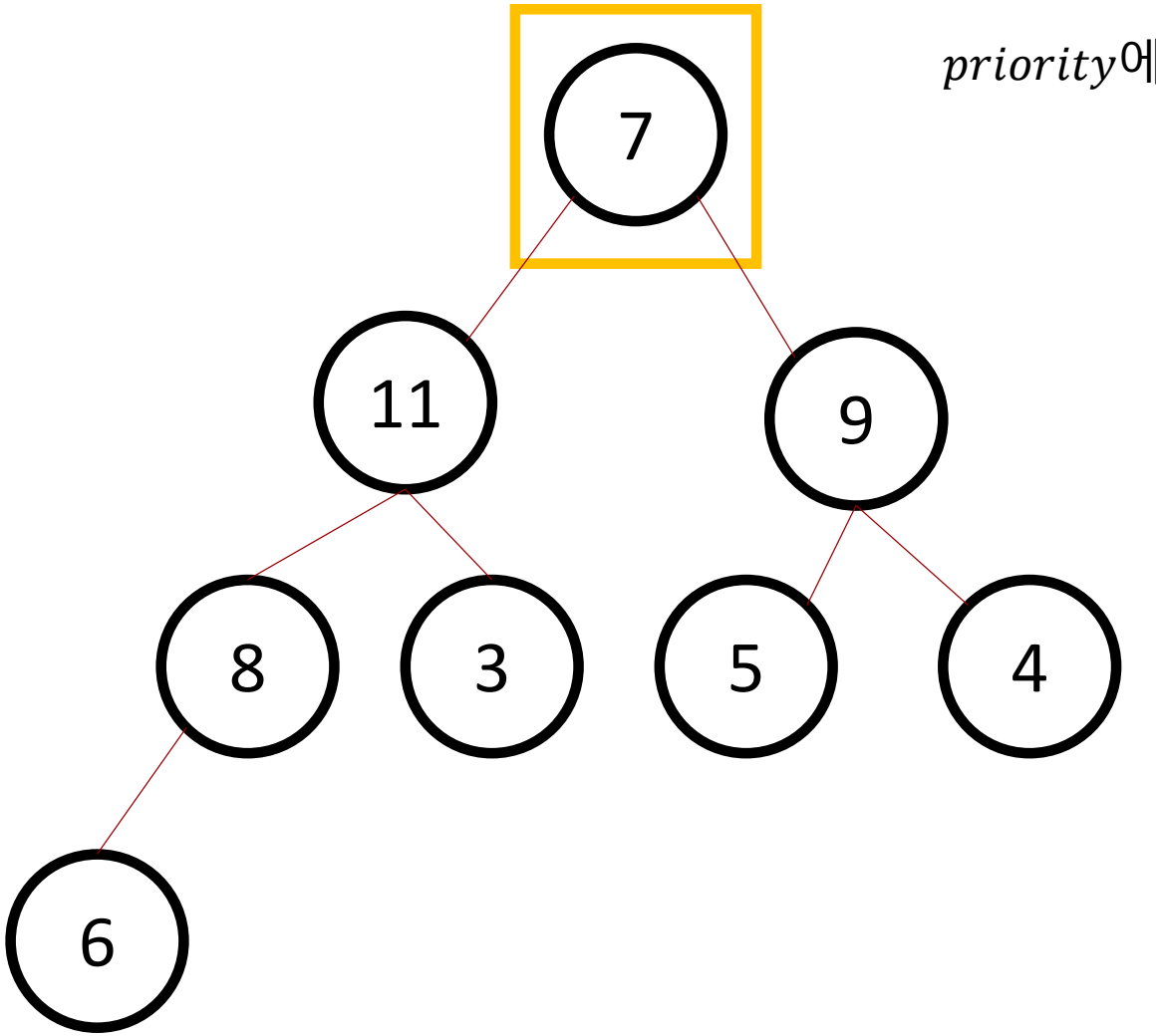
root node always has highest priority



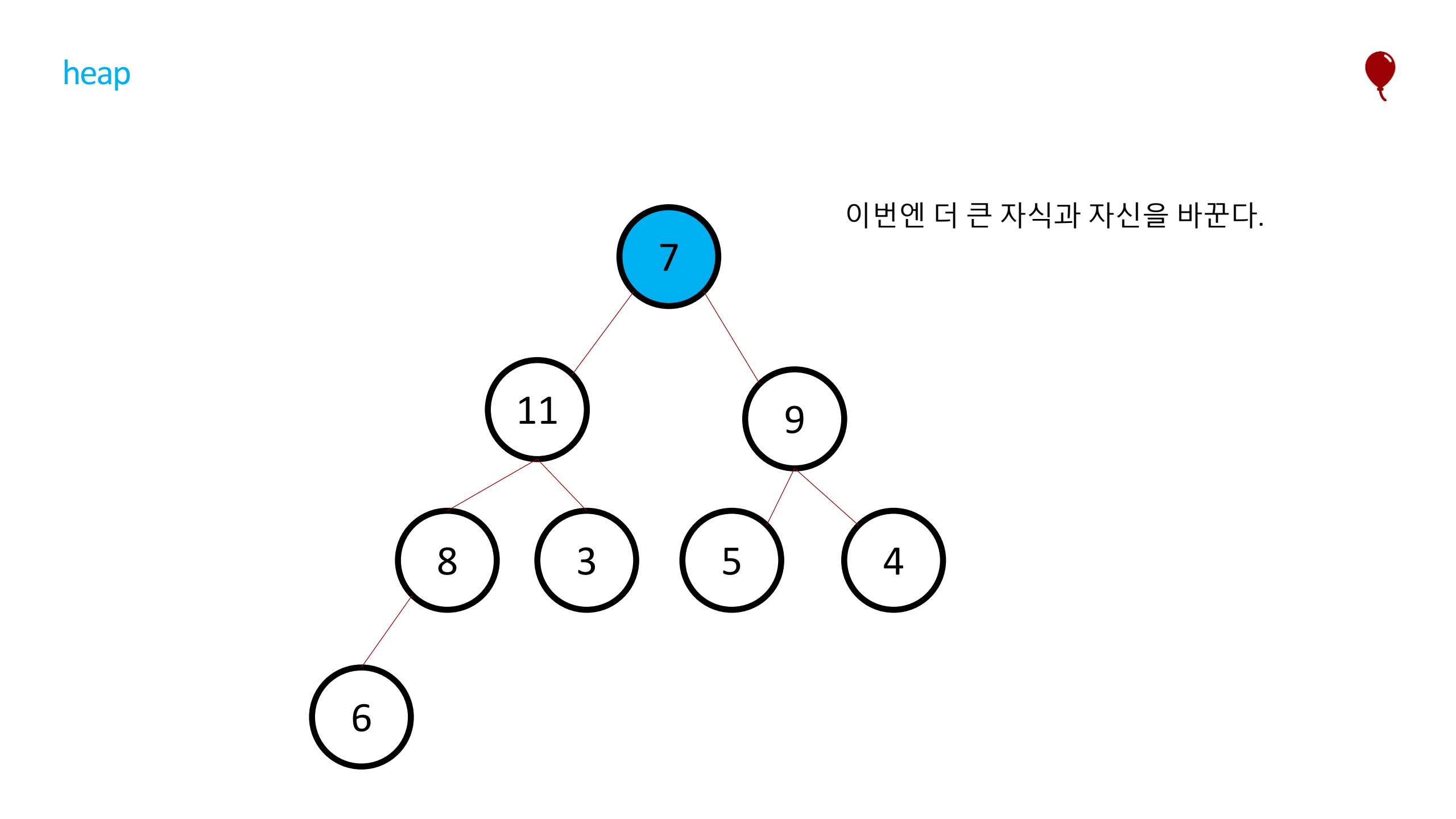
마지막 $index$의 노드를 찾아 $root$로 옮깁니다



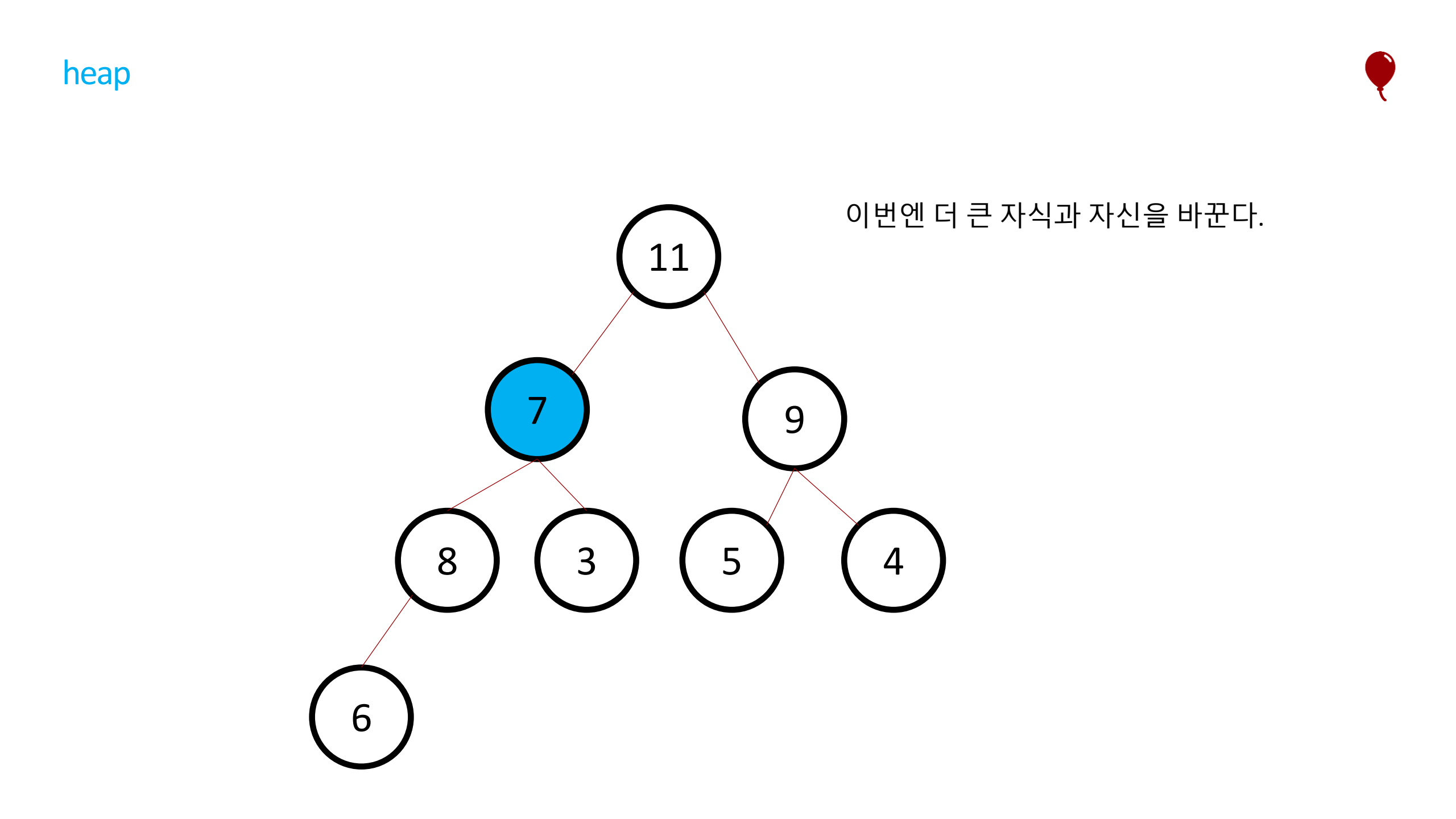
마지막 $index$의 노드를 찾아 $root$로 옮긴다.



