9주차 결과 보고서

전공: 컴퓨터공학 학년: 2학년 학번: 20221559 이름: 박준우

1. linked list를 사용했다. 정렬 된 상태로 삽입, 삭제를 할 때, 일반 array를 사용하면 삽입, 삭제 한 인덱스 뒤의 원소들을 이동시켜 주는 비용이 원소가 많아질수록 커지기 때문에 비효율적이라고 판단하여 linked list 자료구조를 사용했다.

1-1. X부터 Y까지 출력하기

Score\_count 변수가 있어서, 유효하지 않은 범위이면 실패하였다는 문구를 띄웠다.

유효한 범위임이 보장되면, score 정보를 담고있는 linked list 헤더노드 주소를 가지고 와서, 순서를 세어가며 탐색하고, 범위 안의 노드일 경우 출력해준다.

1-2. 문자열을 입력 받아 이름에 해당하는 리스트 출력

마찬가지로 Linked list 헤더의 주소를 가지고 와서 각각의 이름을 입력 받은 이름과 비교하여 출력한다. 없는 경우 실패 문구를 띄워주기 위해, 총 몇개가 일치하였는지 세어준다.

일치한 자료가 없는 경우 실패문구를 띄운다.

1-3. rank번호를 받아 리스트에서 삭제

1번일 경우와 아닌 경우의 행동에 차이가 있다.

삭제할 번호의 앞 노드의 다음 노드 주소를, 삭제할 번호의 다음 노드 주소로 바꿔준 후, 노드를 삭제하면 되는데, 1번 노드를 삭제하는 경우, 전역변수에 저장되어 있는 헤더노드 주소를 바꿔줘야 하기 때문이다.

따라서 입력받은 index가 유효한지 우선 판단한 후, 1번일 경우 헤더노드 주소를 바꿔주고, 1번이 아닐 경우 해당 index 앞의 노드까지 탐색한 후 지울 노드의 주소를 저장해놓고 지울 노드의 앞 노드와 뒤의 노드를 연결해준다.

그리고 지울 노드를 free 해준다.

1-4. 자료구조의 효율성을 시간, 공간적 측면에서 설명

범위 내의 점수를 출력하는 경우, 범위가 1부터 시작하지 않으면 출력하지 않아도 되는 노드를 탐색해야 하므로 일반 배열에 저장하는 것보다 일부 비효율적인 면도 있다.

그러나 정렬된 상태로 유지하는 데에 있어, 삽입, 삭제를 하기 위해서는 배열을 사용하면 목표 index 이후의 값들을 옮겨주어야 하는데, 이 자료는 문자열을 가지고 있기 때문에, 이를 옮겨주는 것에 대한 시간적 비용이 linked list를 사용하면 들지 않게 된다. 각각의 시간복잡도가 O(score\_count \* STRLEN), O(score\_count)가 된다. 따라서 공간적인 측면에서는 포인터를 따로 저장해주어야 한다는 단점이 있지만, 삽입, 삭제 시에 문자열을 이동시키지 않아도 되어 유리한 점이 있다. 공간 복잡도는 기존 선언된 것을 이용하거나, 하나의 노드만 새로 선언해 주기 때문에 O(1)로 차이가 없다.

2. 본 실험 및 숙제를 통해 습득한 내용을 기술하시오

우선 파일입출력에 대한 복습이 되어서 좋았고, linked list에서 별도의 헤더노드를 구현해 주어야 하는 이유에 대해서 이해할 수 있었다. (시작 인덱스의 삭제에서 별도의 구문 작성 필요 등)