

---

<컴퓨터학 실험 I>

## 테트리스 프로젝트 2주차

# 목차

---

- 테트리스 프로젝트 2주차 목표 - 랭킹 시스템
- 테트리스 프로젝트 2주차 구현 결과
- 테트리스 프로젝트 2주차 구현 프로그램 및 Flow chart
  - 랭킹 정보를 위해 사용되는 structure
  - 구현내용
  - 랭킹 시스템 예제
- 테트리스 프로젝트 2주차 실습 평가
- 테트리스 프로젝트 2주차 숙제
- 테트리스 프로젝트 2주차 결과보고서
- 테트리스 프로젝트 3주차 예비보고서

# 테트리스 프로젝트 2주차 목표

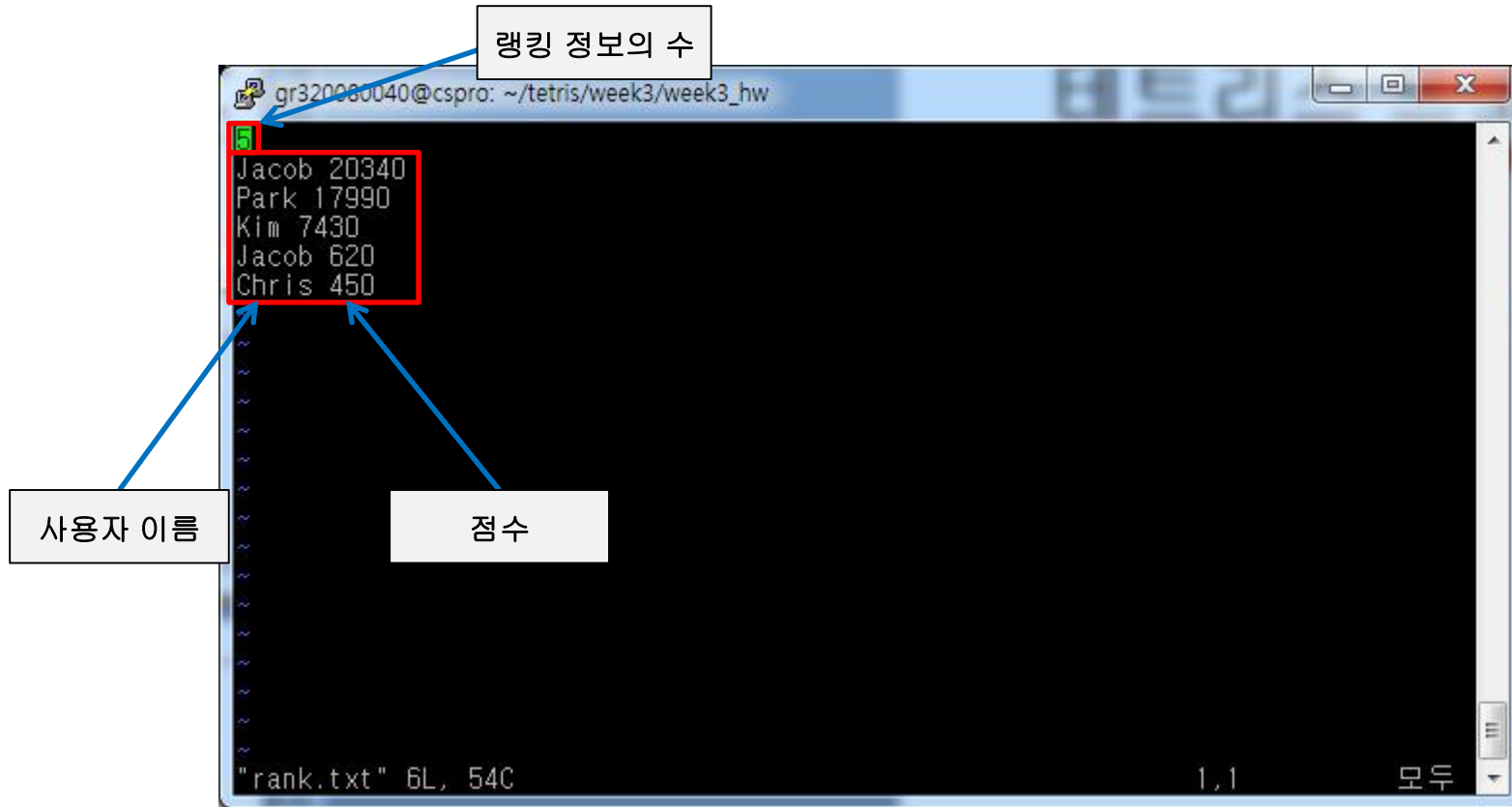
---

## □ 랭킹 시스템(Ranking System)

- 테트리스 게임을 play한 후(game over시), 사용자 이름을 입력받고, 사용자 이름과 점수로 구성되는 랭킹 정보를 등록하고 확인할 수 있는 **랭킹 시스템**을 구현한다.
- 사용자 이름과 점수로 구성된 랭킹 정보들은 다음 번에 테트리스 게임을 실행할 때도 유지하기 위해 rank.txt 파일에 기록된다.
- 테트리스 게임을 실행할 때, rank.txt를 읽어 들여, **자료구조를 구축**하고 랭킹 정보를 프로그램 수행 중에도 유지한다.
- 테트리스 게임이 종료(game over)되면, 사용자 이름을 입력 받고 랭킹 정보(사용자 이름과 점수)를 랭킹 정보를 저장하고 있는 자료구조에 추가한다.

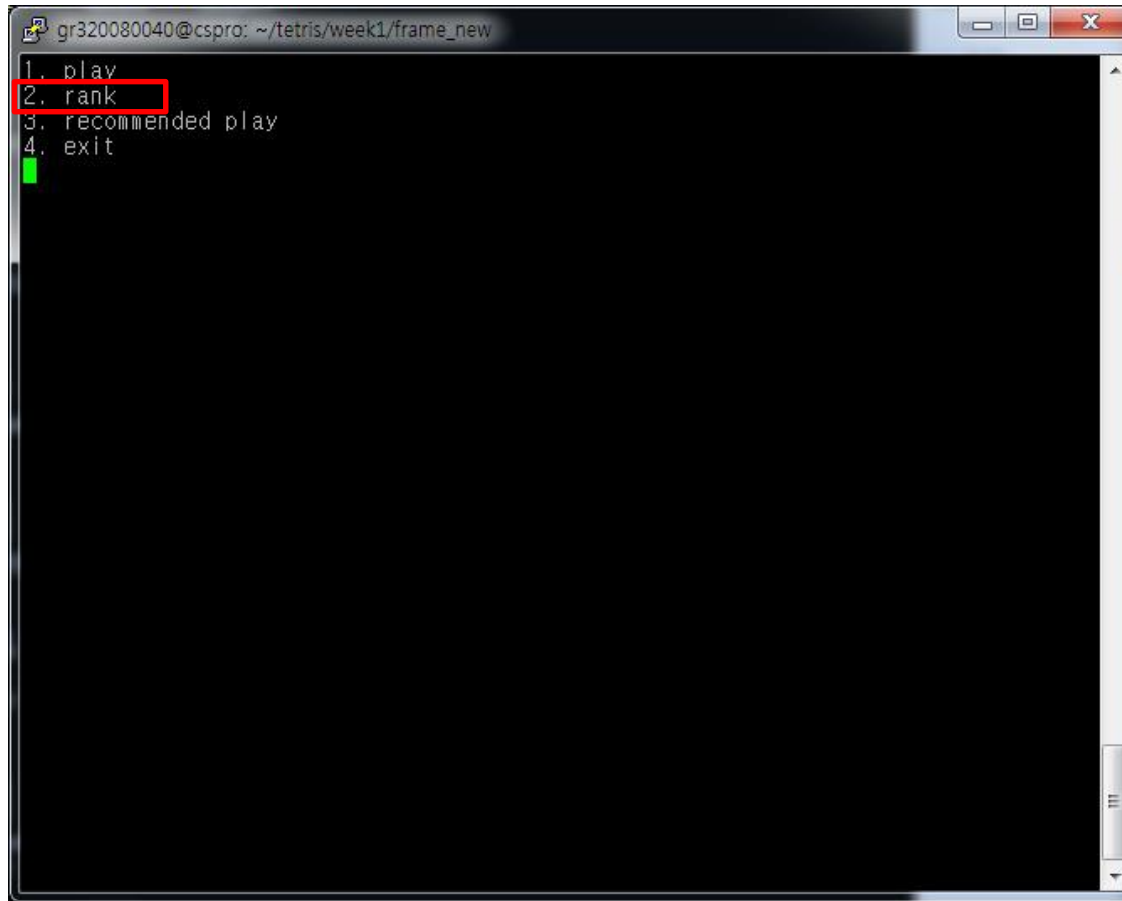
## 테트리스 프로젝트 2주차 실습 구현결과(1/12)

- 입력파일 예제(입력 파일 양식은 학생들마다 달라질 수 있음)
  - rank.txt



# 테트리스 프로젝트 2주차 실습 구현결과(2/12)

- 랭킹을 확인하기 위해 2를 입력



```
gr320080040@cspro: ~/tetris/week1/frame_new
1. play
2. rank
3. recommended play
4. exit
█
```

A terminal window with a black background and white text. The title bar shows the user 'gr320080040' and the directory '~/tetris/week1/frame\_new'. The terminal displays a menu with four options: '1. play', '2. rank', '3. recommended play', and '4. exit'. The option '2. rank' is highlighted with a red rectangular box. Below the menu, a green cursor is visible on the first line of the next line.

# 테트리스 프로젝트 2주차 실습 구현결과(3/12)

## □ 출력1

- 1위~5위까지 랭킹을 확인하기 위해 1과 5를 입력하고 점수 순으로 1위부터 5위까지 랭킹 정보를 확인.

```
gr320080040@csp: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
X: 1
Y: 5
-----
name | score
-----
Jacob | 20340
Park  | 17990
Kim   | 7430
Jacob | 620
Chris | 450
```

주어진 메뉴(1, 2, 3)에 대해 키보드로부터 입력 받아 선택된 메뉴가 화면에 출력되지 않고, 바로 두 개의 랭킹(X, Y)를 입력 받는 문구가 출력되는 이유는 메뉴 선택 시 wgetch() 함수를 사용하기 때문에 scanf() 함수를 사용했을 때와 달리 화면에 그 결과가 출력되지 않는다.  
(이 슬라이드 이후 메뉴 선택하는 모든 출력 예제에서 이와 같은 현상을 관찰할 수 있다)

