실험 PRJ-2 테트리스 2주차/3주

1. 목적

일반적인 게임은 사용자의 기록을 등록하여, 확인할 수 있는 기능을 갖는다. 테트리스 프로젝트에서도 이러한 기능을 하는 랭킹 시스템을 구현한다. 이 시스템은 테트리스 플레이가 종료되면, 사용자의 이름과 점수(score)를 기록하고, 기록된 랭킹 정보들을 확인할 수 있는 기능을 갖는다. 이 랭킹 정보들은 테트리스 프로그램 내에서 효율적인 자료구조를 사용하여 관리되고, 테트리스 프로그램을 종료할 때는 rank.txt 파일에 기록되어, 지속적으로 유지된다.

2. 랭킹 시스템의 랭킹 정보와 input 파일

2-1 랭킹 정보

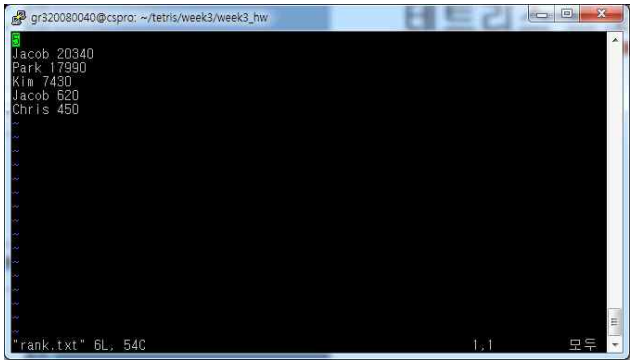
랭킹 시스템에서 랭킹 정보는 사용자의 이름(character 배열)과 점수(score)로 구성된다.

2-2 Input 파일(rank.txt)

랭킹 시스템에서 랭킹 정보를 기록하고 있는 파일은 rank.txt로 다음과 같이 먼저 랭킹 정보의 수(정수)가 기록되고, 사용자 이름, 점수(score) 순으로 랭킹 정보들이 기록된다.

예) rank.txt

```
랭킹정보의수(정수)
사용자이름1      점수1
사용자이름2      점수2
...              ...
```



[그림 1] rank.txt 예제

3. 랭킹 시스템의 기능

3-1 rank.txt로부터 랭킹 정보를 읽어서 랭킹 목록 생성

랭킹 시스템의 첫 번째 기능은 테트리스 게임이 실행될 때, rank.txt로부터 랭킹 정보(사용자 이름과 점수)를 읽어 들여서 랭킹 정보들에 대한 목록을 생성하는 기능이다. 이렇게 구축된 랭킹 목록은 테트리스 게임이 수행되는 동안 계속 유지된다.

3-2 생성된 랭킹 목록으로부터 원하는 랭킹 정보 출력

사용자로부터 두 정수 X, Y(X≤Y)를 입력받아 테트리스 랭킹 정보를 유지하고 있는 랭킹 목록으로부터 점수 순으로 X~Y위까지 랭킹 정보를 출력하는 기능이다.

3-3 게임종료 시 사용자 이름을 입력 받고, 랭킹 정보를 자료구조에 추가

테트리스 게임을 플레이 한 후, 게임 종료시('Q' 또는 'q'를 입력하여 강제 종료한 경우는 제외) 사용자의 이름을 입력 받고, 입력받은 사용자 이름과 점수로 구성된 새로운 랭킹 정보를 랭킹 목록에 추가한다.

3-4 테트리스 프로그램 종료시 갱신된 랭킹 정보를 rank.txt에 기록

테트리스 메뉴에서 4번 exit를 선택했을 때, 초기의 랭킹 정보의 상태와 비교해서 만약 변화가 있다면, 테트리스 랭킹 정보를 유지하고 있는 랭킹 목록을 하나씩 탐색하여 rank.txt에 다시 기록한다. 이 과정은 rank.txt가 항상 최신 랭킹 정보를 유지하게 한다.

4. 랭킹 시스템 구현을 위한 실험에서 이슈 – 자료구조 선택

앞에서 설명한 랭킹 시스템의 기능을 구현하기 위해서는 랭킹 정보를 유지할 수 있는 랭킹 목록이 필요하다. 이 랭킹 목록은 지금까지 배운 자료구조들 중 하나의 자료구조에 의해 구현된다. 지금까지 실험에서는 주어진 자료구조를 사용했지만, 테트리스 프로젝트 2주차 실험에서는 앞에서 언급된 랭킹 시스템의 4가지 기능들을 고려하여, 이 기능들을 가장 효율적으로 수행할 수 있는 자료구조를 선택하여, 랭킹 목록을 저장하여 유지할 수 있도록 구현한다. 이 때, 각 기능들의 효율성은 시간 복잡도를 의미한다. 그리고 이 후 설명에서는 설명이 길어지는 것을 방지하기 위해서, 랭킹 목록을 저장 및 유지하기 위해 선택된 자료구조를 단순히 자료구조라 부르기로 한다.

4-1 자료구조의 선택 시 고려할 사항들

1. 랭킹 정보들을 저장하기 위해 구축된 자료구조에서 원하는 범위(X~Y위)의 정보를 추출할 때, 정렬된 상태로 추출가능한가의 여부와 이 과정을 수행하는데 발생하는 시간 복잡도.
2. 게임이 종료되었을 때, 새로운 사용자 이름을 입력 받고, 새로운 랭킹 정보를 자료구조에 삽입 할 때, 발생하는 시간 복잡도.
3. 원하는 랭킹을 자료구조로부터 삭제 시 발생하는 시간 복잡도(2주차 숙제).
4. 찾길 원하는 사용자 이름과 일치하는 랭킹 정보 모두를 자료구조로부터 추출하기 위해 발생하는 시간 복잡도(2주차 숙제).