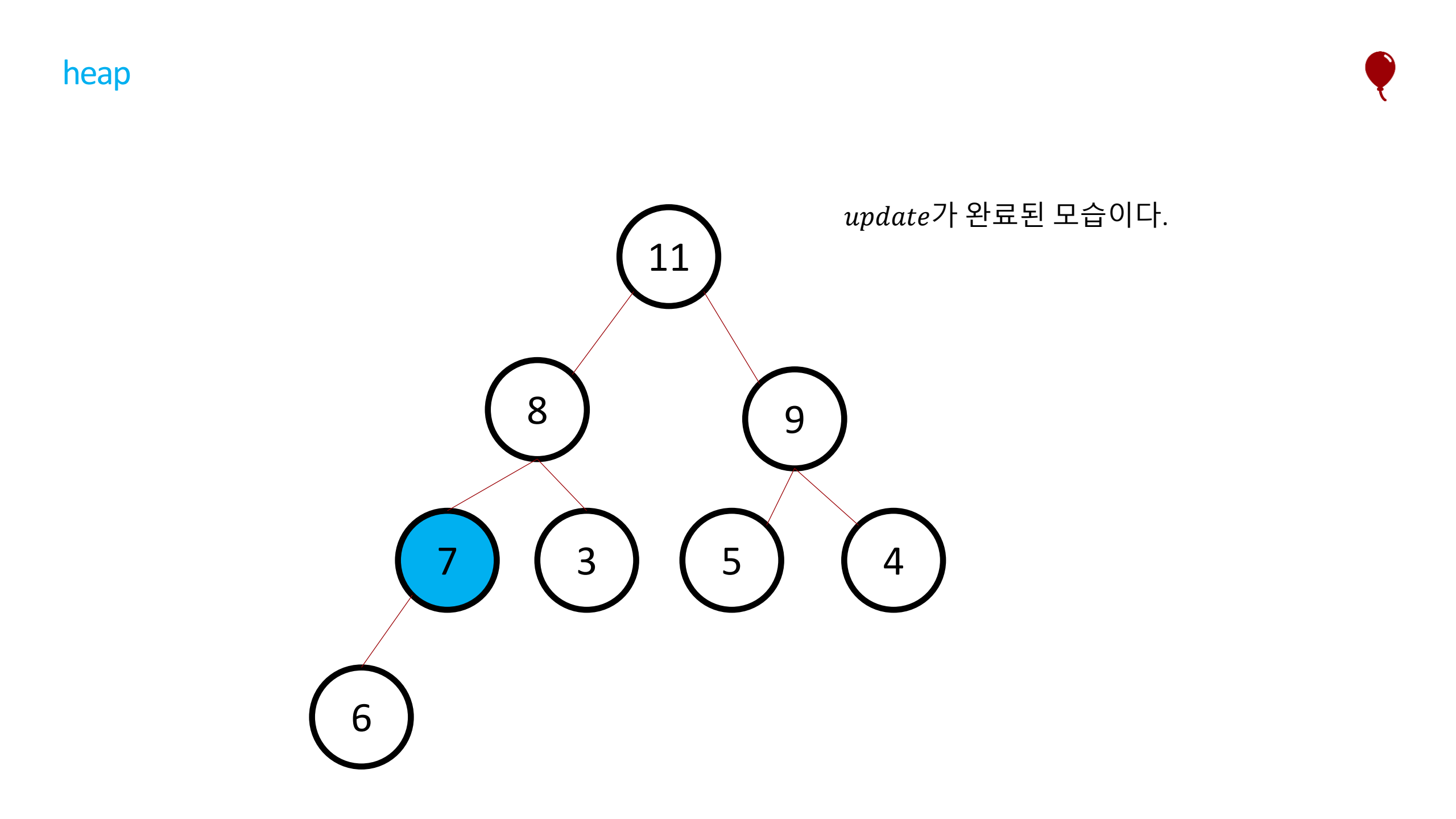
$priority$에 맞게 다시 $update$를 하자.



이번엔 더 큰 자식과 자신을 바꾼다.



이번엔 더 큰 자식과 자신을 바꾼다.



$update$가 완료된 모습이다.

완전 이진 트리이기 때문에 $O(logN)$번의 연산 안에 삽입 삭제가 가능하다.

일반적 큐와 마찬가지로 $top, size, pop, empty$등의 함수가 내장되어 있다.

우리가 원하는 기준의 $priority$ 를 가지는 pq를 만드는 것이 중요하다.

*priority*가 수의 크기에 따른 것일 때 *pq*의 구현

```
priority_queue<ll>pq1;
//기본적 pq는 큰 수 부터 나옴
priority_queue<ll, vector<ll>, greater<> >pq2;
//작은 수 부터 나오게 하는 방법
```

*greater*과 같은 내장된 비교 연산자 말고 우리가 만들어서 비교해보자!

일반적 pq의 구현

```
priority_queue<ll>pq1;
//기본적 pq는 큰 수 부터 나옴
priority_queue<ll, vector<ll>, greater<> >pq2;
//작은 수 부터 나오게 하는 방법
```

$greater$과 같은 내장된 비교 연산자 말고 우리가 만들어서 비교해보자!

pq의 구현

```
2  #include<functional>
3
4  struct cmp {
5      bool operator()(ll x, ll y){return abs(x - 3) > abs(y - 3); }
6  };
7  priority_queue<ll, vector<ll>, cmp >pq3;
8  //3과의 차이가 작은 수 부터 나오게 하는 방법
```

$functional$ 헤더는 pq를 변형할 수 있게 해준다. pq사용할때 항상 넣어주도록 하자

pq의 구현

```
2  #include<functional>
3
4  struct cmp {
5      bool operator()(ll x, ll y){return abs(x - 3) > abs(y - 3); }
6  };
7  priority_queue<ll, vector<ll>, cmp >pq3;
8  //3과의 차이가 작은 수 부터 나오게 하는 방법
```

$functional$ 헤더는 pq를 변형할 수 있게 해준다. pq사용할때 항상 넣어주도록 하자
위 코드는 3과의 차이를 $priority$로 설정한 모습이다.

pq의 구현

```
3  struct D {
4      int a, b, c;
5  };
6  struct cmpD {
7      bool operator() (D x, D y) {
8          if (x.a < y.a)return true;
9          if (x.a > y.a)return false;
10         if (x.b < y.b)return true;
11         if (x.b > y.b)return false;
12         if (x.c > y.c)return true;
13         if (x.c < y.c)return false;
14         return true;
15     }
16 };
17 //일정 한 규칙에 의해 구조체의 priority를 설정하는 방법
18 priority_queue<D, vector<D>, greater<cmpD> > pq4;
```

#11286 절댓값 힙

문제

절댓값 힙은 다음과 같은 연산을 지원하는 자료구조이다.

1. 배열에 정수 x (x ≠ 0)를 넣는다.
2. 배열에서 절댓값이 가장 작은 값을 출력하고, 그 값을 배열에서 제거한다. 절댓값이 가장 작은 값이 여러개일 때는, 가장 작은 수를 출력하고, 그 값을 배열에서 제거한다.

프로그램은 처음에 비어있는 배열에서 시작하게 된다.

입력

첫째 줄에 연산의 개수 N(1≤N≤100,000)이 주어진다. 다음 N개의 줄에는 연산에 대한 정보를 나타내는 정수 x가 주어진다. 만약 x가 0이 아니라면 배열에 x라는 값을 넣는 (추가하는) 연산이고, x가 0이라면 배열에서 절댓값이 가장 작은 값을 출력하고 그 값을 배열에서 제거하는 경우이다. 입력되는 정수는 -2^{31}보다 크고, 2^{31}보다 작다.

출력

입력에서 0이 주어진 회수만큼 답을 출력한다. 만약 배열이 비어 있는 경우인데 절댓값이 가장 작은 값을 출력하라고 한 경우에는 0을 출력하면 된다.