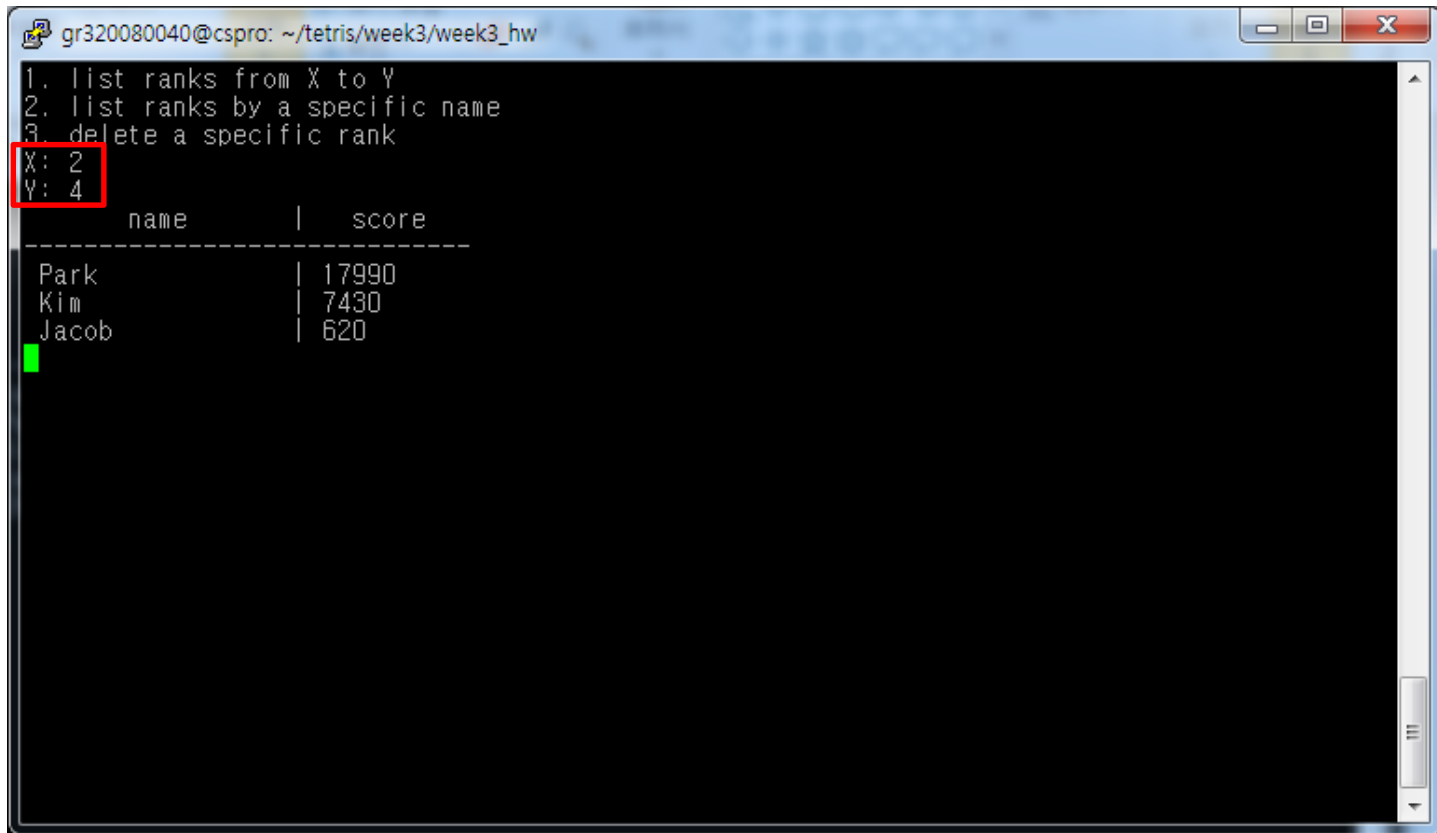
사용자 이름

점수

# 테트리스 프로젝트 2주차 실습 구현결과(4/12)

## □ 출력2

- 2위~4위까지 랭킹을 확인하기 위해 2와 4를 입력하고 점수 순으로 2위부터 4위까지 랭킹 정보를 확인.



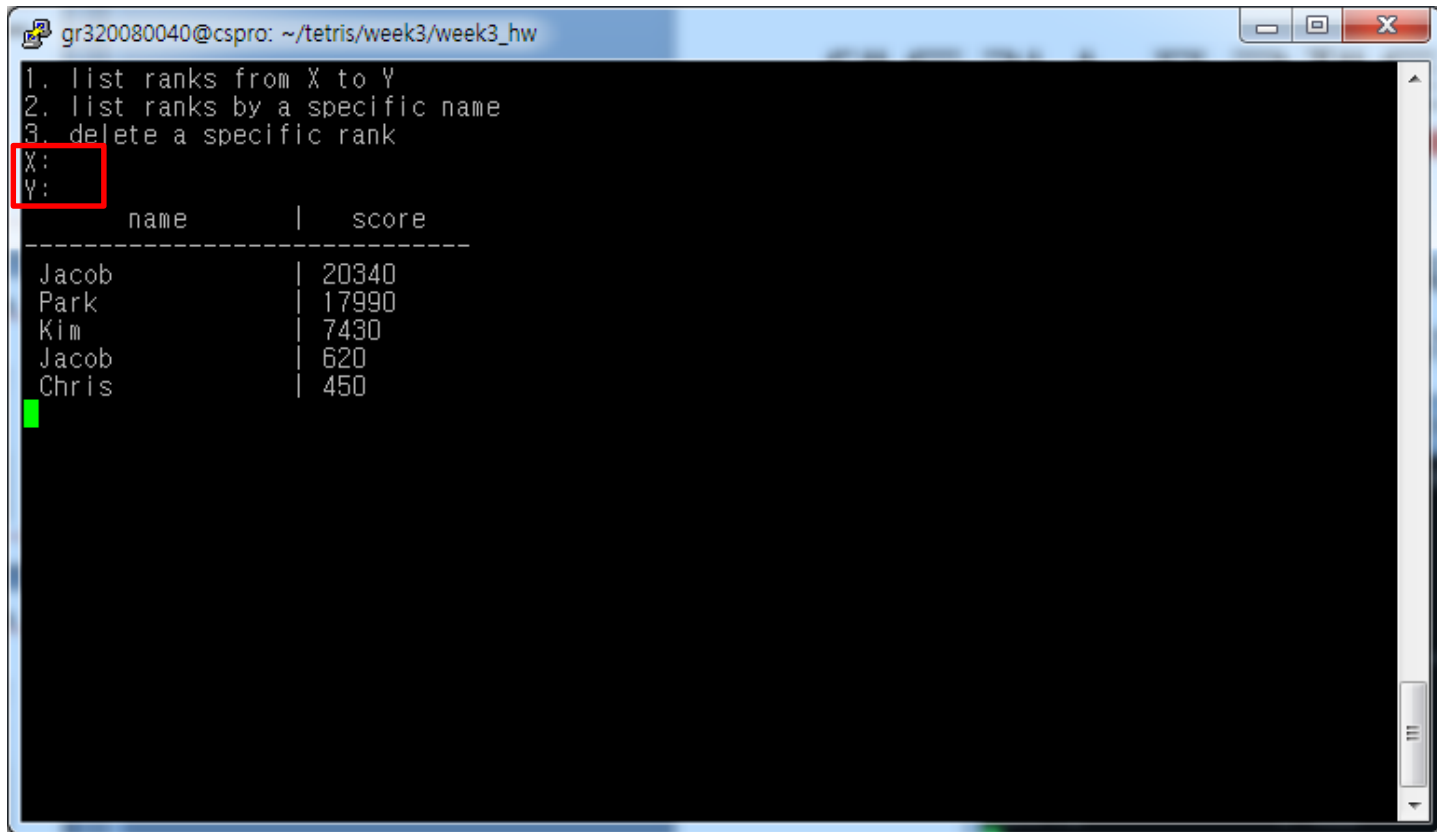
```
gr320080040@csp: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
X: 2
Y: 4
-----
name | score
-----
Park | 17990
Kim  | 7430
Jacob| 620
█
```

The terminal window shows the execution of a command to list ranks from X to Y. The user input 'X: 2' and 'Y: 4' is highlighted with a red box. The output displays a table of ranks with names and scores, sorted in descending order of score. The table shows three entries: Park (17990), Kim (7430), and Jacob (620). A green cursor is visible on the line following the last entry.

# 테트리스 프로젝트 2주차 실습 구현결과(5/12)

## □ 출력3

- X와 Y 모두 입력하지 않았을 경우.
- 점수 순으로 1위~5위까지 모든 순위 출력



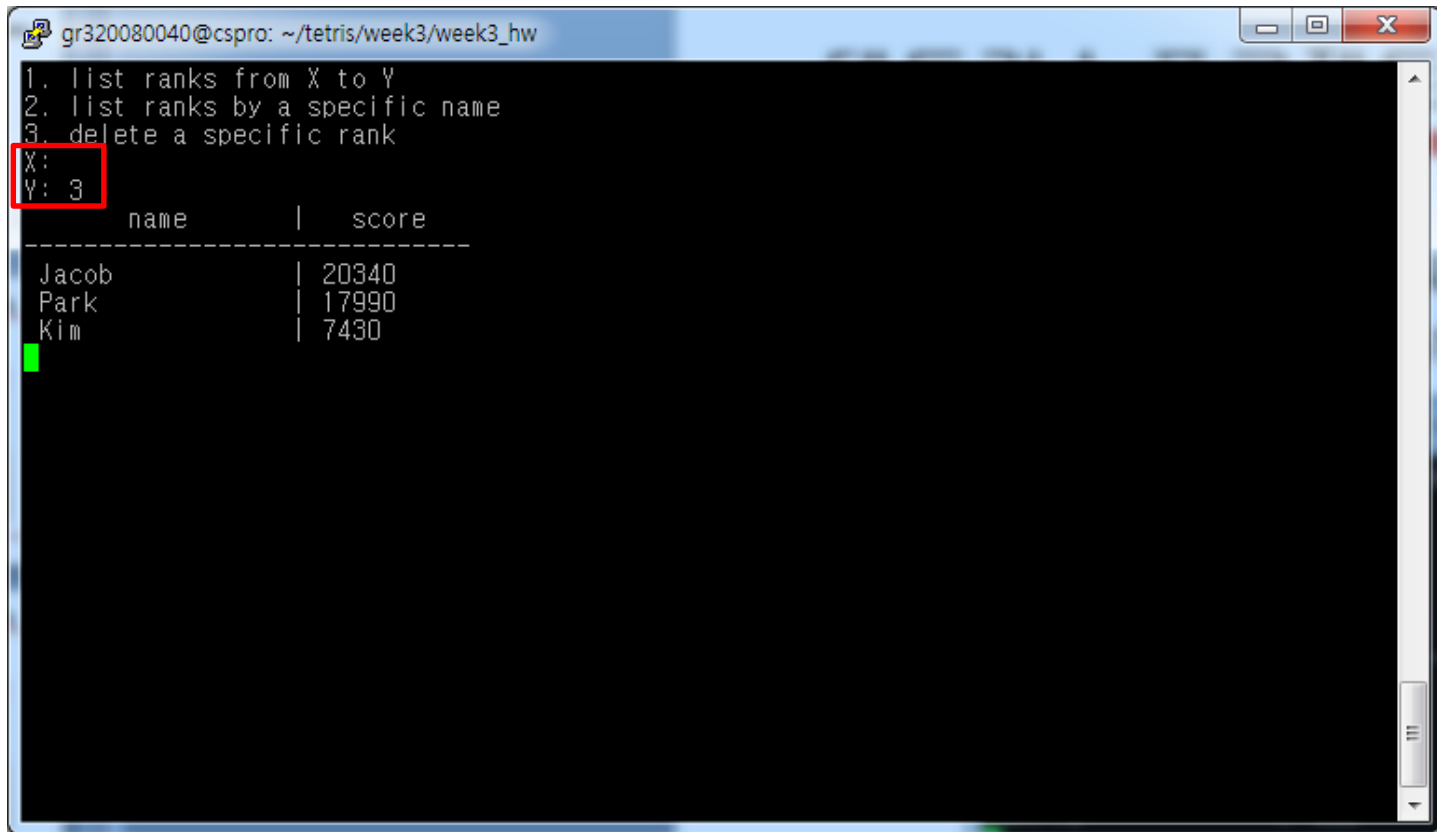
```
gr320080040@cspiro: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
X:
Y:
name | score
-----
Jacob | 20340
Park  | 17990
Kim   | 7430
Jacob | 620
Chris | 450
█
```