4-2 자료구조 선택의 범위

현재까지 강의를 통해 배우거나, 본인이 알고 있는 모든 자료구조를 사용할 수 있다.

5. 랭킹 시스템 예제 – Linked list를 자료구조로 선택한 경우

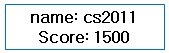
랭킹 시스템을 구현하기 위한 자료구조로, linked list를 선택했을 때, 어떤 과정을 거쳐서 랭킹 정보가 linked list에 저장 및 구성되는지를 예를 통해서 확인해보자. 이 때, rank.txt는 빈 파일이라고 가정한다. 그리고 아래 그림에서 굵은 화살표는 탐색된 링크를 나타낸다.

☞ 테트리스 게임 전

[그림 2] 빈 head 노드

☞ 1번째 테트리스 게임 플레이(play) 후, 사용자 cs2011이 1500점 획득하고 게임 종료.

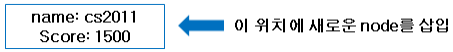
- 사용자 이름 cs2011, 점수 1500을 갖는 head 노드(node)가 생성된다.



[그림 3] 사용자 이름 cs2011, 점수 1500을 갖는 새로운 head 노드 생성

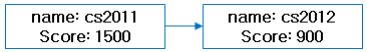
☞ 2번째 테트리스 게임 플레이 후, 사용자 cs2012이 900점 획득하고 게임 종료.

- 링크(link)를 따라 가면서 점수를 비교하여 새로운 노드를 삽입할 적절한 위치를 탐색한다.



[그림 4] 탐색을 통해서 찾은 새로운 랭킹 정보를 삽입할 위치

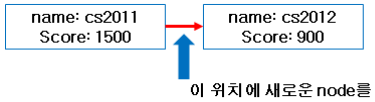
- 사용자 이름 cs2012, 점수 900을 갖는 새로운 노드를 삽입한다.



[그림 5] 사용자 이름 cs2012, 점수 900을 갖는 새로운 노드 삽입

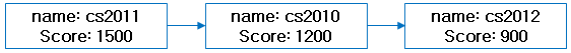
☞ 3번째 테트리스 게임 플레이 후, 사용자 cs2010이 1200점 획득하고 게임 종료.

- 링크를 따라 가면서 점수를 비교하여 새로운 노드를 삽입할 적절한 위치를 탐색한다.



[그림 6] 탐색을 통해서 찾은 새로운 랭킹 정보를 삽입할 위치

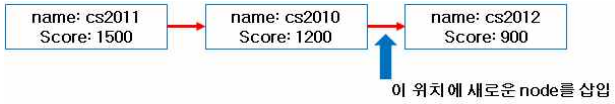
- 사용자 이름 cs2010, 점수 1200을 갖는 새로운 노드를 삽입한다.



[그림 7] 사용자 이름 cs2010, 점수 1200을 갖는 새로운 노드 삽입

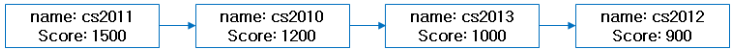
☞ 4번째 테트리스 게임 플레이 후, 사용자 cs2013이 1000점 획득하고 게임 종료.

- 링크를 따라 가면서 점수를 비교하여 새로운 노드를 삽입할 적절한 위치를 탐색한다.



[그림 8] 탐색을 통해서 찾은 새로운 랭킹 정보를 삽입할 위치

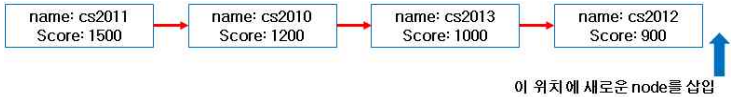
- 사용자 이름 cs2013, 점수 1000을 갖는 새로운 노드를 삽입한다.



[그림 9] 사용자 이름 cs2013, 점수 1000을 갖는 새로운 노드 삽입

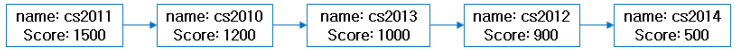
☞ 5번째 테트리스 게임 플레이 후, 사용자 cs2014이 500점 획득하고 게임 종료.

- 링크를 따라 가면서 점수를 비교하여 새로운 노드를 삽입할 적절한 위치를 탐색한다.



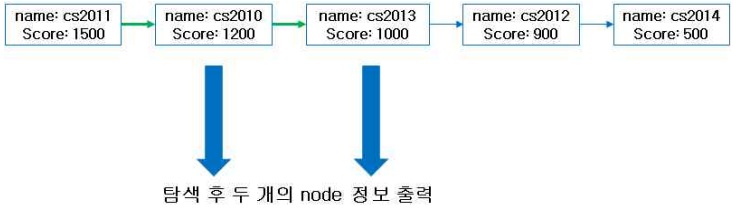
[그림 10] 탐색을 통해서 찾은 새로운 랭킹 정보를 삽입할 위치

- 사용자 이름 cs2014, 점수 500을 갖는 새로운 노드를 삽입한다.



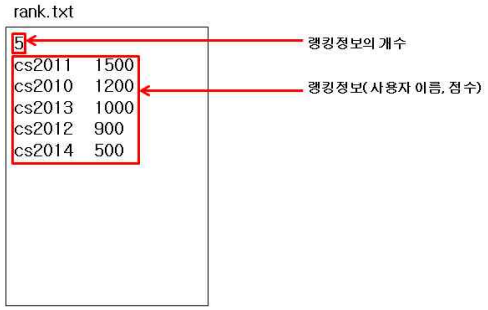
[그림 11] 사용자 이름 cs2014, 점수 500을 갖는 새로운 노드 삽입

☞ X=2, Y=3을 입력받아 2위부터 3위의 랭킹 정보를 출력할 때.