#11286 절댓값 힙

```cpp
struct cmp {
    bool operator()(ll x, ll y) {
        if (abs(x) == abs(y))return x > y;
        return abs(x) > abs(y);
    }
};
priority_queue<ll, vector<ll>, cmp> pq;
int main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0), cout.tie(0);
    int t, a;
    cin >> t;
    while (t--) {
        cin >> a;
        if (!a) {
            if (pq.empty())cout << '0' << '\n';
            else cout << pq.top()<<'\n', pq.pop();
        }
        else pq.push(a);
    }
}
```

Accepted

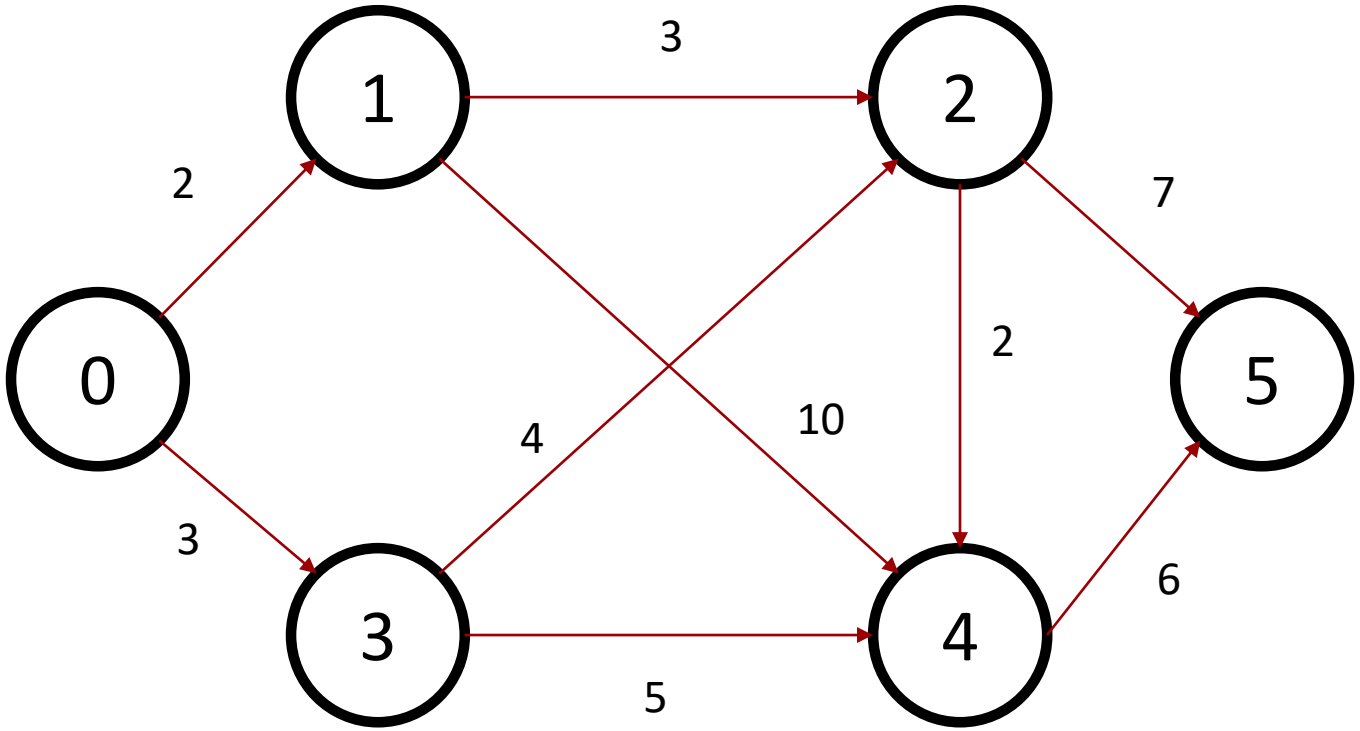
Dijkstra Algorithm

✓아직 방문하지 않은 정점들 중 시작점으로부터 거리가 가장 짧은 정점 u를 방문한다.

✓해당 정점 u와 인접하고 아직 미방문 상태인 정점들의 거리를 갱신한다.

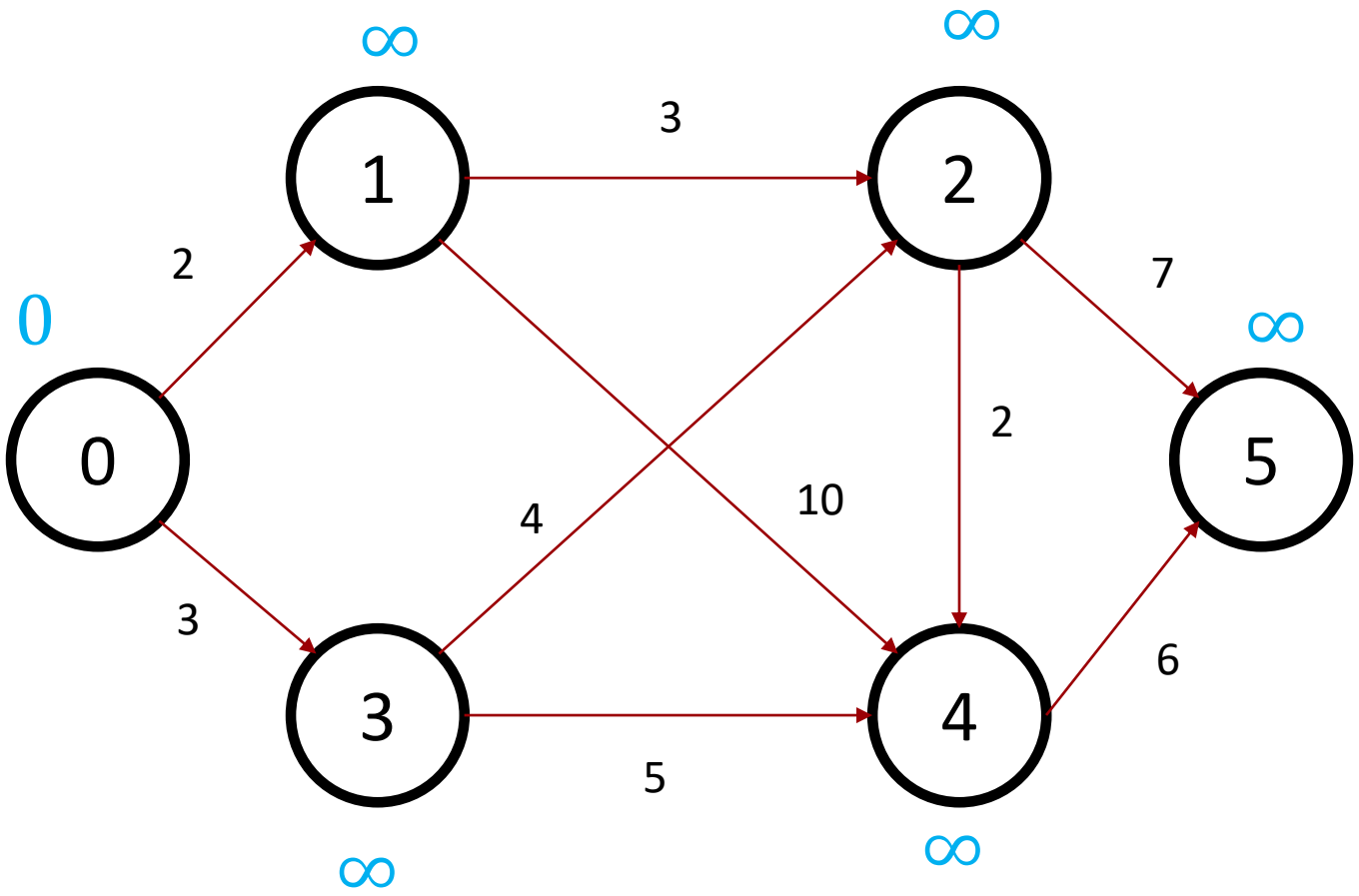
✓이를 V번 반복한다.

