9 주차 결과보고서

전공: 컴퓨터공학과 학년: 2 학년 학번: 20231632 이름: Jumagul Alua

1. 작성한 랭킹 시스템에서 사용된 자료구조는 단일 연결 리스트이다. 이 리스트의 각 노드는 플레이어의 이름과 점수를 저장하며, 다음 노드를 가리키는 포인터를 포함한다.

Choice 1:

```
if (choice == '1') {
    echo();
    printw("X: ");
    scanw("%d", &X);
    noecho();
    echo();
    printw("Y: ");
    scanw("%d", &Y);
    noecho();

    if(Y<=X||Y>score_number||X<1) {
        printw("search failure: no rank in the list");
    }
    else {
        count=0;
        Node *currentNode=liststart;
        printw("      name      |   score\n");
        printw("----------------------------\n");
        for(i=1; i<=Y; i++) {
            if(i>=X) {
                count++;
                printw("%-16s| %d\n", currentNode->name, currentNode->score);
                currentNode = currentNode->next;
            }
            else {
                currentNode = currentNode->next;
            }
```

사용자로부터 시작 순위(X)와 끝 순위(Y)를 입력받아 해당 범위의 순위를 출력하는 함수이다. 올바른 입력인 경우, 리스트의 첫 번째 노드부터 탐색을 시작한다. currentNode 포인터를 사용하여 리스트의 노드를 순차적으로 탐색하면서, 현재 노드의 순위가 X 와 Y 사이에 있는 경우, 해당 노드의 이름과 점수를 출력한다. 출력 후, currentNode 를 다음 노드로 이동시켜 계속 탐색한다. 탐색 시에는 리스트의 모든 노드를 확인해야 하므로 최악의 경우 O(n)의 시간 복잡도를 가진다.

Choice 2:

```
} else if (choice == '2') { //by a specific name
    char name[NAMELEN+1];
    Node *currentNode = liststart;
    Node *prevNode = NULL;
    int found = 0;

    echo();
    printw("Input the name: ");
    scanw("%s", name);
    noecho();

    printw("      name      |   score\n");
    printw("---------------------------\n");
    while(currentNode!=NULL){
        if(!strcmp(currentNode->name, name)){
            found=1;
            printw("%-16s| %d\n", currentNode->name, currentNode->score);
            if (prevNode == NULL) {
                liststart = currentNode->next;
            } else {
                prevNode->next = currentNode->next;
            }
        Node* tempNode = currentNode;
        currentNode = currentNode->next;
        free(tempNode);
        score_number--;
    } else {
        prevNode = currentNode;
        currentNode = currentNode->next;
    }
    }
    if(!found){
    printw("search failure: no rank in the list\n");
```

리스트의 첫 번째 노드부터 시작하여 마지막 노드까지 탐색한다. 현재 노드의 이름과 사용자가 입력한 이름을 비교하고 노드를 삭제하기 위해 링크를 수정도 해준다. 랭킹 시스템에 2 를 선택하면, 특정 이름으로 순위를 검색하고 삭제하는 기능을 추가했고 여기서도 최악의 경우에는 모든 노드를 탐색할거라 O(n) 시간 복잡도가 필요하고 공간 복잡도는 O(1)이 된다.

Choice 3:

```
} else if (choice == '3') {//delete a specific rank
    int number;
    echo();
    printw("Input the rank: ");
    scanw("%d", &number);
    noecho();
    if (number<1 || number>score_number){
        printw("\nsearch failure: no rank in the list\n");
    }
    else {
        Node *currentNode = liststart;
        Node *prevNode = NULL;
        Node *tempNode = NULL;

        for(int i=1; i<number; i++){
            prevNode = currentNode;
            currentNode = currentNode -> next;
        }
        if(prevNode==NULL){
            liststart = currentNode -> next;
        }else{
            prevNode->next = currentNode->next;
        }
        free(currentNode);
        score_number--;
    }
}
getch();
```

Node *currentNode 와Node *prevNode 를 사용하여 리스트를 탐색한다. 첫 번째 노드부터 시작하여 삭제하고자 하는 순위 번호에 도달할 때까지 노드를 순차적으로 탐색한다. 삭제할 노드가 리스트의 첫 번째 노드인 경우 리스트의 시작 노드를 현재 노드의 다음 노드로 설정해주고, 만약, 삭제할 노드가 리스트의 중간이나 마지막 노드인 경우, 이전 노드의 다음 노드를 현재 노드의 다음 노드로 설정한다. 그 다음에 노드를 삭제하고 메모리를 해제한다. 최악의 경우 O(n)의 시간 복잡도를 가진다. 추가적인 메모리는 모두 상수 개수의 변수들에 해당하므로, 이 부분의 공간 복잡도는 O(1)이다.

2. 이번 실험에서는 연결 리스트를 사용하여 테트리스 게임의 랭킹 시스템을 구현했다. 이를 통해 동적 메모리 할당과 해제, 사용자 입력 처리, 화면 출력 등의 기술을 익혔다. 구현한 랭킹 시스템은 순위 출력, 이름으로 검색, 순위 삭제 등의 기능을 제공하여 게임의 사용자 경험을 향상시켰다. 대부분 최악의 경우라도 O(n) 시간 복잡도를 보여줬기 때문에 메모리 관리를 향상시킬 수 있었어 좋았다.