7주차 예비보고서

전공: 아트앤테크놀로지학과 학년: 3학년 학번: 20201116 이름: 이수빈

1. 랭킹 시스템 구현 자료구조
   1. Linked List
   2. Priority Queue
2. 새로운 랭킹을 삽입 및 삭제를 구현하기 위한 pseudo code, 시간 및 공간 복잡도
   1. **Linked List를 이용한 priority queue**

삽입: 정렬되어 있는 링크드 리스트에 새로운 노드를 삽입하면 된다. 따라서 insertion sort와 같은 원리로 들어가게 된다

*pseudo code:*

Node \*cur= head;

if(head==NULL) head = new; //empty list

else

while(cur->next ->data > cur ->data){

cur = cur->next;

}

시간 복잡도 : O(n)

(탐색을 위한 시간 복잡도 : O(n), 삽입 시간 복잡도 : O(1) )

공간 복잡도 : O(n)

(노드의 개수 만큼의 공간이 필요함)

삭제:

*pseudo code:*

Node \*cur= head;

Node \*prev= head;

while(!cur){

if(cur->name = rank\_name) {

prev ->next = cur->next;

break;

}

prev = cur;

cur = cur->next;

}

시간 복잡도 : O(n)

(삭제하고자 하는 이름을 찾아서 해당 노드를 삭제)

공간 복잡도 : O(1)

* 1. **Binary Heap**

삽입: 이 문제의 경우 가장 큰 값이 트리의 가장 위에 있는 max heap의 형태로 구현한다.

*pseudo code:*

insert(arr[], n, i){

int max = root\_node;

int l = 2i+1, int r=2\*i+2;

if(l<n and arr[] >arr[i]) max = r;

if (r<n and arr[r] > arr[max]) max = r;

if (max !=i) {

temp = arr[i];

arr[i]=arr[max]l

arr[max]=temp};

insert(arr[], n, max);

}

시간 복잡도 : O(lgn)

공간 복잡도 : O(n)

삭제:

*pseudo code:*

delete(arr[], n, i){

int max = root\_node;

int l = 2i+1, int r=2\*i+2;

if(max == n) delete;

if(l<n and arr[] >arr[i]) max = r;

if (r<n and arr[r] > arr[max]) max = r;

if (max !=i) {

temp = arr[i];

arr[i]=arr[max]l

arr[max]=temp};

delete(arr[], n, max);

}

시간 복잡도 : O(lgn)

공간 복잡도 : O(n)

1. 원하는 정렬된 랭킹(*x*~*y*위, *x*≤*y*, *x*, *y*는 정수)의 정보를 얻는 방법
   1. **Linked List를 이용한 priority queue**

리스트에서 x번째부터 y번째 원소를 차례대로 출력한다.

*pseudo code:*

Node \*cur= head;

Node \*prev= head;

int count =0;

while(!cur){

if(count <x =and count >=y {

print(cur);

}

if(count ==(y-x)+1) break;

prev = cur;

cur = cur->next;

}

시간 복잡도: O(n)

공간 복잡도 : O(1)

* 1. **Binary Heap**

*pseudo code:*

위의 delete와 동일하지만 같은 name을 찾는 대신 풀력하고자 하는 인덱스에 해당하는 값이면 출력하는 구조를 갖게 한다.

시간 복잡도 : O(lgn)

공간 복잡도 : O(n)