Dijkstra Algorithm



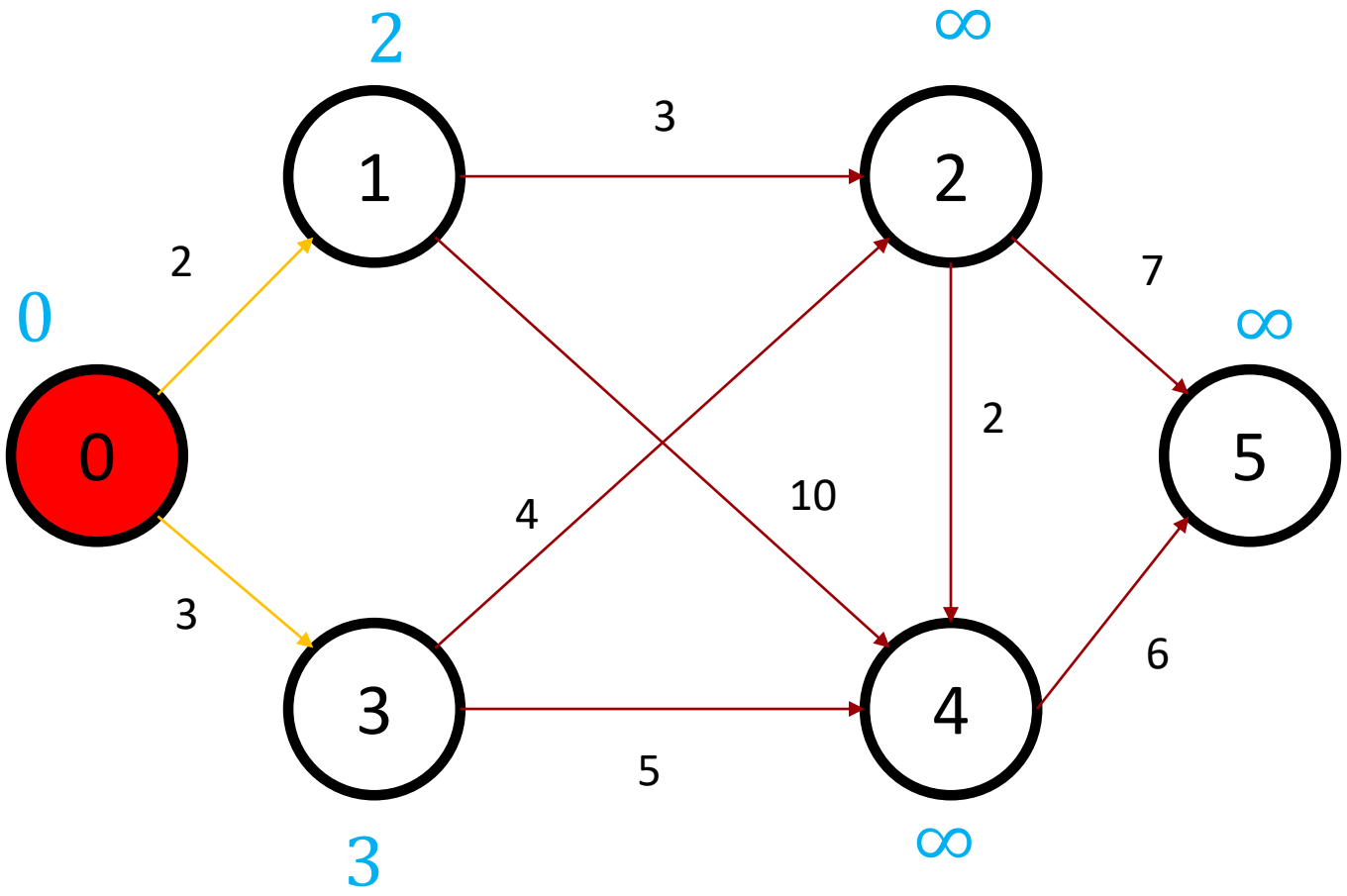
$how\ can\ we\ find\ the\ shortest\ path's\ of\ 0\ to\ 5$