[그림 12] 2위~3위 정보를 추출할 때, 탐색된 링크와 추출된 노드들

☞ 테트리스 프로그램 종료 후 랭킹 정보를 저장하고 있는 rank.txt
- 먼저 랭킹 정보의 총 개수를 기록하고, 구성된 linked list를 처음부터 탐색하여 각 랭킹 정보를 사용자 이름과 점수 순으로 기록한다.



[그림 13] 랭킹 목록을 포함하는 linked list의 정보가 기록된 rank.txt

6. 랭킹 시스템을 위한 노드 구조, 구현 및 결과

6-1 랭킹 정보를 저장하기 위해 사용하는 노드 구조(node structure)

```
Typedef struct _Node {
        char name[NAMELEN];
        int score;
        ...
       } Node;
```

[그림 14] 랭킹 시스템에서 사용되는 노드 구조

위의 NAMELEN은 tetris.h에 16으로 define되어 있다.

6-2 구현할 함수의 전체 흐름과의 관계

아래 그림에서 음영 처리된 부분이 2주차 실습에서 구현할 함수들이다. 이들은 총 4가지 함수로 `createRankList()`, `rank()`, `newRank()`, `writeRankFile()`이다.

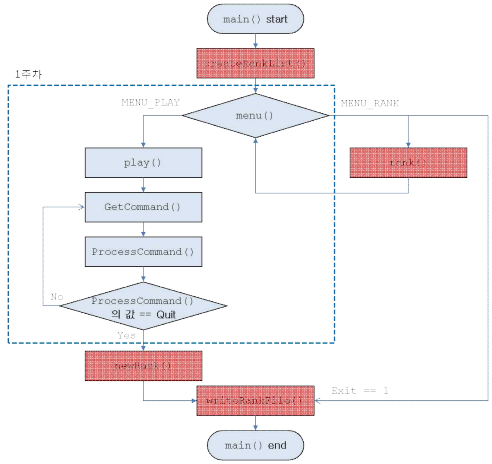
☞ **`createRankList()`** : Input 파일인 rank.txt로부터 랭킹 정보를 읽어들여, 선택한 자료구조에 저장하는 기능 2-1을 구현.

☞ **`rank()`** : 랭킹 정보를 확인하기 위해 두 개의 정수(X, Y)를 입력받아 X~Y위까지 화면에 출력하는 기능 2-2를 구현.

☞ **`newRank()`** : 더 이상 블록을 필드에 쌓을 수 없을 때, 즉 게임 종료 시 사용자의 이름을 입력받고, 사용자의 이름과 점수로 구성된 새로운 랭킹 정보를 자료구조에 추가하는 기능 2-3을 구현.

☞ **`writeRankFile()`** : 테트리스 프로그램을 종료시 자료구조에 저장된 랭킹 정보를 rank.txt에 기록하는 기능 2-4를 구현.

아래의 flow chart는 2주차 실험의 전체적인 과정(1주차 포함)을 나타내고 있고, flow chart에서 진하게 표시된 4개의 함수를 구현하게 된다.



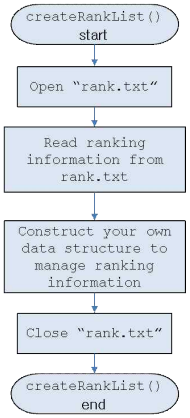
[그림 15] 랭킹 시스템의 flow chart 및 구현할 함수들

테트리스 2주차 프로젝트의 전체적인 과정을 보여주는 위의 flow chart에서 확인할 수 있듯이, 랭킹 시스템을 구현하기 위한 4가지 함수들은 진하게 표시되어 있다. 위 flow chart와 관련해 구현할 각 함수의 역할을 설명하면 다음과 같다. 테트리스 프로그램 수행 시 `createRankList()` 함수를 통해서 rank.txt로부터 랭킹 정보를 읽어들여 선택한 자료구조에 저장하고, 테트리스 기본 메뉴에서 2번을 선택한 경우, `rank()` 함수에서 두 개의 정수 X, Y를 입력받고, X~Y위까지 원하는 범위의 랭킹 정보를 자료구조로부터 추출하여 화면에 출력한다. 그리고 테트리스 게임이 정상적으로 종료가 되면, `newRank()` 함수에서는 사용자의 이름을 입력받고, 새로운 랭킹 정보를 자료구조에 삽입하며, 테트리스 프로그램을 종료하게 되면, `writeRankFile()` 함수가 호출되어 랭킹 정보들이 rank.txt에 다시 기록된다.

6-3 구현할 함수

1. createRankList()

이 함수에서는 rank.txt로부터 랭킹 정보를 차례로 읽어 들여 선택한 자료구조에 저장하는 기능을 한다. 여기서 구축된 자료구조는 테트리스 프로그램을 수행하는 동안 유지된다. 이 함수에서의 수행 과정은 다음과 같이 flow chart로 나타낼 수 있다.

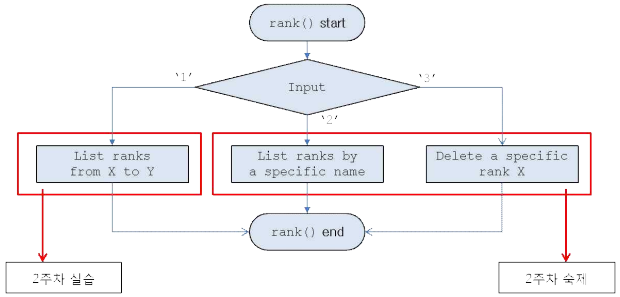


[그림 16] createRankList()의 flow chart

함수 createRankList()에서는 가장 먼저 랭킹 정보들이 저장되어 있는 rank.txt 파일을 열고, 랭킹 정보의 개수를 읽어 들인다. 이 랭킹 정보의 수만큼 반복하면서 랭킹 정보들을 하나씩 읽고, 읽어들인 랭킹 정보들을 자료구조에 하나씩 저장하게 된다. 이 모든 동작이 종료되면, rank.txt 파일을 닫고 함수를 종료한다.

