9주차 예비보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211522 이름: 김정환

**1.**

.................

-Linked List

Linked List에서는 랭킹에 새로운 노드를 삽입하고 제거하는 과정이 Simple Array에 비해서 메메모리를 사용할 때 낭비없이 사용할 수 있다. 추가할 때는 점수를 비교하여 더 작은 점수를 가진 노드가 나올 때까지 탐색하여 삽입한다. 제거 시에는 연결을 끊고, 메모리를 해제하여 랭킹에서 제거할 수 있다. 단점으로는 인덱스로 접근하는 과정은 Simple Array와 비교하여 복잡하다고 할 수 있다.

-Simple Array

Linked List와 달리 기존의 C언어의 배열을 이용하는 구조이다. 장점으로는 인덱스로 접근하여 순서대로의 접근이 더 간단히 구현할 수 있는 장점이 있다. 그러나 노드를 새로 삽입하는 과정이나 제거하는 과정에서 뒷 부분의 노드를 모두 한 칸 씩 조정해야 하는 단점이 있다.

................

**2.**

.......................

-Linked List

1) 새로운 랭킹 삽입 시

새로운 랭킹 삽입 시에는 최악의 경우의 수가 제일 마지막에 삽입될 때까지 탐색하는 경우이다. 주어진 랭킹의 길이를 n이라 하면 시간 복잡도는 O(n)이고, n의 길이를 가진 리스트가 주어져야 하므로 공간 복잡도 역시 O(n)이다.

2) 랭킹 삭제 시

랭킹 삭제 시에도 최악의 경우의 수는 제일 마지막까지 탐색하고 삭제하는 경우이다. 주어진 랭킹의 길이를 n이라 하면 시간 복잡도는 O(n)이고, n의 길이를 가진 리스트가 주어져야 하므로 공간 복잡도 역시 O(n)이다.

-Simple Array

1) 새로운 랭킹 삽입 시

새로운 랭킹 삽입 시에는 원하는 지점까지 탐색 후 해당 지점부터 가장 뒤까지 한 칸씩 뒤로 밀고, 새로운 랭킹을 삽입한다. 이 경우에는 항상 끝까지 계산해야 하므로 시간 복잡도는 O(n)이고, 공간 복잡도 역시 O(n)이다.

2) 랭킹 삭제 시

랭킹 삭제 시에는 삭제를 원하는 지점부터 끝까지 한 칸씩 당겨와서 제거한다. 이 경우에는 최악의 경우가 제일 앞까지 하나씩 당기는 경우이므로 시간 복잡도는 O(n)이고 공간 복잡도 역시 O(n)이다.

.........................

**3.**

.......................

-Linked List

x~y위에 접근하기 위해서는 head 노드부터 x위에 도달할 때까지 탐색 후 해당 지점부터 y위까지 접근하는 방식을 사용한다. x위에 도달한 것을 확인하는 방법은 head부터 탐색할 때 idx를 1로 하고 다음 link로 넘어갈 때마다 idx를 하나씩 증가시켜 판단하는 방법이 있다.

-Simple Array

Simple Array의 경우에는 x위부터 y위까지 인덱스를 통해서 접근하는 방식으로 구현할 수 있다.

.........................