Dijkstra Algorithm

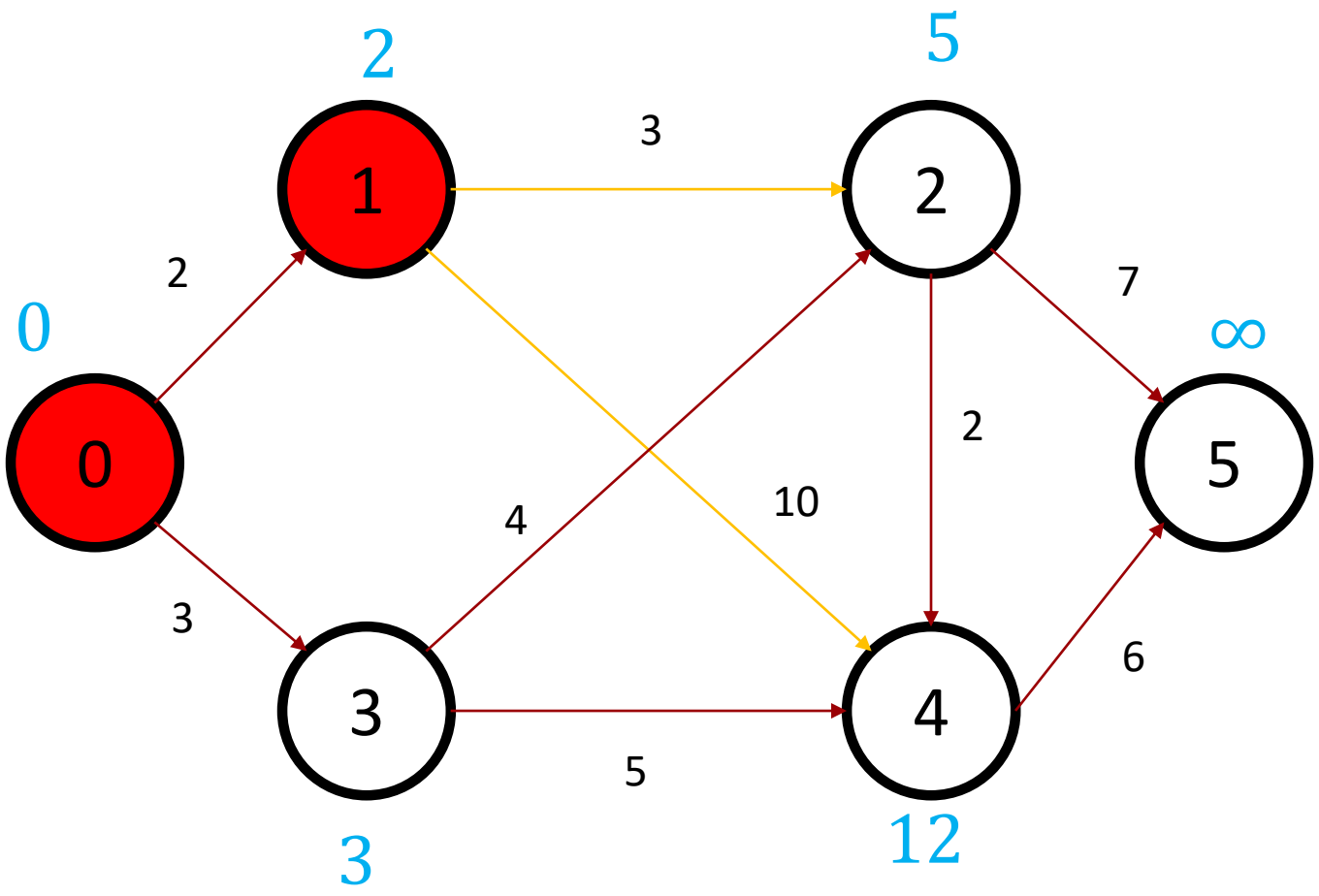


거리와 방문체크를 표시하는 배열을 만든다.