2. rank()

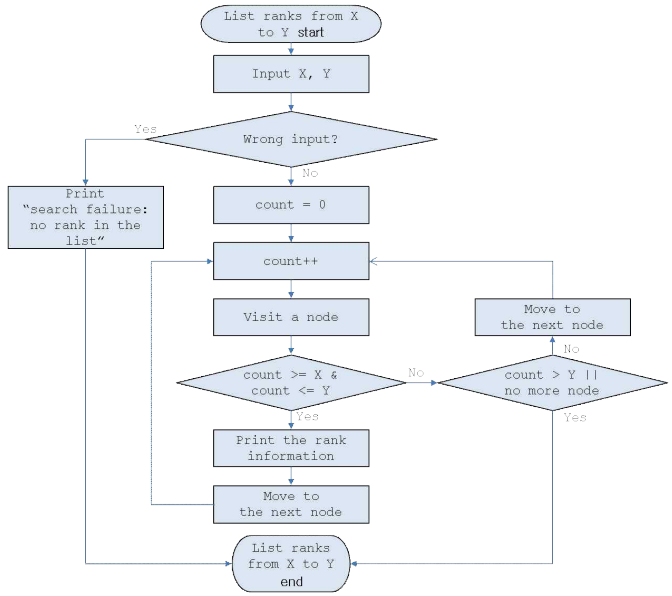
함수 rank()에서는 아래와 같이 기본적으로 3가지 기능을 수행한다. 첫 번째 기능인 랭킹 정보를 X~Y위까지 출력하는 기능은 2주차 실습에서 구현되고, 나머지 2개의 기능은 2주차 숙제에서 구현한다. 두 번째 기능은 사용자 이름이 일치하는 모든 랭킹 정보를 찾는 기능이고, 세 번째 기능은 입력 받은 랭킹을 삭제하는 기능이다.



[그림 17] rank()의 flow chart

기능 1: X~Y순위까지 출력

이 기능은 X~Y위까지 출력하는 기능으로 수행 과정은 flow chart와 같다.



[그림 18] X~Y순위까지 출력하는 기능 1의 flow chart

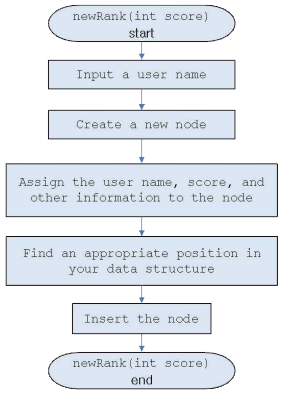
위의 flow chart에서 확인할 수 있듯이, 이 기능은 다음과 같은 과정을 통해서 수행된다. 가장 먼저 정수 X, Y를 입력받고, 잘못된 입력인지(조건 X≤Y 포함)를 체크한다. 만약 잘못된 입력을 받았다면, 일치하는 정보가 없다는 메시지 "search failure: no rank in the list"를 출력하고, 정상적인 입력을 받았다면, 자료구조에서 원하는 범위의 랭킹 정보를 추출하는 과정을 수행한다. 원하는 범위의 랭킹 정보 추출은 count 변수를 바탕으로 한다. 점수 순으로 랭킹 정보를 탐색할 때마다 count 변수를 증가시키면서(점수 순으로 1위 탐색 후 count 변수 1증가, 2위 탐색 후 count 변수 1증가하는 방식), count가 X일 때부터 Y일 때까지 랭킹 정보를 찾아내서 출력하게 된다. 만약 count가 X보다 크거나 같고, Y보다 작거나 같은 범위를 벗어나게 되면, 두 가지 경우로 나뉘어 처리되는데, 첫 번째, count값이 Y보다 크지 않고, 탐색할 다른 노드들이 있다면 다음 노드를 탐색하고, 두 번째, count 값이 Y보다 크거나, 더 이상 탐색할 노드가 없다면 기능을 종료한다.

※ 참고: 위의 count 변수를 이용한 과정은 구현을 쉽게 만들어주거나, 위 기능의 설명을 위한 부가적인 변수로 반드시 사용될 필요는 없다.

3. newRank()

이 함수는 테트리스 게임 종료시, 사용자의 이름을 입력받고, 입력받은 사용자의

이름과 점수로 이루어진 랭킹 정보를 자료구조에 삽입하는 기능을 한다. 이 함수는 테트리스 게임 종료시 호출되고, 그 수행 과정은 다음의 flow chart를 통해서 확인할 수 있다.

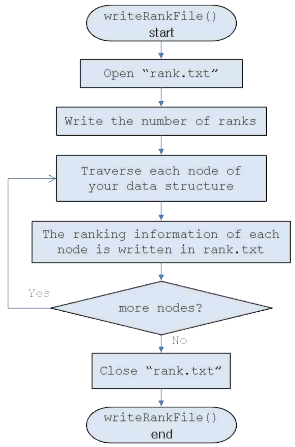


[그림 19] newRank()의 flow chart

위 flow chart에서 확인할 수 있듯이, 가장 먼저 사용자의 이름을 입력받고, 입력받은 사용자 이름과 플레이된 최종 score로 구성된 새로운 랭킹 정보, 즉 새로운 노드를 생성한다. 그리고 이 새로운 노드를 랭킹 목록을 저장하고 있는 자료구조의 적절한 위치를 찾아 삽입하고 종료한다.

4. writeRankFile()

테트리스 프로그램이 종료될 때, 구축된 자료구조에 랭킹 정보들을 다시 rank.txt에 저장하는 기능을 한다. 이 과정은 다음과 같은 flow chart를 따라 수행된다.



