7주차 결과 보고서

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

본 실험에서 ranking system을 구현하기 위해 linked list를 사용하였다.

Graphical user interface, text, application, chat or text message

Description automatically generated

각 노드는 점수와 이름을 저장할 수 있는 변수와 다음 노드를 가리키는 포인터로 구성되어 있다. 이를 구현하기 위해 노드의 가장 첫 번째 원소를 가르키는 포인터 head를 전역변수로 지정해준다. linked list 자료구조를 구현하기 위해 사용한 함수는 다음과 같다.

* Node \*make\_new (int n, char\* s)

이 함수는 점수와 이름을 입력 해주면 이 정보를 포함한 노드를 만들어 준다. 이때, Node가 가리키는 포인터는 NULL 값이다.

* void sortedInsert(Node \*\*head, Node \*new)

구현한 Linked List에 새로운 노드를 삽입 할 때, 내림차순으로 정렬되게 알맞은 위치에 삽입해주는 함수이다. 이 함수는 rank.txt의 정보로 linked list를 구현할 때, 원소 하나하나를 받아와 차례대로 이 함수를 이용해 linked list에 삽입 해 줄 때 사용된다.

* void printRank(Node \*\*head, int start, int end)

이 함수는 ranking system의 1번째 기능을 위해 구현한 함수이다. 출력하고 싶은 rank의 시작 값과 끝 값을 넘겨주면 count 변수로 순서를 세며 해당 되는 순서에 있는 점수와 이름을 출력해준다. 그리고 end보다 count가 커지면 더 이상 출력할 것이 없으므로 끝을 내준다.

* int find\_name(Node \*\*head, char \*search\_name)

이 함수는 ranking system의 2번째 기능을 위해 구현한 함수이다. 찾고자 하는 이름을 인자로 넘겨주면 linked list를 훑으며 찾고자 하는 이름과 해당 노드의 이름이 일치하면 출력해준다. 만약 해당 이름이 없다면 0을 반환한다.

* bool find\_rank(Node \*\*head, int rank)

이 함수는 ranking system의 3번째 기능을 위해 구현한 함수이다. 제거하고자 하는 rank 값을 넘겨주면 count 변수로 순서를 세며 해당 노드를 찾아 prev->next=cur->next 의 방법으로 해당 노드를 linked list에서 제외해준다. 만약 해당 rank를 삭제할 수 없다면 false를 반환한다.

Linked List로 구현한 이유는 구조를 짜기에 간편하기 때문이다. Binary Heap을 사용하면 시간 복잡도는 개선이 되겠지만( linked list와 binary heap의 각각 삽입과 삭제에 대한 시간 복잡도는 예비 보고서에 기술하였다), 구현이 복잡하다. 또한 rank의 개수가 크지 않은 파일을 다루기 때문에 linked list를 사용해도 충분하다. 또한 linked list는 Node의 개수만큼만의 공간 복잡도를 갖기 때문에 (O(n)) 공간 복잡도 측면에서도 좋다.

1. 본 실험을 통해 linked list의 구조와 구현 방법에 대해서 복습 할 수 있었다. 습득한 내용은 문제 1번에 나와 있는 내용들이다.