9주차 예비보고서

전공: 컴퓨터공학과 학년: 4학년 학번: 20212022 이름: 이예준

**1.**

Linked List를 이용하여 랭킹 시스템을 구현할 수 있다.

처음에 입력 받은 파일에서 랭킹 정보(이름, 점수)를 가져와 새로운 노드에 저장하고,

해당 노드를 오름차순으로 Linked List 형식으로 저장한다.

또한 게임이 종료됐을 경우에는 사용자의 이름을 입력 받으면 종료될 시점의 점수와

입력 받은 이름을 새로운 노드에 저장한 뒤 Linked List의 알맞은 위치를 찾아 삽입한다.

Array을 이용하여 랭킹 시스템을 구현할 수 있다.

먼저 이름과 점수를 저장하는 구조체를 만들어 그 구조체를 요소로 가지고 있는 배열을 선언한다.

입력 받은 파일에서 랭킹 정보(이름, 점수)를 가져와 인덱스 순서대로 배열에 저장한다.

그 다음 qsort와 같은 정렬 함수를 이용하여 오름차순으로 정렬한다.

만약 게임 종료 후 정보를 삽입할 때는 배열의 인덱스를 하나 더 늘려서 저장한 뒤

한 번 더 정렬 함수로 배열을 오름차순으로 정렬한다.

(처음에 배열을 선언할 때 충분한 크기로 선언한다.)

**2.**

새로운 랭킹 삽입 및 삭제 수도코드 시공간 복잡도

**-Linked List**

**void Insert()**

Node \*newNode = (Node\*)malloc(sizeof(Node)) ;

newNode->score = 점수;

strcpy( newNode->name, 이름 ) ;

newNode->next = NULL;

if( Linked List의 첫 노드 ) head = newNode ;

else{

Node \*cur = head;

Node \*prev = NULL;

if( 첫 노드보다 점수가 높다면 ){

첫번째 위치에 노드 삽입 ;

}

else{

while(cur != NULL & cur->score > newNode->score){

prev = cur;

cur = cur->next;

}

if( cur != NULL ) 중간 위치에 노드 삽입 ;

else 끝 위치에 노드 삽입 ;

}

}

**Bool Delete(int Target)**

if (head == NULL) return false;

if ( 첫번째 노드가 Target이면 ){

Node \*tmp = head;

head = head->next;

free(Node);

return true;

}

else {

Node \* cur = list->next;

Node \* prev = list;

while (cur != NULL && cur->score != Target) {

prev = cur;

cur = cur->next;

}

if ( 끝까지 Target이 없으면 ) return false;

prev->next = cur->next;

free(cur);

return true;

}

두 함수 모두 목표하는 위치를 찾기 위해 Linked List를 한바퀴 순회하므로

시간 복잡도는 O(n)이며, Insert에서 동적할당을 통해 추가적인 메모리를 사용하지만

단 한번만 사용하므로 두 함수의 공간 복잡도는 O(1)이다.

**-Array (구조체: Array, 배열: array)**

**void Insert()**

list\_num+=1;

array[list\_num].score = score;

array[list\_num].name = name;

qsort(array, array 크기, sizeof(Array), compare);

**Bool Delete(int Target)**

for(i=0 ~ size of array)

if( array[i].score == target ) break;

if( I == size of array) return false;

for(j=i ~ size of array)

array[j].score = array[j+1].score ;

array[j].name = array[j+1].name ;

return true;

Insert의 시간 복잡도와 공간 복잡도 모두 O(1)이다.

Delete의 시간 복잡도는 O(n)이고, 공간 복잡도는 O(1)이다.

**3.**

**-Linked List**

for(int count = 1; list != NULL; list = list->next, count++){  
 if ( X <= count <= Y ){

printw(“%s %d\n”, list->name, list->score);

}

else break;

}

Linked List를 한 바퀴 순회하기 때문에 시간 복잡도는 O(n)이며,

추가적인 메모리를 사용하지 않기 때문에 공간 복잡도는 O(1)이다.

**-Array**

for (i= X ~ Y ){

printw(“%s %d\n”, array[i].name, array[i].score);

}

배열을 한 바퀴 순환하기 때문에 시간 복잡도는 O(n)이며,

추가적인 메모리를 사용하지 않기 때문에 공간 복잡도는 O(1)이다.