# 테트리스 프로젝트 2주차 실습 구현결과(6/12)

## □ 출력4

- X를 입력 안 하고, Y로 3을 입력하는 경우.
- 점수 순으로 1위~3위까지 출력



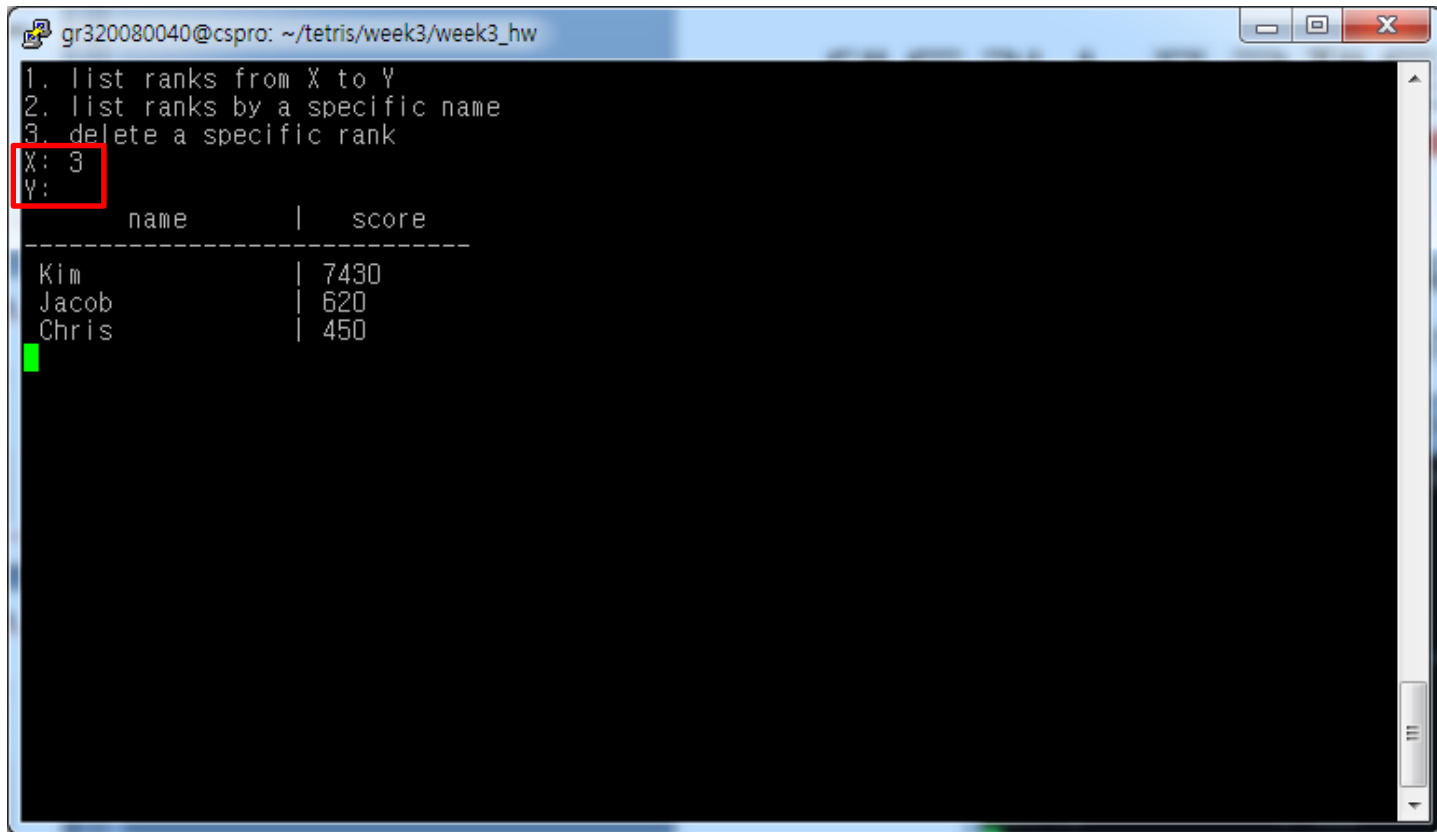
```
gr320080040@csp: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
X:
Y: 3
name | score
-----
Jacob | 20340
Park  | 17990
Kim   | 7430
```

The image shows a terminal window with a blue title bar. The window title is "gr320080040@csp: ~/tetris/week3/week3\_hw". The terminal content shows a list of three commands: "1. list ranks from X to Y", "2. list ranks by a specific name", and "3. delete a specific rank". Below these, the user has entered "X:" and "Y: 3", which are highlighted by a red rectangle. The output is a table with two columns: "name" and "score", separated by a vertical line. The table is preceded by a dashed line and followed by a green cursor. The table contains three rows: "Jacob | 20340", "Park | 17990", and "Kim | 7430".

# 테트리스 프로젝트 2주차 실습 구현결과(7/12)

## □ 출력5

- X로 3을 입력하고, Y를 입력 안 하는 경우.
- 점수 순으로 3위~5위까지 출력



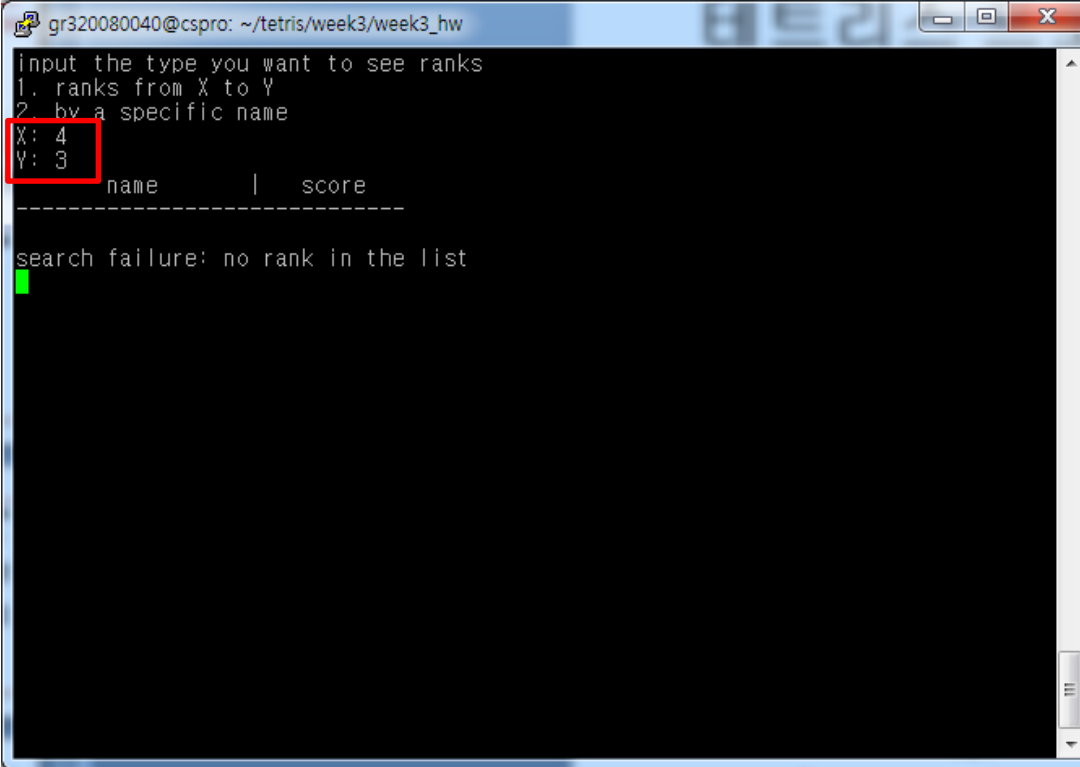
```
gr320080040@csp: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
X: 3
Y:
-----
name | score
-----
Kim  | 7430
Jacob | 620
Chris | 450
```

The terminal window shows the execution of a command to list ranks from X to Y. The user has entered 'X: 3' and 'Y:' (with Y being empty). The output displays a table of ranks from 3rd to 5th place, sorted by score. The table has two columns: 'name' and 'score'. The data rows are Kim (7430), Jacob (620), and Chris (450). A green cursor is visible at the bottom left of the terminal window.

# 테트리스 프로젝트 2주차 실습 구현결과(8/12)

## □ 출력6

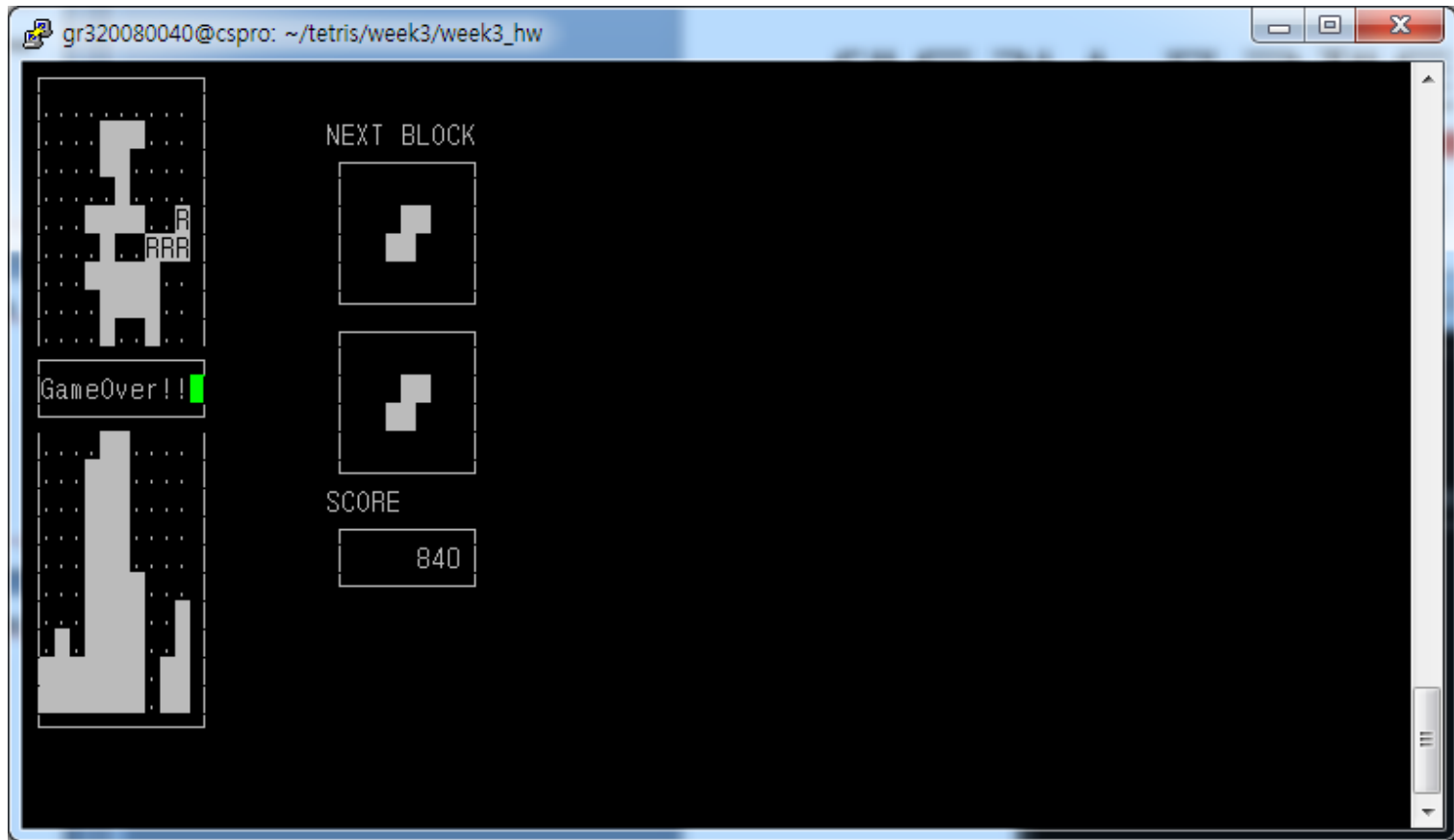
- X로 4을 입력하고, Y로 3을 입력 하는 경우.
- 어떤 랭킹도 출력하지 않고, “search failure: no rank in the list”라는 메시지 출력.



```
gr320080040@csp: ~/tetris/week3/week3_hw
input the type you want to see ranks
1. ranks from X to Y
2. by a specific name
X: 4
Y: 3
name      |  score
-----
search failure: no rank in the list
```