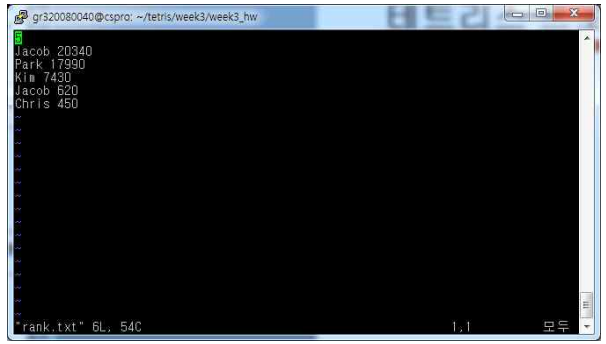
[그림 20] writeRankFile()의 flow chart

함수 writeRankFile()은 먼저 랭킹 정보를 기록하기 위해서 rank.txt 파일을 연다(open). 자료구조에 저장하고 있는 랭킹 정보의 개수를 count하거나 알아내(이 정보를 다른 정수형 변수에 저장하고 있어도 된다. 하지만, 랭킹 정보 삽입 및 삭제시 저절한 조치를 취해주어야 한다) 랭킹 정보의 수를 rank.txt의 최상단에 기록하고 자료구조의 각 노드를 탐색하면서 랭킹 정보, 즉, 사용자 이름과 점수로 구성된 랭킹 정보들을 차례로 rank.txt파일에 기록한다. 더 이상 탐색할 노드가 없을 때, rank.txt를 닫고, 이 과정을 종료한다.

6-4 구현 결과

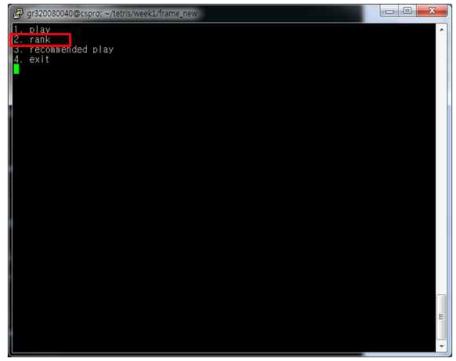
1. 랭킹 정보 확인

- 입력 파일 rank.txt



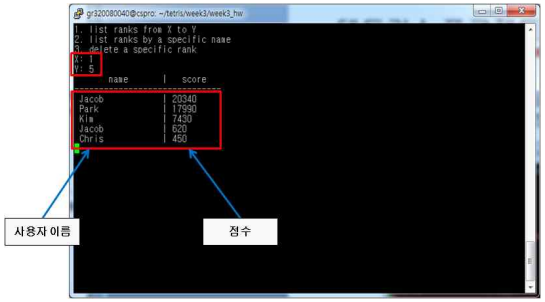
[그림 21] 입력 파일 rank.txt

- 랭킹 정보를 확인하기 위해 2를 입력



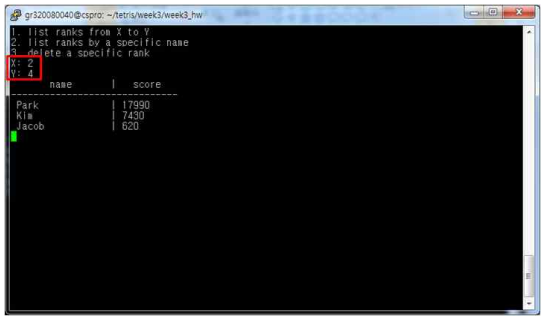
[그림 21] 테트리스 메뉴 화면(2번 rank 선택)

- 출력1 : 1위부터 5위까지 순위를 확인하기 위해 X=1, Y=5를 입력하고 점수 순으로 1위부터 5위까지 출력



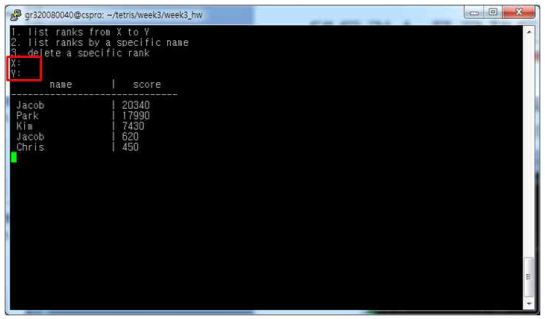
[그림 22] 점수 순으로 1~5위까지 출력된 결과(입력 X=1, Y=5)

- 출력2 : 2위부터 4위까지 랭킹을 확인하기 위해 X=2, Y=4를 입력하고 점수 순으로 2위부터 4위까지 출력



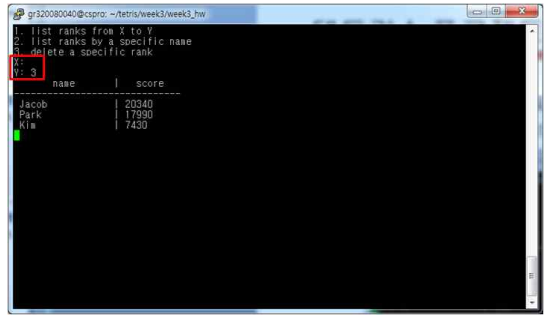
[그림 23] 점수 순으로 2~4위까지 출력된 결과(입력 X=2, Y=4)

- 출력3 : X와 Y를 모두 입력하지 않았을 경우. 모든 순위 출력(1~5위)