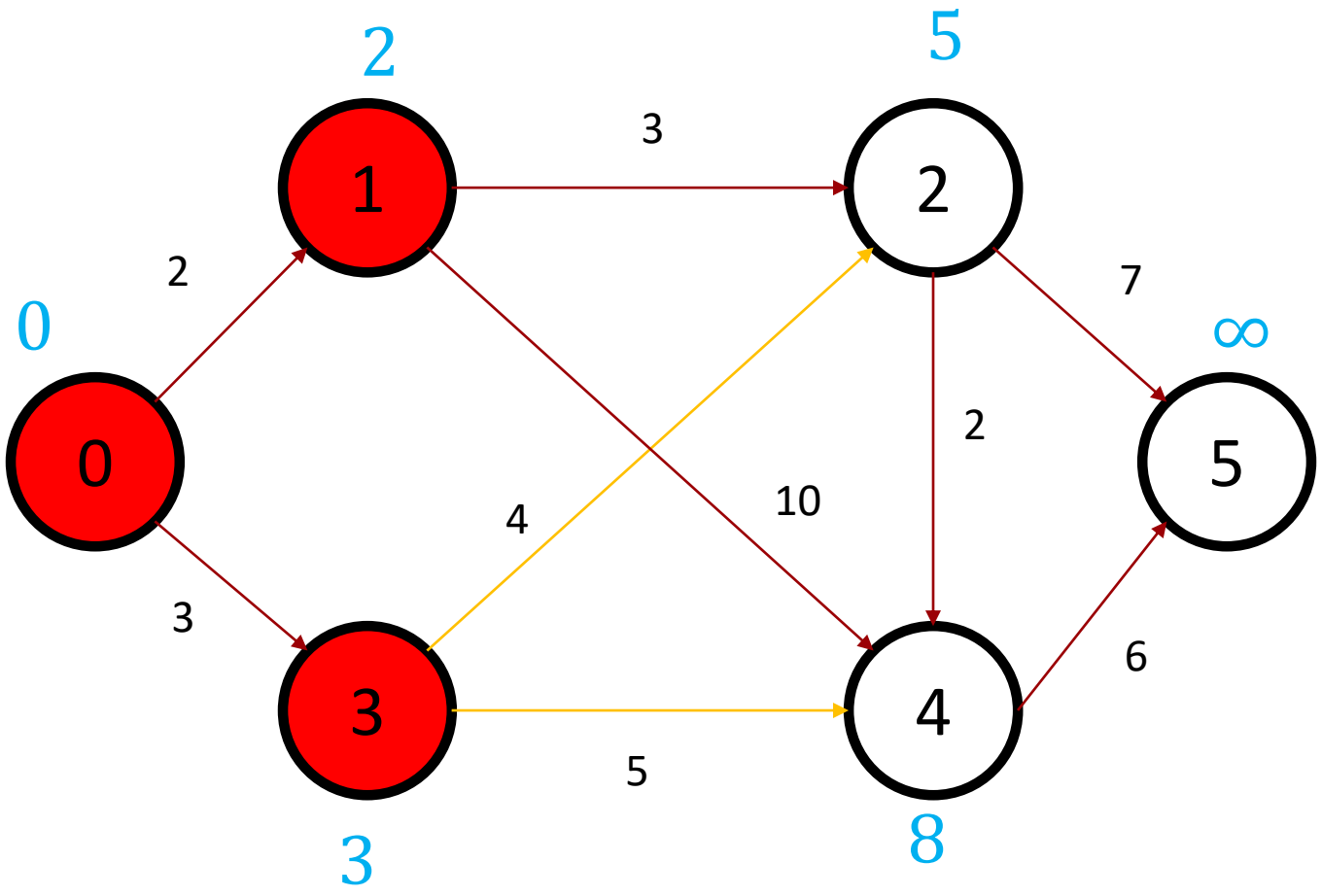
Dijkstra Algorithm



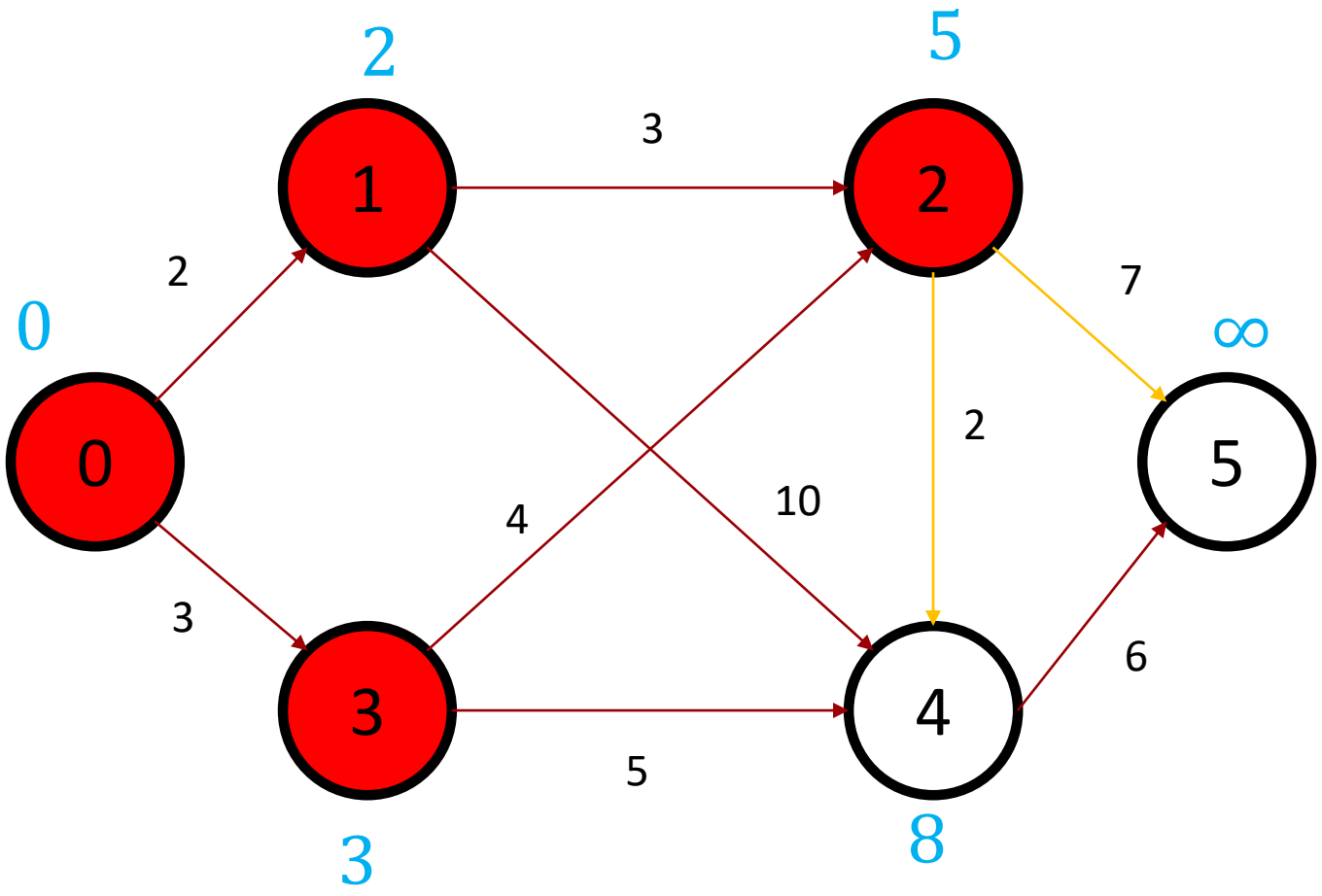
Dijkstra Algorithm



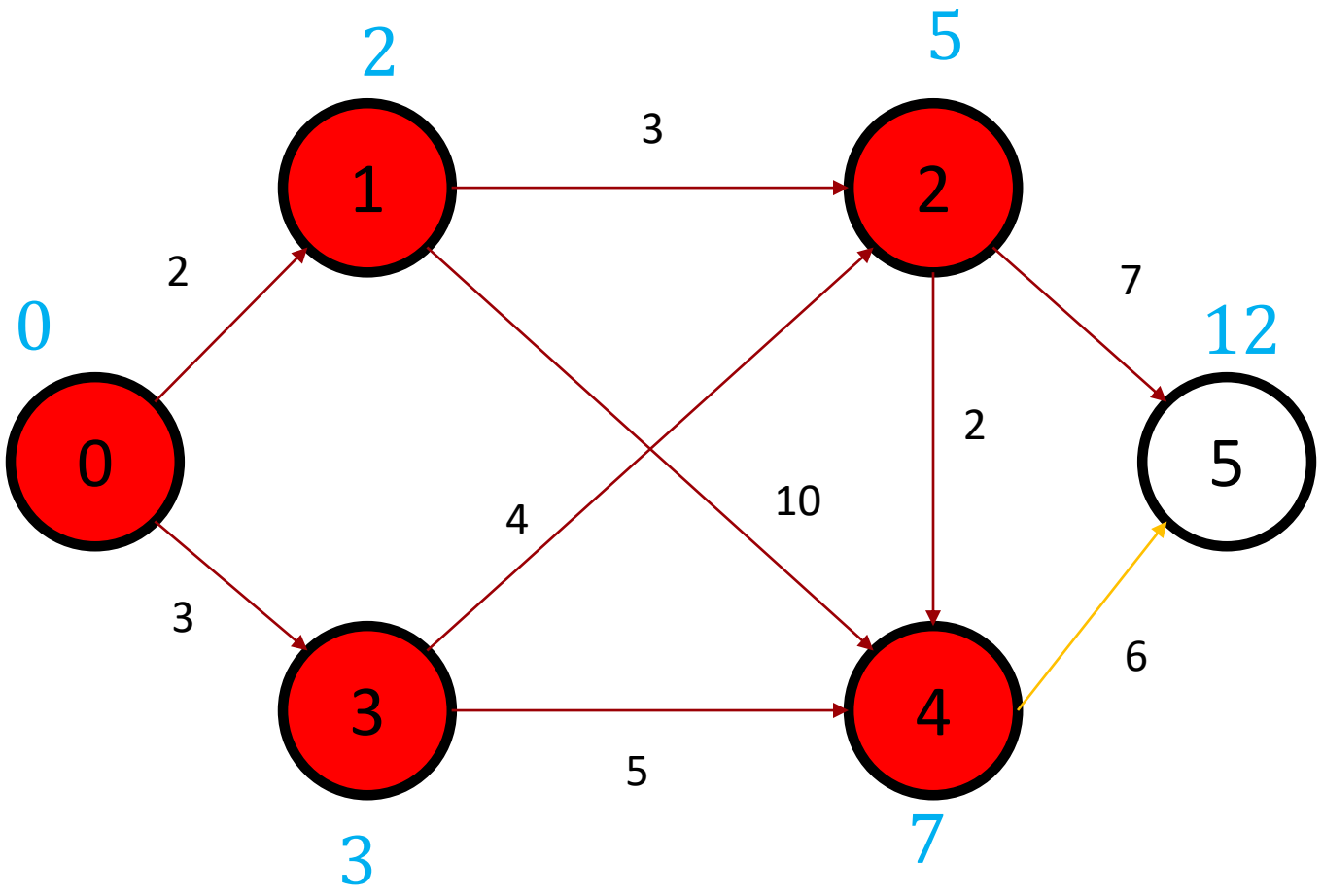
Dijkstra Algorithm



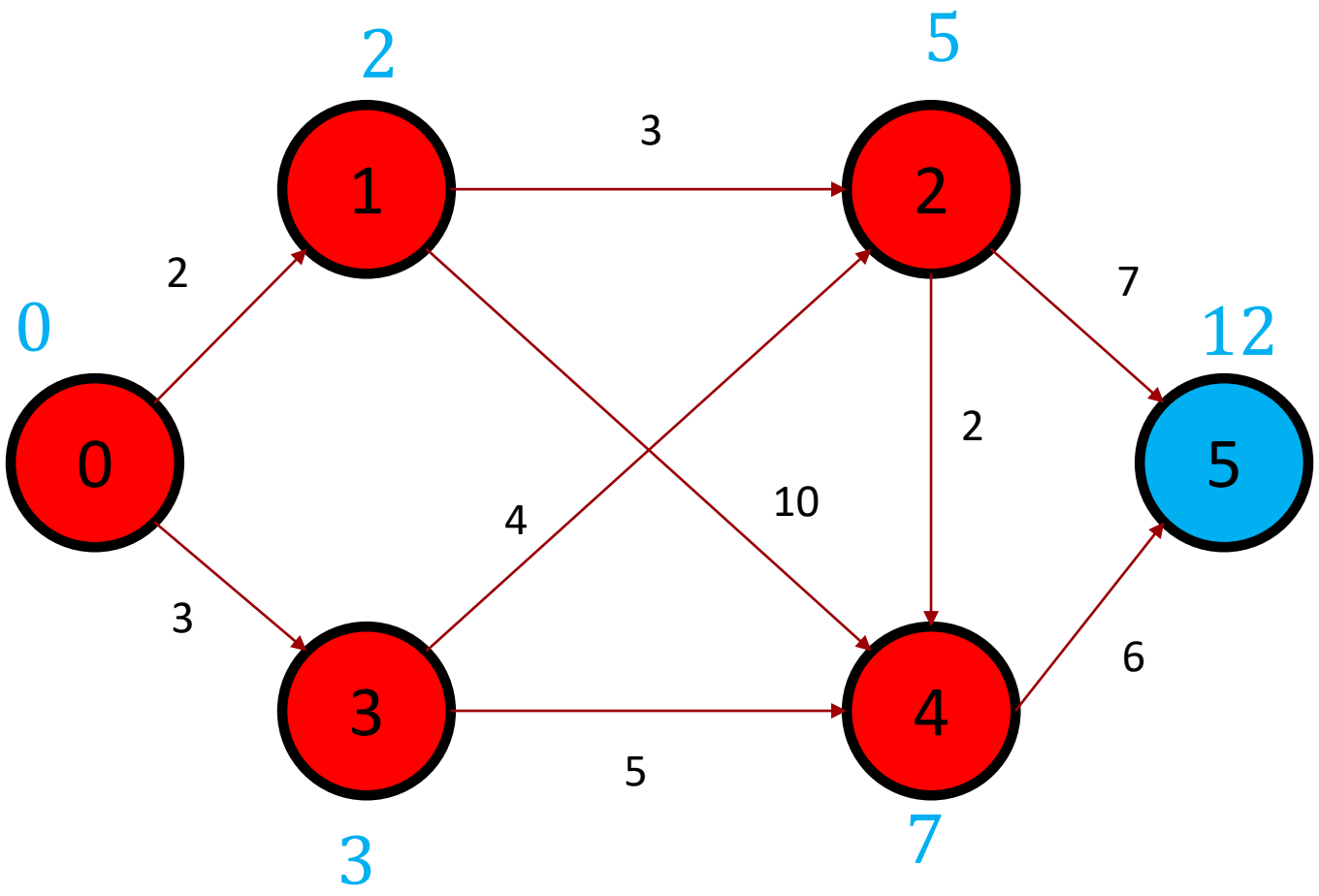
Dijkstra Algorithm



Dijkstra Algorithm



Dijkstra Algorithm



$answer\ is\ 12$

Dijkstra Algorithm

✓구현은 pq를 이용한다.

✓방문체크를 함에 있어서 유의해야 한다. pq에서 나온 값이 현재 거리와 일치하는지 여부를 통해 방문체크를 할 수 있다.

✓시간복잡도는 $O((V+E)logE)$

Dijkstra Algorithm

✓구현은 pq를 이용한다.

✓방문체크를 함에 있어서 유의해야 한다. pq에서 나온 값이 현재 거리와 일치하는지 여부를 통해 방문체크를 할 수 있다.

✓시간복잡도는 $O((V+E)logE)$

#1753 최단경로

문제

방향그래프가 주어지면 주어진 시작점에서 다른 모든 정점으로의 최단 경로를 구하는 프로그램을 작성하시오. 단, 모든 간선의 가중치는 10 이하의 자연수이다.

입력