# 테트리스 프로젝트 2주차 실습 구현결과(9/12)

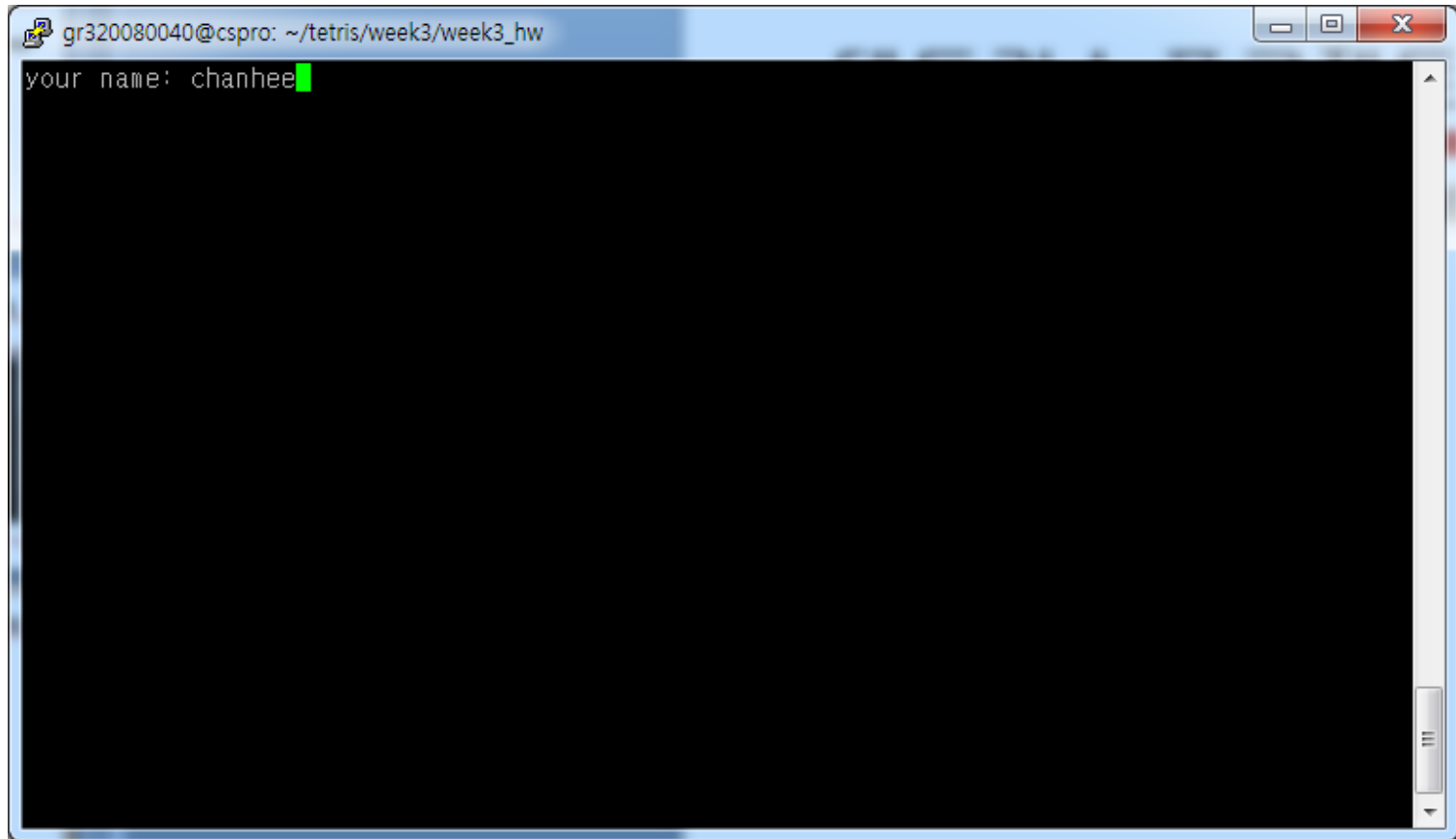
- 새로운 랭킹 정보를 등록하기 위해 게임 종료



# 테트리스 프로젝트 2주차 실습 구현결과(10/12)

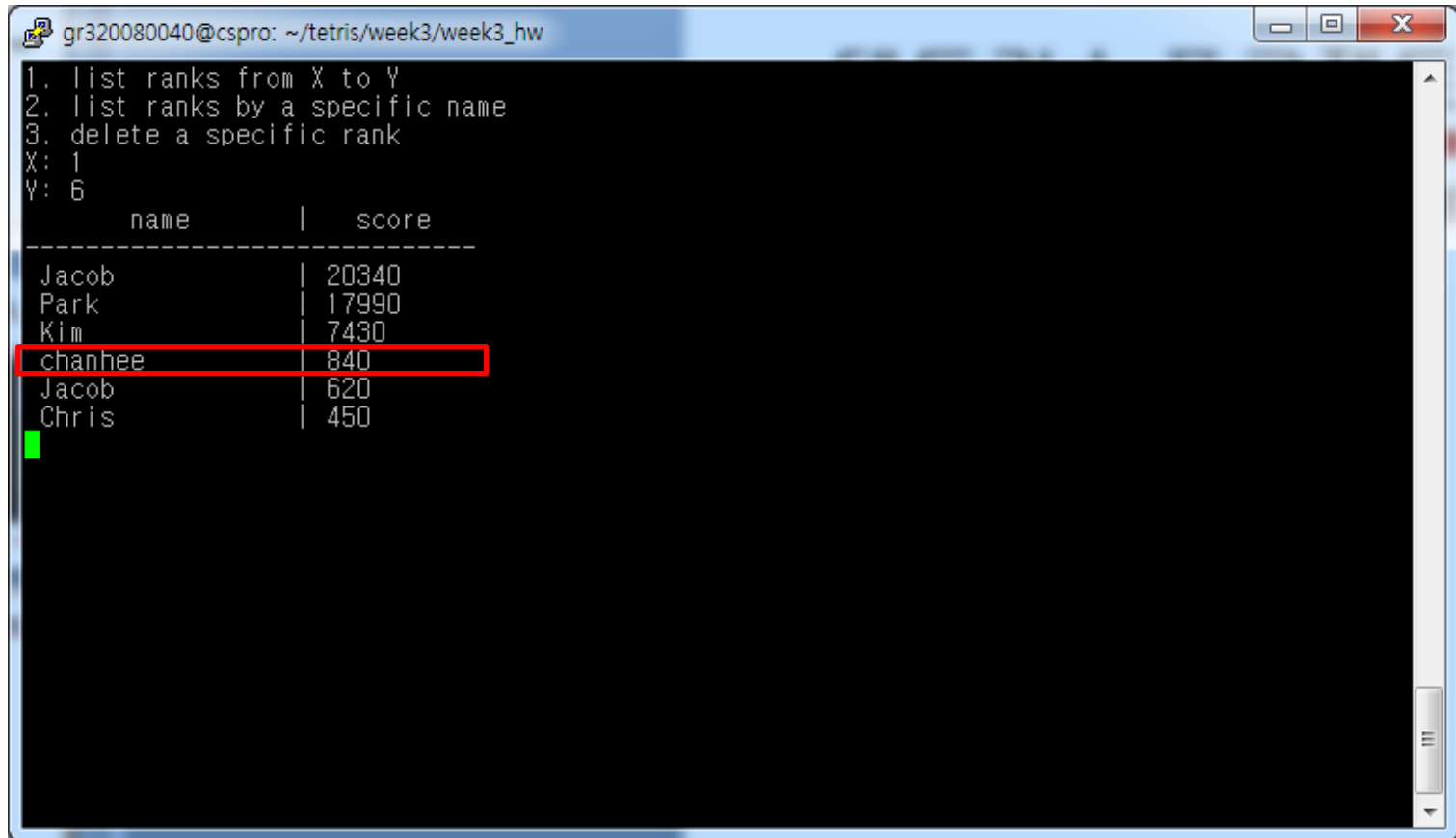
---

- 사용자 이름 입력(예제에서는 “chanhee”)



# 테트리스 프로젝트 2주차 실습 구현결과(11/12)

- chanhee의 랭킹 정보를 확인

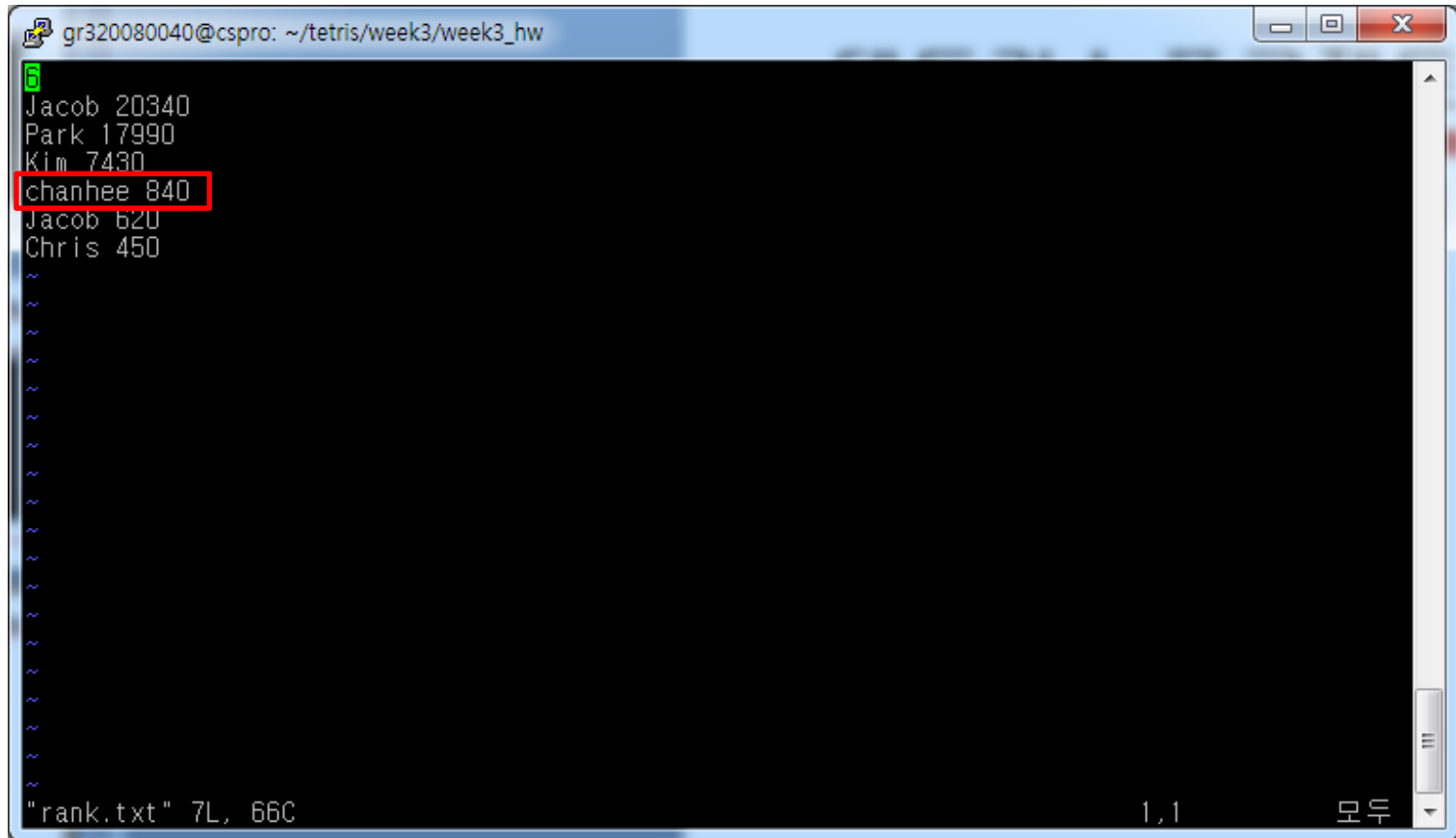


A terminal window titled 'gr320080040@cspiro: ~/tetris/week3/week3\_hw' displays a list of Tetris scores. The list is formatted with columns for 'name' and 'score', separated by a vertical line. The entries are: Jacob (20340), Park (17990), Kim (7430), chanhee (840), Jacob (620), and Chris (450). The entry for 'chanhee' is highlighted with a red rectangular box. The terminal also shows a list of commands: 1. list ranks from X to Y, 2. list ranks by a specific name, and 3. delete a specific rank, followed by 'X: 1' and 'Y: 6'.

```
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
X: 1
Y: 6
```

name	score
Jacob	20340
Park	17990
Kim	7430
chanhee	840
Jacob	620
Chris	450

## 테트리스 프로젝트 2주차 실습 구현결과(12/12)



---

# 테트리스 프로젝트 2주차

## 구현 프로그램 설명, Flow Chart 및 함수표

Function() : 구현할 함수



# 랭킹 시스템 예제(1/7)

---

## □ 가정

- 가정1: 랭킹 시스템을 구현하기 위한 자료구조로 linked list를 사용.
- 가정2: rank.txt에는 빈 파일이다.