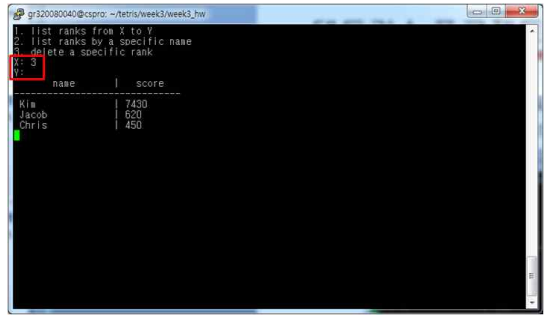
[그림 24] 아무것도 입력하지 않을 때, 1~5위까지 출력된 결과

- 출력4 : X를 입력하지 않고, Y=3을 입력하는 경우. 1~3위까지 출력



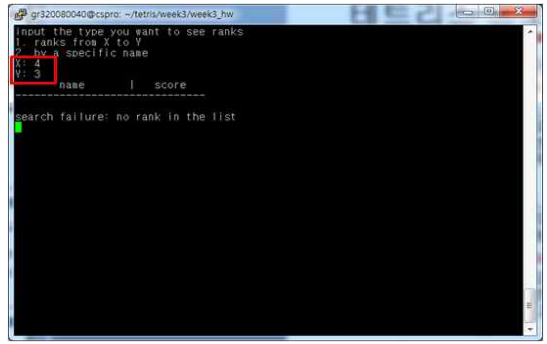
[그림 25] X는 입력하지 않고, Y=3일 때, 1~3위까지 출력된 결과

- 출력5 : X=3을 입력하고 Y를 입력하지 않는 경우. 3~5위까지 출력



[그림 26] X=3이고, Y는 입력하지 않았을 때, 3~5위까지 출력된 결과

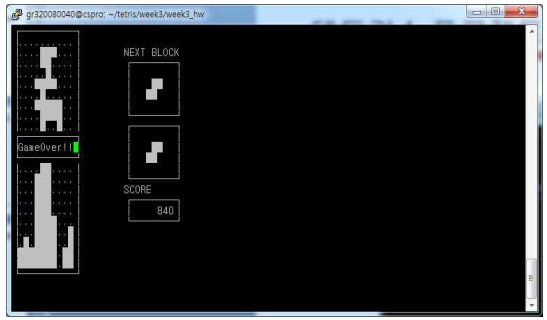
- 출력6 : X=4, Y=3을 입력하는 경우. 어떤 랭킹도 출력하지 않고 “search failure: no rank in the list”를 출력.



[그림 27] X=4이고, Y=3일 때, 일치하는 랭킹 정보가 없을 때의 결과

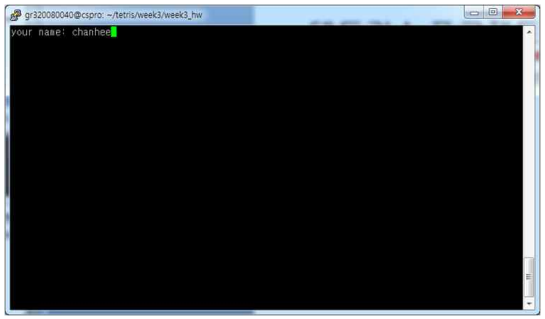
2. 게임 종료시 랭킹 정보 등록

- 새로운 랭킹 정보를 등록하기 위해 게임 종료



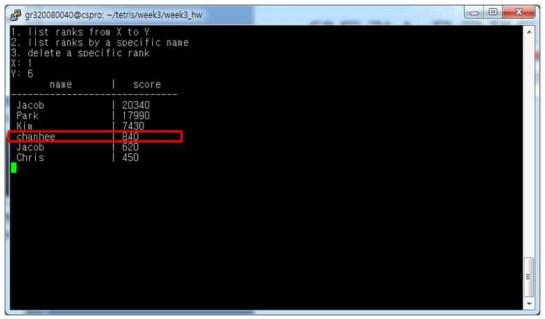
[그림 28] 테트리스 게임 종료

- 사용자 이름 입력 : “chanhee” 입력



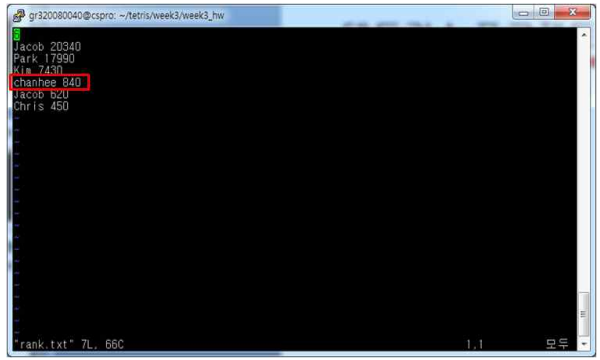
[그림 29] 테트리스 게임 종료 후 사용자 이름 입력(“chanhee”)