첫째 줄에 정점의 개수 V와 간선의 개수 E가 주어진다. ($1 \leq V \leq 20,000$, $1 \leq E \leq 300,000$) 모든 정점에는 1부터 V까지 번호가 매겨져 있다고 가정한다. 둘째 줄에는 시작 정점의 번호 K($1 \leq K \leq V$)가 주어진다. 셋째 줄부터 E개의 줄에 걸쳐 각 간선을 나타내는 세 개의 정수 (u, v, w)가 순서대로 주어진다. 이는 u에서 v로 가는 가중치 w인 간선이 존재한다는 뜻이다. u와 v는 서로 다르며 w는 10 이하의 자연수이다. 서로 다른 두 정점 사이에 여러 개의 간선이 존재할 수도 있음에 유의한다.

출력

첫째 줄부터 V개의 줄에 걸쳐, i번째 줄에 i번 정점으로의 최단 경로의 경로값을 출력한다. 시작점 자신은 0으로 출력하고, 경로가 존재하지 않는 경우에는 INF를 출력하면 된다.

#1753 최단경로

```
priority_queue<pair<ll, ll>, vector<pair<ll, ll>>, greater<> >q;
const ll n_ = 20202, inf = 1e18;
ll V, E, K, dist[n_];
vector<pair<ll, ll>>v[n_];
```

거리를 저장할 $dist$배열, 간선 정보를 인접리스트로 저장할 $vector$
그리고 pq에 현재 정점의 거리와 현재 정점을 $pair$로 해서 넣는다.

#1753 최단경로

```
21      while (E--) {
22          ll a, b, c;
23          cin >> a >> b >> c;
24          v[a].push_back({ b,c });
25      }
26      fill(dist, dist + V + 1, inf);
27      dist[K] = 0;
28      q.push({ 0,K });
29      while (q.size()) {
30          auto cur = q.top();
31          q.pop();
32          if (dist[cur.second] < cur.first)continue;
33          for (auto nxt : v[cur.second]) {
34              if (dist[nxt.first] <= cur.first + nxt.second)continue;
35              dist[nxt.first] = cur.first + nxt.second;
36              q.push({ dist[nxt.first],nxt.first });
37          }
38      }
```

현재 거리와 저장되어 있는 최소거리와 비교를 하며, 다익스트라 알고리즘을 진행해 나간다. 마지막에는 시작 정점과의 최소 거리가 저장되어 있을 것이다.

#추천문제

필수문제

11286	절댓값 힙
11000	강의실 배정
1753	최단경로
1916	최소비용 구하기
20046	Road Reconstruction

연습문제

11279	최대 힙
1927	최소 힙
17396	백도어
13549	숨바꼭질 3
1261	알고스팟
1504	특정한 최단 경로
1715	카드 정렬하기
4485	녹색 옷 입은 애가 젤다지?
2665	미로만들기
10282	해킹
11779	최소비용 구하기 2
1238	파티
17940	지하철
1655	가운데를 말해요
1202	보석 도둑
5719	거의 최단 경로
9446	아이템 제작
1854	K번째 최단경로 찾기