## □ 테트리스 게임 play 전.

Empty Head node

## □ 1번째 테트리스 게임 play 이후, 사용자 cs2011이 1500점 획득하고 game over.

- 사용자 cs2011, score 1500을 갖는 head node가 생성된다.

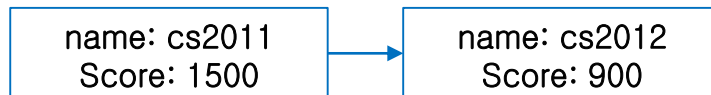
name: cs2011  
Score: 1500

## 랭킹 시스템 예제(2/7)

- 2번째 테트리스 게임 play 이후, 사용자 cs2012가 900점 획득하고 game over.
  - Link를 따라가면서, score를 비교하여, 새로운 node를 삽입할 적절한 위치를 탐색한다.

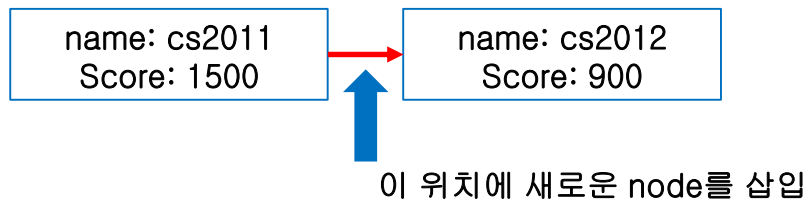


- 사용자가 cs2012, score로 900을 갖는 새로운 node 삽입한다. 그 결과로 정렬된 linked list가 생성된다.

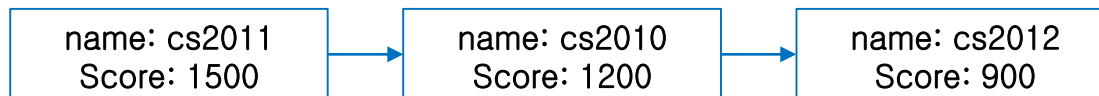


## 랭킹 시스템 예제(3/7)

- 3번째 테트리스 게임 play 이후, 사용자 cs2010이 1200점 획득하고 game over.
- Link를 따라가면서, score를 비교하여, 새로운 node를 삽입할 적절한 위치를 탐색한다.



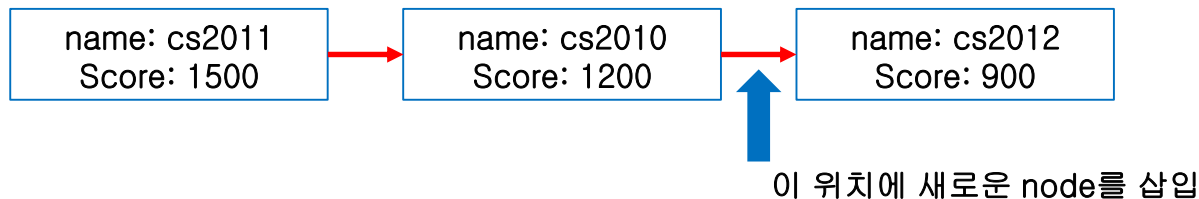
- 사용자가 cs2010, score로 1200을 갖는 새로운 node 삽입한다. 그 결과로 정렬된 linked list가 생성된다.



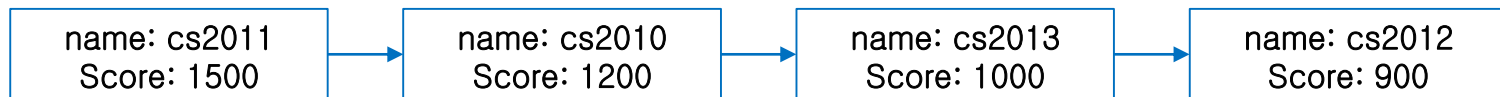
→ : 적절한 위치를 찾기 위해 탐색된 Link

## 랭킹 시스템 예제(4/7)

- 4번째 테트리스 게임 play 이후, 사용자 cs2013이 1000점 획득하고 game over.
  - Link를 따라가면서, score를 비교하여, 새로운 node를 삽입할 적절한 위치를 탐색한다.



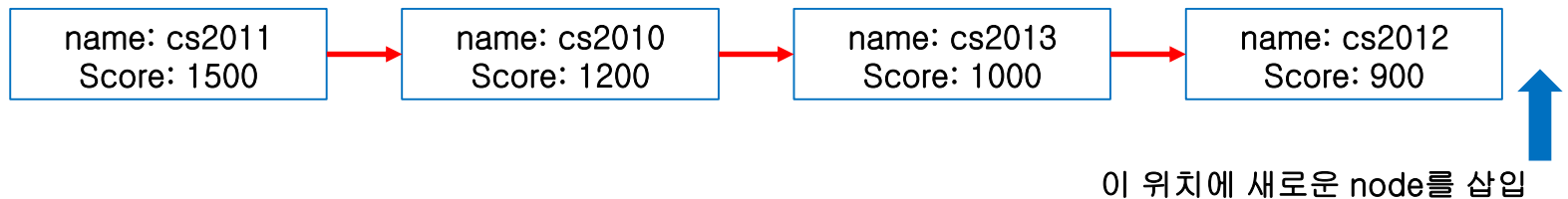
- 사용자가 cs2013, score로 1000을 갖는 새로운 node 삽입한다. 그 결과로 정렬된 linked list가 생성된다.



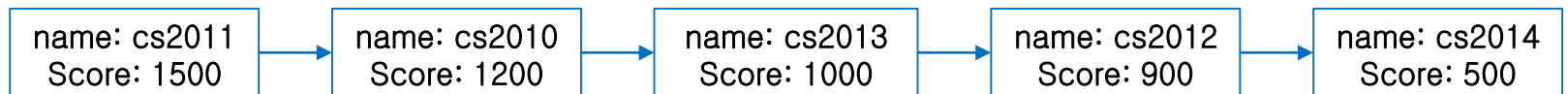
→ : 적절한 위치를 찾기 위해 탐색된 Link

## 랭킹 시스템 예제(5/7)

- 5번째 테트리스 게임 play 이후, 사용자 cs2014이 500점 획득하고 game over.
  - Link를 따라가면서, score를 비교하여, 새로운 node를 삽입할 적절한 위치를 탐색한다.



- 사용자가 cs2014, score로 500을 갖는 새로운 node 삽입한다. 그 결과로 정렬된 linked list가 생성된다.



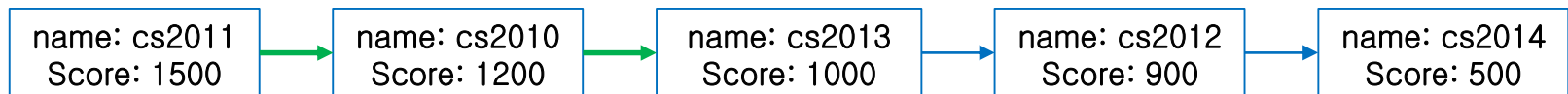
→ : 적절한 위치를 찾기 위해 탐색된 Link

# 랭킹 시스템 예제(6/7)

## □ 랭킹 정보 출력 시 예제

- 랭킹 정보를 유지하기 위한 linked list는 정렬된 상태이므로 출력하길 원하는 랭킹의 범위(2개의 정수)를 입력 받았을 때, 헤드 노드(head node)부터 링크를 따라가면서 원하는 수 만큼의 정보를 출력하면 된다.

## □ 예) 2와 3을 입력했을 때(2위~3위까지 순위를 알고 싶을 때)

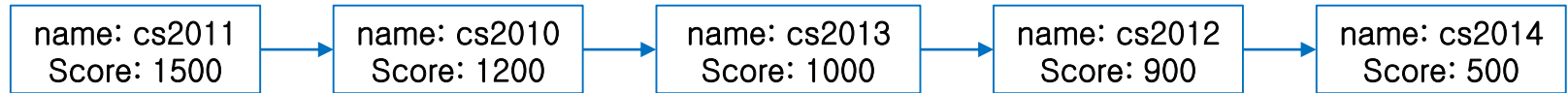


→ : 랭킹 정보를 출력하기 위해 탐색된 Link

탐색 후 두 개의 node 정보 출력

# 랭킹 시스템 예제(7/7)

- 랭킹 정보들을 저장하고 있는 linked list



- rank.txt의 예  
rank.txt

5		랭킹정보의 개수
cs2011	1500	
cs2010	1200	랭킹정보( 사용자 이름, 점수)
cs2013	1000	
cs2012	900	
cs2014	500	

# 테트리스 프로그램 전체 흐름과의 관계

## □ 키입력에 대한 동작 - 메뉴 2번

### ● `createRankList()`

- 테트리스 프로그램 시작 시 rank.txt로부터 ranking 정보를 입력 받아 지정된 자료구조를 구축한다.

### ● `rank()`

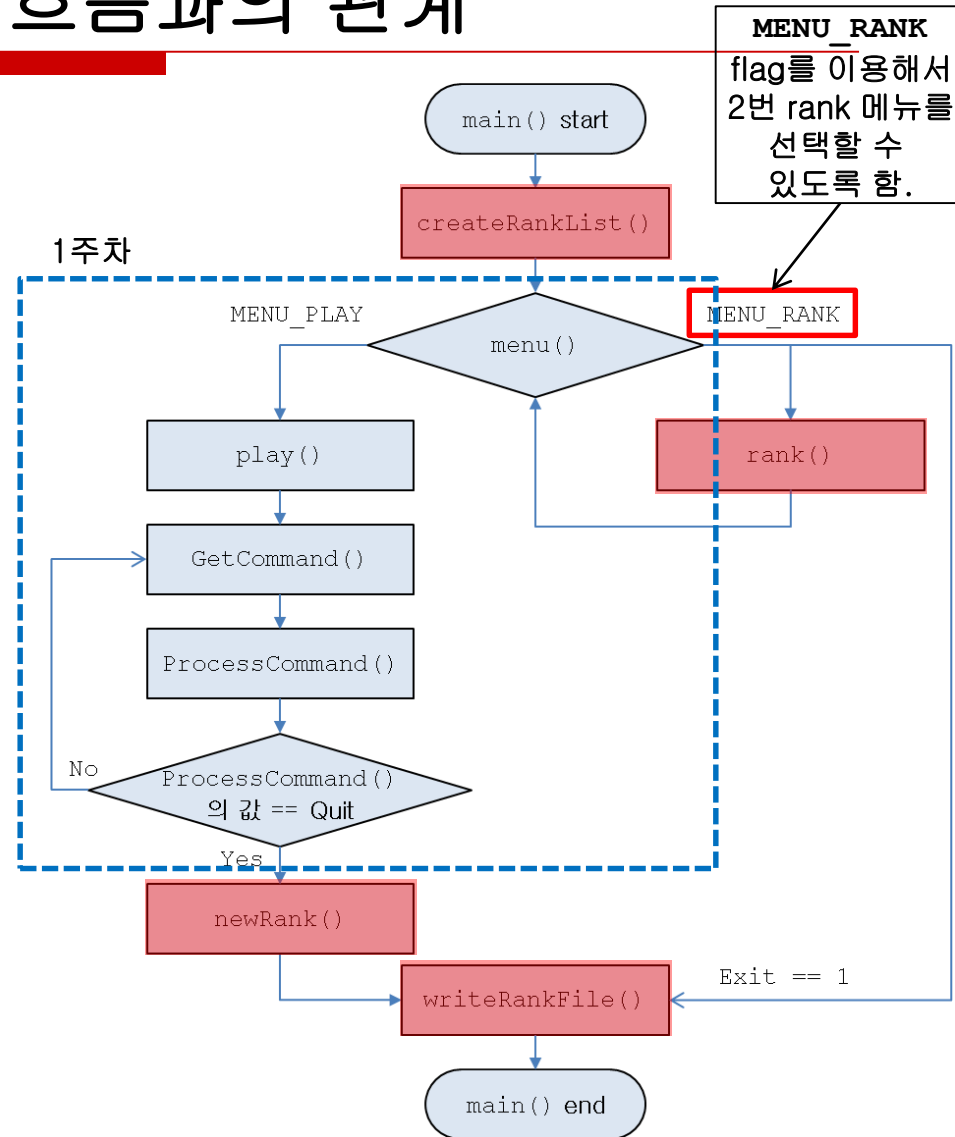
- 정수 2개, x, y( $x \leq y$ )를 입력 받고 랭킹 정보(x위~y위)를 화면에 출력한다.