- chanhee의 랭킹 정보 확인



[그림 30] 새로운 랭킹 정보가 랭킹 목록을 저장하는 자료구조에 잘 삽입된 상태

- 프로그램 종료 후 rank.txt에서 갱신된 랭킹 정보(chanhee, 840) 확인



[그림 31] 새로운 랭킹 정보가 rank.txt에 잘 기록된 상태

7. 테트리스 프로젝트 2주차 숙제 및 보고서 작성

7-1. 예비보고서

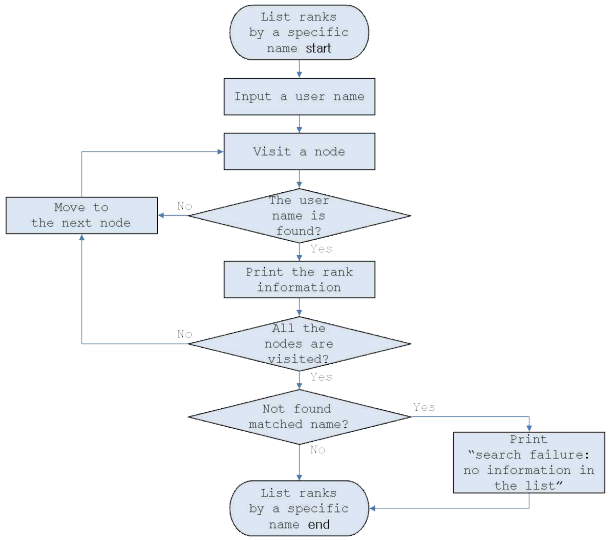
1. 2주차 실습에 구현하는 랭킹 시스템에 대한 자료를 읽어보고, 이를 구현하기 위한 자료구조를 2가지 이상 생각한다.
2. 생각한 각 자료구조에 대해서 새로운 랭킹을 삽입 및 삭제를 구현하기 위한 pseudo code를 작성하고, 시간 및 공간 복잡도를 계산한다.
3. 생각한 각 자료구조에서 사용자가 부분적으로 확인하길 원하는 정렬된 랭킹(x~y위, $x \le y$, x, y는 정수)의 정보를 얻는 방법을 간략히 요약해서 pseudo code로 작성하고, 시간 및 공간 복잡도를 계산한다.

7-2. 숙제

테트리스 2주차 숙제에서는 기존의 원하는 범위의 랭킹 정보를 출력하는 모드에 앞에서 언급한 `rank()`의 기능 2, 3을 추가적으로 구현하는 것을 목적으로 한다.

1. 기능 2(사용자 이름으로 탐색): 사용자의 이름을 입력 받아, 입력받은 사용자 이름에 해당하는 모든 랭킹 정보를 찾고, 찾은 랭킹 정보들을 화면에 출력하는 기능이다.
   - 구축된 자료구조를 유지 및 확장하거나 새로운 자료구조를 사용할 수 있다.
   - 랭킹정보들을 출력하는 양식은 기능 1과 같다.
   - 사용자의 이름을 입력받는다.
   - 일치하는 사용자가 있을 때, 해당 사용자의 모든 랭킹 정보를 기존 랭킹 시스템의 출력 방식과 같이, 사용자 이름과 점수 순으로 화면에 출력한다.
   - 일치하는 사용자가 없을 때는 “search failure: no name in the list”를 출력한다.

기능 2의 전체적인 과정은 다음 flow chart에서 확인할 수 있다.

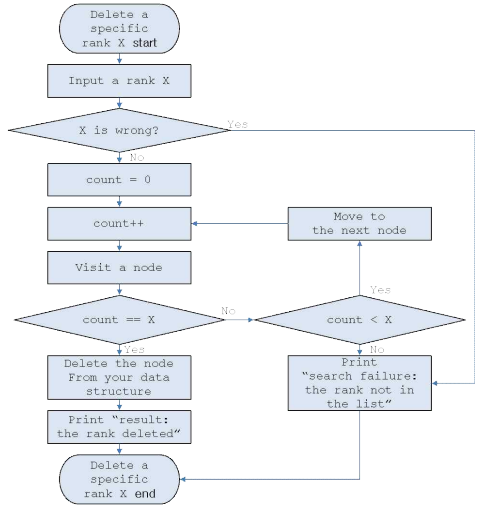


[그림 32] 기능 2의 flow chart

위 기능 2의 flow chart에 대한 간략한 설명은 다음과 같다. 먼저 사용자의 이름(character 배열)을 입력받고, 랭킹목록을 저장하고 있는 자료구조의 노드들을 하나씩 보면서 일치하는 이름이 있는지 체크한다. 만약 일치하는 이름이 있다면 해당 랭킹 정보를 갖는 노드의 정보를 화면에 출력하고 그렇지 않으면, 다음 노드로 이동한다. 이 과정을 반복하게 되는데, 만약 모든 노드를 탐색했다면, 자료구조의 탐색을 종료한다. 그 후에는 입력받은 사용자 이름과 일치하는 사용자 이름이 적어도 한 개 이상 존재한다면, 해당하는 랭킹 정보들이 화면에 출력되었으므로, 기능 2를 종료하고, 일치하는 이름이 존재하지 않는다면, "search failure: no information in the list"라는 메세지를 출력하고 기능 2를 종료한다.

2. 기능 3(랭킹 삭제): 삭제하길 원하는 랭킹 정보를 삭제 하는 기능이다.
   - 구축된 자료구조나 새로운 자료구조를 사용할 수 있다.
   - 삭제하길 원하는 랭킹(정수)을 입력 받는다.
   - 일치하는 랭킹이 있을 때는 해당하는 랭킹 정보를 삭제하고, "result: the rank deleted"를 출력한다.
   - 일치하는 랭킹이 없을 때는 "search failure: the rank not in the list"를 출력한다.

기능 3의 전체적인 과정은 다음 flow chart에서 확인할 수 있다.

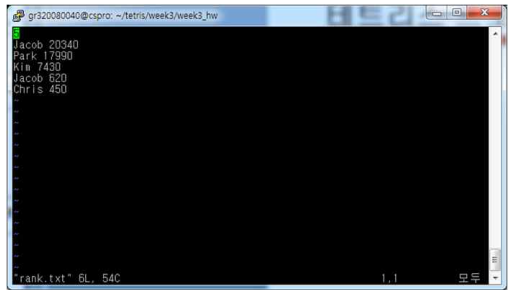


[그림 33] 기능 3의 flow chart

위 기능 3의 flow chart에 대한 간략한 설명은 다음과 같다. 가장 먼저, 삭제하길 원하는 랭킹(정수)을 입력받고, 입력받은 랭킹이 존재하는지 확인한다. 해당 랭킹이 범위를 벗어나지 않는다면, count 변수를 사용해서 입력받은 랭킹에 대응하는 자료구조 내의 노드를 찾아 삭제한 후, 화면에 "result: the rank deleted"를 출력하고, 해당 랭킹이 범위를 벗어난다면, 화면에 "search failure: the rank not in the list"를 출력하고 종료한다.