### ● `newRank(int score)`

- 게임 종료(gameover) 시, 사용자의 이름을 입력 받고, 사용자 이름과 score를 자료구조에 추가, 저장한다.

### ● `writeRankFile()`

- 추가된 랭킹 정보가 있으면, 새로운 정보를 rank.txt에 기록하고, 추가된 정보가 없으면 그대로 종료한다.





# 테트리스 프로젝트 2주차 구현 함수

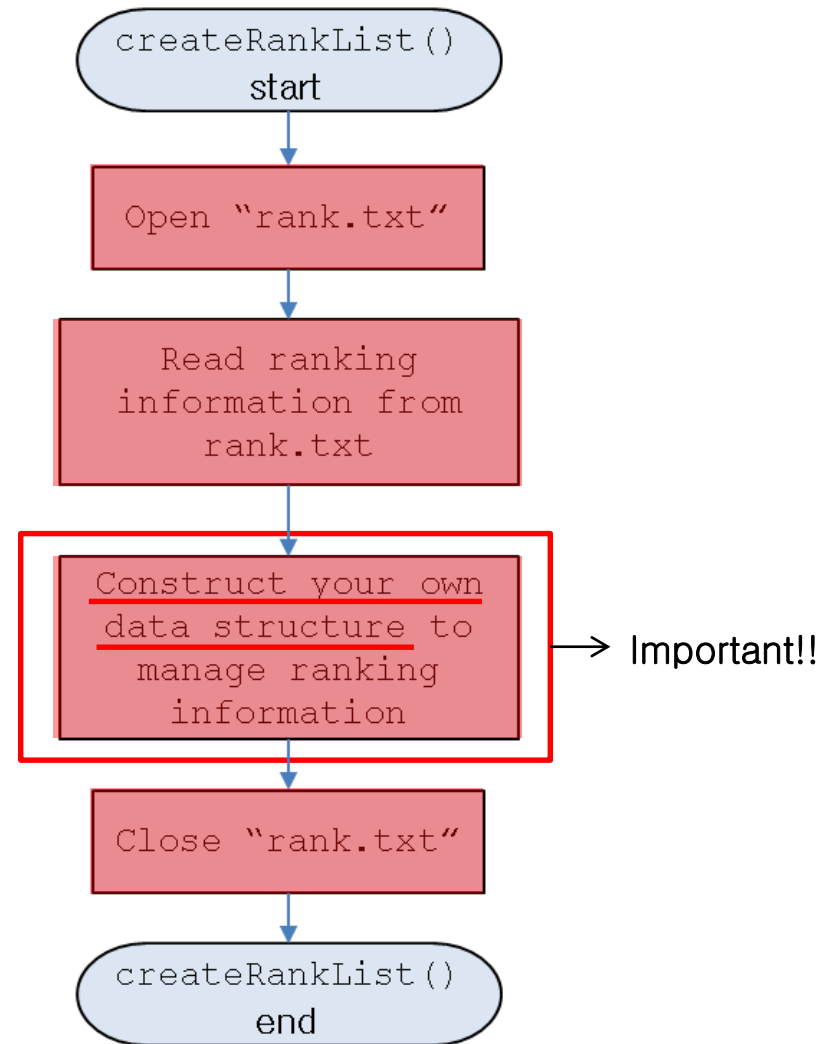
---

- 테트리스 프로젝트 2주차 실습에서는 앞의 flow chart에서 명시된 4가지 함수들을 구현한다.
- 각 함수에서 대한 모든 flow chart는 예제에서 설명된 자료구조인 linked list를 바탕으로 한다.

## 2주차 구현 함수 - createRankList()

### □ createRankList()

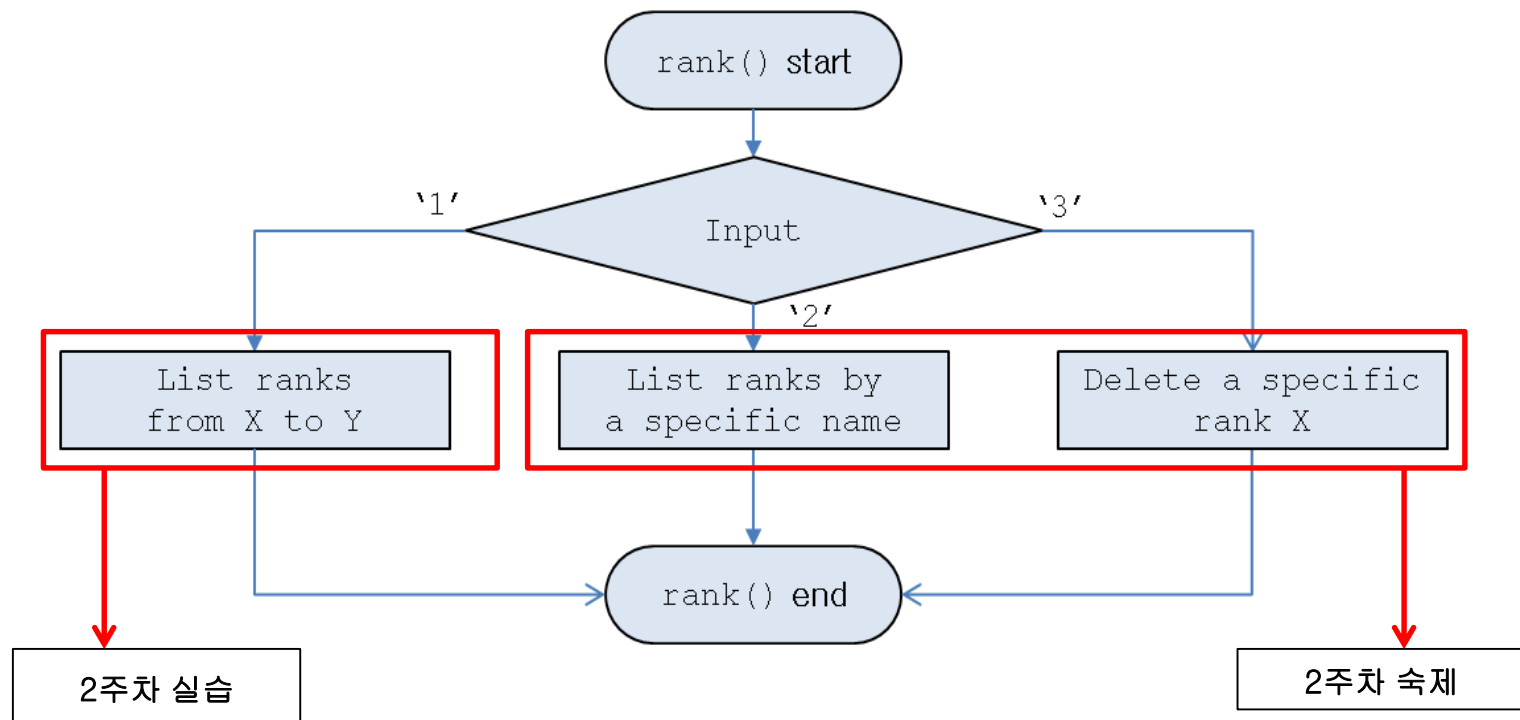
- Input 파일인 “rank.txt”에서 랭킹 정보를 읽어 들여 랭킹 정보를 저장하는 자료구조를 이용하여 랭킹 목록을 만든다.
- “rank.txt” 파일을 연다.
- “rank.txt” 파일에서 랭킹 정보들을 읽어 들인다.
- 랭킹 정보들을 랭킹 정보를 저장 및 유지하기 위해 선택된 자료구조에 저장하면서 랭킹 목록을 만든다.
- “rank.txt” 파일을 닫는다.



## 2주차 실습 구현 함수 - rank () (1/3)

### □ 함수 rank ()

- 다음 flow chart에서 rank () 에서 수행되는 기본 기능들을 확인할 수 있다.

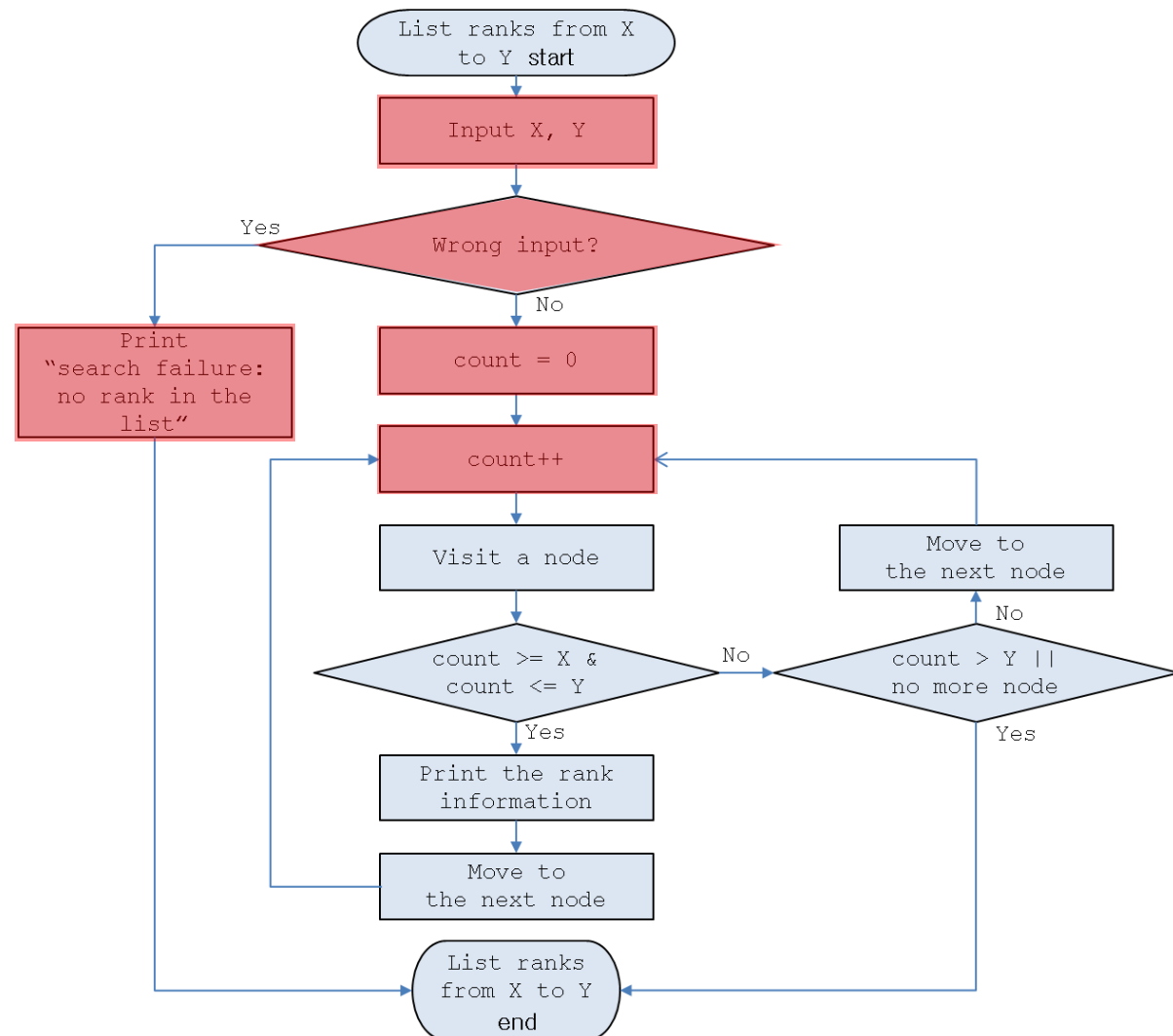


## 2주차 실습 구현 함수 - rank () (2/3)

### □ List ranks

from X to Y

- 점수순으로 X~Y위 까지 출력하는 기능
- 정수 X, Y 를 입력 받는다.
- X, Y가 잘못된 input 인지 체크한다.
- 잘못된 input이라면, 메시지 “search failure: no rank in the list”를 출력한다.
- 변수 count를 0으로 초기화한다.
- 변수 count값을 증가한다.