- 구현 결과

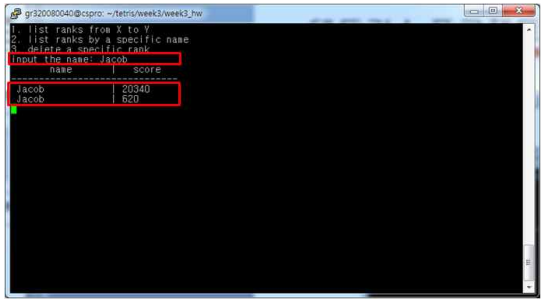
1. 입력파일



[그림 34] 입력 파일 rank.txt

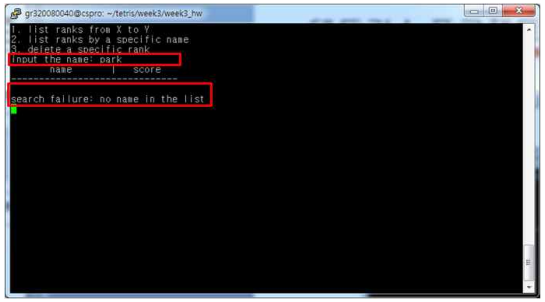
2. 기능 2의 예제

- Jacob을 입력하여 Jacob의 모든 랭킹 정보들을 찾는 경우



[그림 35] 사용자 이름 “Jacob”과 일치하는 모든 랭킹 정보를 찾은 결과

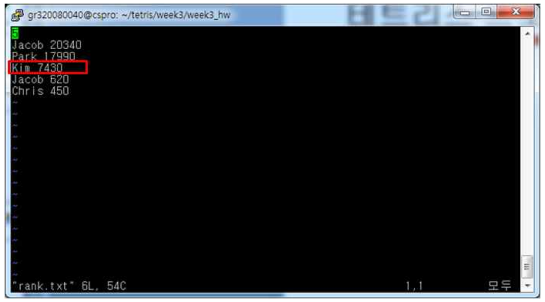
- park을 입력하여 랭킹 정보를 찾지만, 그 정보들이 존재하지 않는 경우



[그림 36] 존재하지 않는 사용자 이름 “park”과 일치하는 모든 랭킹 정보를 찾은 결과

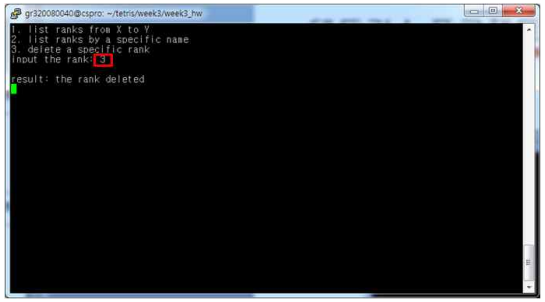
3. 기능 3

- Kim의 정보를 삭제할 예정



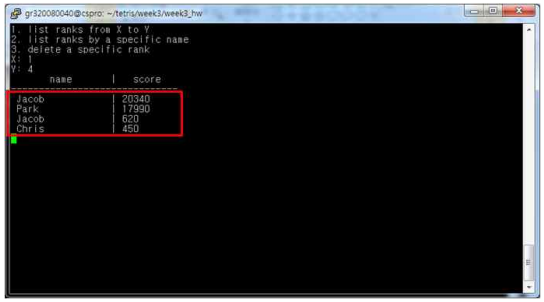
[그림 37] 삭제할 Kim의 랭킹 정보를 보여주는 입력 파일 rank.txt

- Kim의 정보를 삭제하기 위해 3을 입력



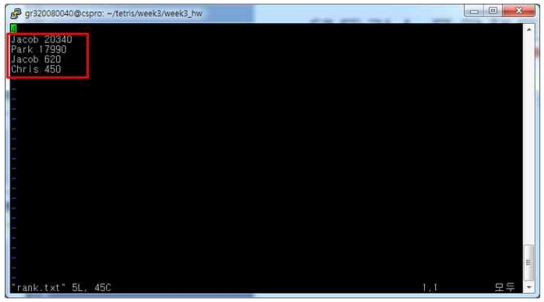
[그림 38] Kim의 정보를 삭제 하기 위해 3을 입력하고, 랭킹이 삭제된 결과

- Kim의 랭킹 정보가 삭제된 것을 확인



[그림 39] Kim의 정보가 삭제된 결과를 확인

- 프로그램 종료 후 rank.txt에서 Kim의 랭킹 정보가 삭제되었는지 확인



[그림 40] Kim의 정보가 삭제된 결과를 rank.txt에서 확인

- 존재하지 않는 랭킹 정보 삭제 시, 화면에 메시지 “search failure: the rank not in the list”를 출력

[그림 41] 존재하지 않는 랭킹을 삭제 하려고 할 때의 결과

7-3. 결과 보고서

아래의 사항을 작성하여 다음 실험 시간에 제출하시오.

1. 실험시간에 작성한 랭킹 시스템의 자료구조와 랭킹 시스템의 각 기능에 대한 알고리즘을 요약하여 기술하시오. 본인이 선택한 랭킹 시스템을 구현하기 위한 자료구조가 왜 효율적인지 시간 및 공간 복잡도를 통해 보이고, 설명하시오.
2. 본 실험 및 숙제를 통해 습득한 내용을 기술하시오.

실험 PRJ-1 2주차 랭킹 시스템 예비보고서

전공: 학년: 학번: 이름

실험 PRJ-1 2주차 랭킹 시스템 결과보고서

전공: 학년: 학번: 이름