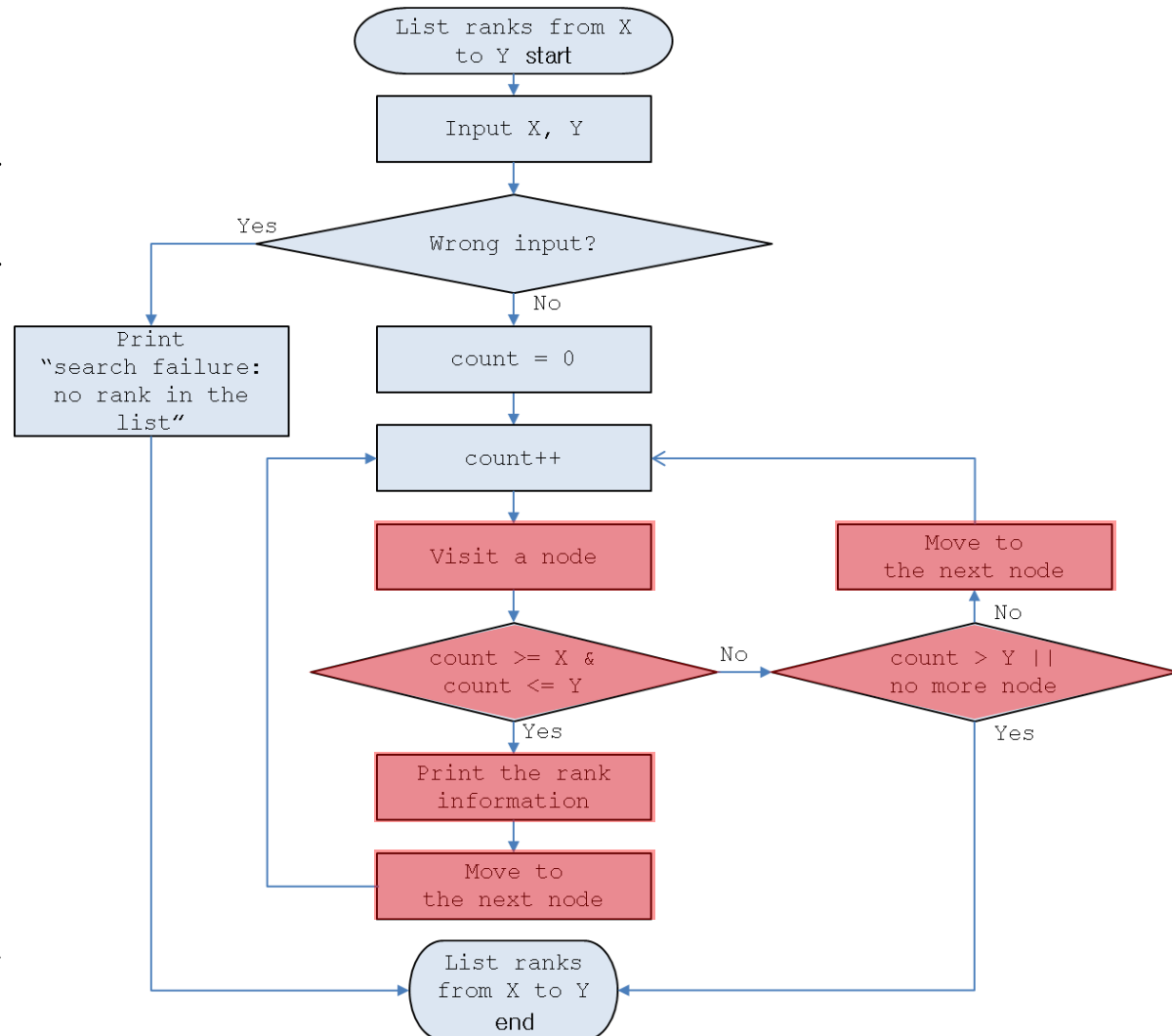


## 2주차 실습 구현 함수 - rank () (3/3)

### □ List ranks

from X to Y

- 랭킹 정보를 저장하고 있는 자료구조의 각 노드를 방문한다.
- 만약 count가 X보다 크거나 같고, Y보다 작거나 같은지 체크한다.
- 그렇다면, 랭킹 정보를 출력하고, 다음노드로 이동한다.
- 그렇지 않다면, 범위를 벗어나거나, 노드가 더 있는지 확인하고 다음노드로 이동하거나 종료한다.



## 2주차 실습 구현 함수 - newRank ()

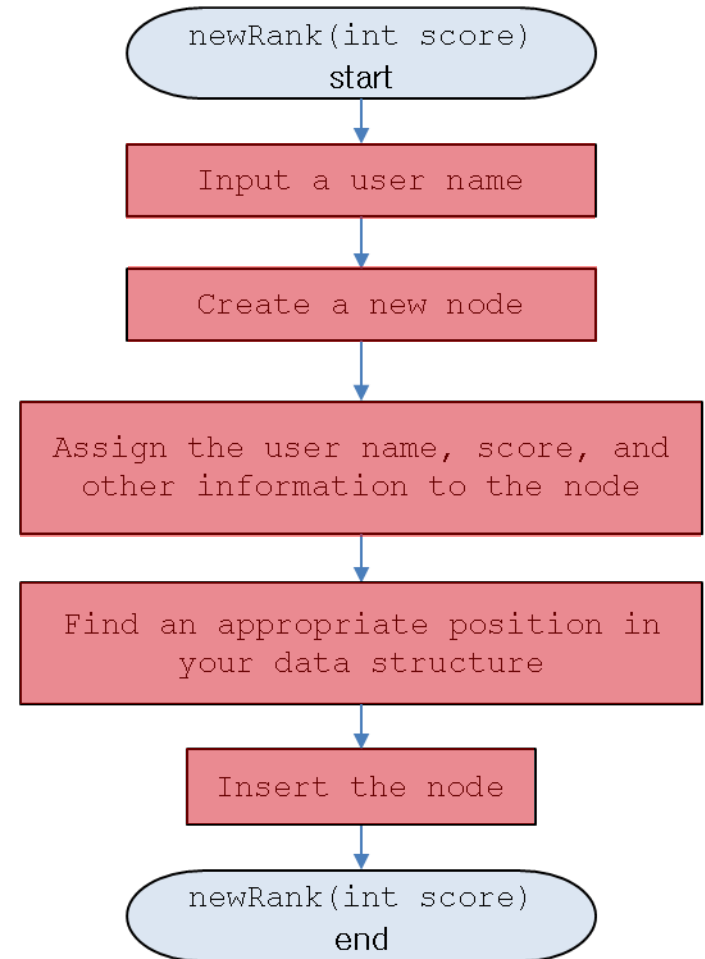
### □ newRank ()

- Input

- int score: gameover시,  
얻은 score

- Gameover시 호출되는 함수로,  
사용자 이름을 입력받고, score  
와 함께 랭킹 정보를 구성하여 선  
택된 자료구조로 구성된 랭킹 목  
록의 적절한 위치에 해당 노드를  
삽입한다.

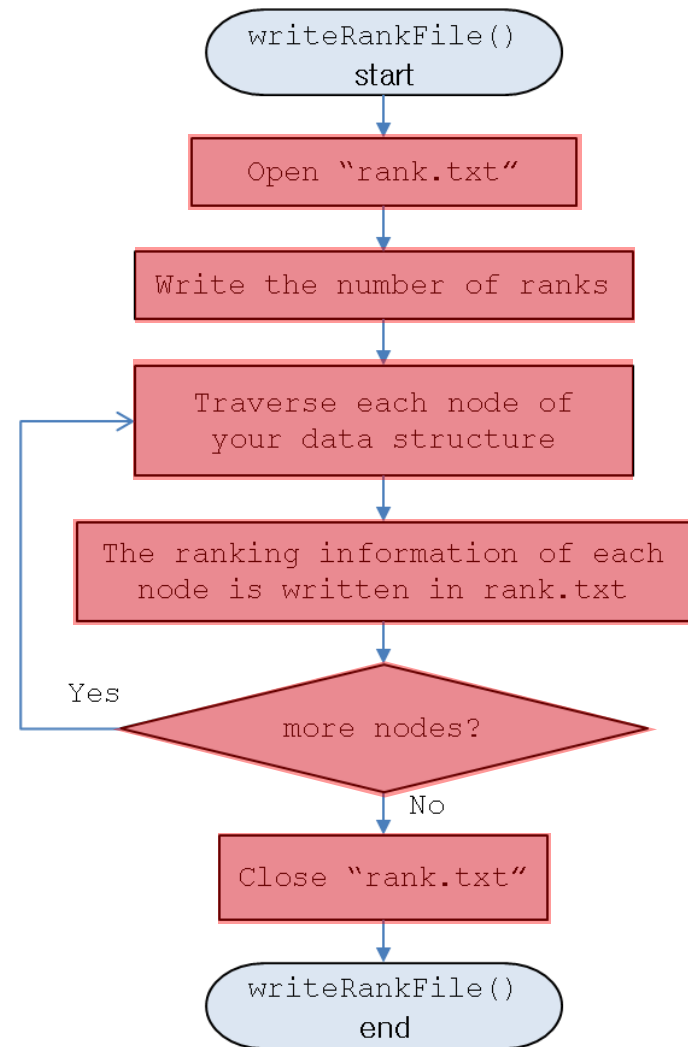
- 사용자 이름을 입력받는다.
- 사용자 이름과 플레이 후 얻은  
score를 이용해서 새로운 노드를  
생성한다.
- 랭킹 정보를 저장하는 자료구조  
에서 적절한 위치를 찾고, 생성된  
새로운 노드를 삽입한다.



## 2주차 실습 구현 함수 - writeRankFile()

### □ writeRankFile()

- 추가된 랭킹 정보가 있으면, 새로운 정보를 “rank.txt”에 기록하고, 추가된 정보가 없으면 그대로 종료한다.
- Input 파일인 “rank.txt”를 연다.
- 랭킹 정보들의 수를 “rank.txt”에 기록한다.
- 랭킹정보들을 저장한 각 정보를 탐색하여, “rank.txt”파일에 랭킹 정보를 기록한다.
- 탐색할 노드가 더 있는지 체크하고, 탐색할 노드가 있다면 다음 노드로 이동하고, 그렇지 않으면, 파일을 닫고 종료한다.



# 랭킹 정보 저장을 위한 structure

---

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

## □ 랭킹 정보를 저장하기 위한 structure

- **char name[NAMELEN]**
  - 사용자의 이름(name)을 저장하기 위한 character array
- **int score**
  - 사용자의 점수를 저장하기 위한 정수형 score 변수
- “...”은 랭킹 정보들을 자료구조에 저장하기 위해서 다른 변수들을 추가할 수 있다는 것을 의미한다.
- NAMELEN은 상수로 16으로 tetris.h에 define되어 있음(수정가능).



## 구현내용(1/2)

---

- 메뉴 2번 rank가 랭킹 시스템으로써 동작할 수 있도록 프로그램을 구현한다.
- Flag `MENU_RANK`를 새롭게 define한다.
  - `#define MENU_RANK 2`

## 구현내용(2/2)

---

### □ 자료구조 선택 시 고려사항

- Game over가 된 후새로운 사용자 이름을 입력 받고, 사용자 이름과 테트리스 게임을 play한 점수를 기존에 존재하는 랭킹 리스트에 삽입 시 발생하는 시간 복잡도.
- 원하는 랭킹을 삭제 시 발생하는 시간 복잡도(고려사항).
- 구축된 자료구조에서 랭킹 정보 화면에 출력하기 위해, 원하는 범위의 순위를 추출 할 때, 정렬된 상태로 추출 가능한가의 여부와 시간 복잡도.

### □ 결정된 자료구조를 사용해서 위 flow chart의 4가지 함수, `createRankList()`, `rank()`, `newRank()`, `writeRankFile()`을 구현한다.

# (참고)NCURSES 라이브러리 함수

## NCURSES란

텍스트모드에서 Window, Panel, Menu, Mouse, Color등을 쉽게 사용할 수 있도록 도와주는 라이브러리입니다. curses라이브러리도 거의 같은 역할을 합니다. ncurses는 curses의 새로운 버전입니다. (new curses)

## NCURSES설치

NCURSES는 LINUX 라이브러리이며, 현재 최신 버전의 리눅스의 경우 gcc같은 개발프로그램 설치시 자동으로 설치됩니다.

## NCURSES를 사용하려면

```
#include <ncurses.h> //소스파일 안에 추가
```

컴파일시

```
gcc 소스파일네임 -lncurses
```

## NCURSES FUNCTIONS

### 설정 및 초기화 관련 함수

#### - initscr()

curses모드를 시작한다. curses를 사용하기 위해 반드시 써줘야 한다.

#### -endwin()

curses모드를 종료한다. curses모드를 사용하고 종료할 때 반드시 써줘야한다.

#### -raw() and cbreak()

Line buffering을 사용할지의 유무를 설정한다. 사용할 경우 cbreak, 사용하지 않고자할 경우 raw를 사용하면된다. 보통 기본설정은 사용을 하는 것으로 되어있다.

#### -echo() and noecho()

사용자로부터 입력을 받은 문자를 출력할지 여부를 결정한다. echo는 출력

noecho는 출력을 하지 않는다. (getch 함수 사용시)

#### -keypad()

F1, F2, 방향키 등과 같이 특수한 키들을 사용할 수 있게 해준다.

```
keypad(stdscr, TRUE);
```

```
KEY_F(1)~ KEY_F(12)
```

```
KEY_LEFT, KEY_RIGHT, KEY_UP, KEY_DOWN
```

#### -color\_start()

color모드를 시작한다. 색깔을 넣고자 할때 전에 반드시 써주어야 하는 함수이다.

(has\_colors()의 리턴값이 TRUE 일때만 사용가능하다.)

# (참고)NCURSES 라이브러리 함수

## 출력 관련 함수

### ◆refresh()

화면에 찍은 내용을 갱신한다. curses 모드에서는 출력함수를 이용해 출력해도

refresh()를 쓰기전까진 화면에 나타나지 않는다.

### ◆clear()

화면을 깨끗이 지운다.

### ◆printw

printf와 쓰임새와 사용방법이 같다.(그러나 curses 모드가 시작된 이후 종료되기 전까지는 printf와 같은 일반 출력함수로는 화면에 출력할 수 없다.)

ex)

```
printw("stringWn");
printw("%sWn", "string");
....
```

### ◆addch

putchar와 같이 char의 글자 하나를 출력할 때 사용한다. putchar와 다른 점이라면 curses 모드에서 사용할 수 있는 옵션 줄 수 있다.

ex)

```
#include<ncurses.h>
```

```
int main(void){
    int cnt;
    char s[] = "SPLUG";
    initscr();
```

```
addch(s[0]|A_BOLD);
for (cnt = 1; cnt < 5; cnt++)
    addch(s[cnt]);
refresh();
endwin();
}
```

### ◆addstr()

addstr(string) 해당 문자열을 출력한다.

### ◆move(y,x)

y줄 x번째로 커서를 이동시킨다. x, y의 값은 0~화면의 최대크기의 int 값을

갖는다. 화면의 범위를 벗어날 경우 세그멘테이션 오류가 날 수 있다.

### ◆mvprintw(), mvaddch()

move한 뒤 printw, addch한것과 같은 결과를 출력한다.

예를 들면

```
move(10, 10);
printw("This is 10, 10");
이것은
mvprintw(10, 10, "This is 10, 10");
과 같다.
```

# (참고)NCURSES 라이브러리 함수

## 입력 관련 함수

### ◆scanw() and mvscanw()

scanw는 scanf와 사용방법과 쓰임이 같다. 해당 문자열이나 char int등을 입력 받는다.

mvscanw는 mvprintw와 비슷하게 해당 좌표로 이동후 입력을 받는다.

mvscanw(row, col, "%d", int \*);

### ◆getstr()

string입력을 받는다.

ex)

```
char str[10];
```

```
getstr(str);
```

이렇게 하면 'Wn'를 입력박기전까지의 문자열을 str에 저장한다.

### ◆getch()

한 문자를 사용자로부터 입력을 받는다. getchar과는 달리 'Wn'문자를 입력 받을 때까지

기다리지 않고 바로 입력받은 문자 하나를 리턴해주고 종료한다. 입력받은 문자의 표기 유무는 echo() : 유 와 noecho() : 무로 선택한다.

## Attributes

글자에 특수한 효과를 주기 위해 사용된다.

### attron() and attrset()

글자에 특수한 효과를 준다. attron() 중첩사용하면 중첩효과를 attrset()은 그 밑으로 모두 설정을 attrset()로 해준다.

#### ◆사용 방법

attron(인자);

한꺼번에 여러효과를 주고자할 때 |(OR)를 이용하면 된다.

attron(인자1| 인자2);

이렇게 하면 인자1과 인자2의 효과가 중첩되어 나타난다. 중첩갯수에 제한 없음.

### ◆attroff()

attron()이나 attrset()으로 준 효과를 끈다. 인자는 attron(), attrset()과 동일하다.

ex)

```
attron(A_BLINK);
```

```
printw("HI!!Wn");
```

```
attroff(A_BLINK);
```

### ◆init\_pair()

컬러를 설정할 때 쓰인다.

```
init_pair(int index, int font_color, int blink_color);
```

여기서 index는 설정한 color쌍을 불러 낼때 쓰이게 되는번호이다. 1보다 크거나 같은수 지정 가능

두 번째인자는 글씨색을 세 번째 색은 블록의 색깔을 의미한다.

여기서 색은 ncurses라이브러리에 미리 정의 되어있다.

### ◆init\_color()

미리 정의되어 있는 색깔을 바꿀수 있다.

예를 들어 COLOR\_RED의 색깔을 재 정의 하고자 할 경우

```
init_color(COLOR_RED, r, g, b);
```

여기서 r, g, b는 각각 red, green, blue의 삼원색으로 0~ 1000사이의 값을 가진다.

### ◆COLOR\_PAIR(n)

attron(), attset(), attroff()의 인자로 쓰인다.

여기서 n은 앞에서 init\_pair()에서 첫 번째 인자로 숫자로 들어간 수와 일치 한다.

0은 보통 모양을 의미 한다.

### ◆mvchgat()

attron(COLOR\_PAIR(n))과 유사한 효과를 내는 데 쓰인다.

```
mvchgat(int start_Y, int start_X, int char_num, Attribute, int index, NULL)
```

첫 번째와 두 번째는 효과를 주려고하는 곳의 시작 부분을 세 번째 인자는 효과를 주고자

하는 문자의 숫자를 의미한다. 여기서 -1은 라인의 끝까지를 의미한다. Attribute는 A\_BOLOD나 A\_BLINK와 같은 효과를 의미한다. 네 번째 인자는 init\_pair()에서 설정해준 color 쌍의 인덱스 번호와 일치한다. 역시 0은 기본 모양. 마지막인자는 항상 NULL이다.

# (참고)NCURSES 라이브러리 함수

## 현재 정보를 얻는 함수들

### ◆getmaxyx()

현재 화면의 가로 세로의 크기를 구한다.

```
getmaxyx( win, max_y, max_x);
```

win의 y값의 크기와 x값의 크기를 구해 각각 max\_y, max\_x에 넣어 준다.

여기서 win은 window포인터를 의미하며 표준화면은 stdscr을 넣어 주면 된다.

### ◆getyx()

현재 커서의 위치를 구한다.

```
getyx(win, y, x);
```

win(window pointer)의 커서의 위치를 찾아 y, x에 넣어준다.

## ACS 문자들(특수문자)

ncurses에서는 여러 가지 특수한 문자들을 ACS\_xxx 이런식으로 미리 정의해 놓았다. 여기서 원하는 문자를 찍고자할 때 미리 정의된 것을 이용하면 손쉽게 찍을 수 있다.

## 커서 설정

curs\_set() 이용하여 설정하며 0~2사이의 인자값을 가진다. 0에 가까울수록 커서가 안보인다.

```
ex ) curs_set(0);
```

## 마우스 이벤트

간단한 마우스 이벤트도 ncurses를 통해서 제어 할 수 있다.

## ◆이벤트 얻기(getmouse())

```
MEVENT event;
```

```
ch = getch();
```

```
if(ch == KEY_MOUSE)
```

```
if(getmouse(&event) == OK)
```

```
{
```

```
    /*Do some thing with the event*/
```

```
}
```

ncurses에서는 MEVENT 라는 구조체가 미리 정의 되어있다.

구조체를 살펴보면

```
typedef struct {
```

```
    short id;    //device id
```

```
    int x, y, z;  //마우스의 커서 위치
```

```
    mmask_t bstat; //버튼의 상태 비트를 나타낸다.
```

```
}
```

사용방법은 event라고 선언된 구조체에 마우스 이벤트를 받았다면 마우스버튼 1을 눌렀는지를 검사하려면

```
if(event.bstat & BUTTON1_PRESSED)
```

```
   printw("Left button pressed"); 이런식으로 한다.
```

참고: <https://wiki.kldp.org/wiki.php/NCURSES-Programming-HOWTO>

# 테트리스 프로젝트 2주차 실습&과제 평가

---

- 선택한 자료구조의 효율성에 대해서 평가한다.
  - (과제결과보고서 작성1)효율성을 평가하고 왜 그렇게 생각하는지 상세하게 기술한다
- 자료구조의 효율성은 테트리스의 랭킹 시스템에서 자료구조에 저장된 랭킹 정보 이용하는 동작의 시간 및 공간 복잡도로 평가된다.
- 1<sup>st</sup> 평가 기준
  - Game over가 되고, 새로운 랭킹 정보(사용자 이름, 점수)가 등록될 때, 시간 및 공간 복잡도
- 2<sup>nd</sup> 평가 기준
  - 원하는 랭킹 범위를 입력 받고, 랭킹을 추출하기 위한 과정에서 자료구조를 탐색 및 랭킹 추출에서의 시간 및 공간 복잡도

# 테트리스 프로젝트 2주차 과제&

- 메뉴 2번의 rank에서 새로운 모드 2개를 추가한다.
- 모드 1: 사용자의 이름을 입력 받아, 입력 받은 사용자 이름에 해당하는 모든 랭킹 정보를 찾고, 해당 사용자의 랭킹 정보를 화면에 출력한다.
  - 구축된 자료구조를 유지 및 확장하거나, 새로운 자료구조를 사용할 수 있다.
  - 일치하는 사용자가 있을 때, 해당 사용자의 모든 랭킹 정보를 기존 랭킹 시스템의 출력 방식과 같이, 사용자 이름과 score 순으로 화면에 출력한다.
  - 일치하는 사용자가 없을 때는 “search failure: no name in the list”를 출력한다.
  - (과제결과보고서 작성2) 사용자의 이름을 입력하고 해당하는 랭킹 정보 출력 화면 첨부
  - (과제결과보고서 작성3) 사용자 이름으로 원하는 사용자의 이름을 검색할 때의, 시간 및 공간 복잡도
- 모드 2: 원하는 랭킹 정보를 삭제한다.
  - 구축된 자료구조나 새로운 자료구조를 사용할 수 있다.
  - 입력은 삭제하길 원하는 랭킹(정수)를 입력 받는다.
  - 일치하는 랭킹이 있을 때는 랭킹 정보를 삭제하고, “result: the rank deleted”를 출력한다.
  - 일치하는 랭킹이 없을 때는 “search failure: the rank not in the list”를 출력한다.
  - (과제결과보고서 작성3) 선택한 자료구조에 맞춰 삭제 알고리즘을 그림으로 표현한다.



## 테트리스 프로젝트 2주차 숙제 - 모드 1(1/3)

## □ 입력 파일

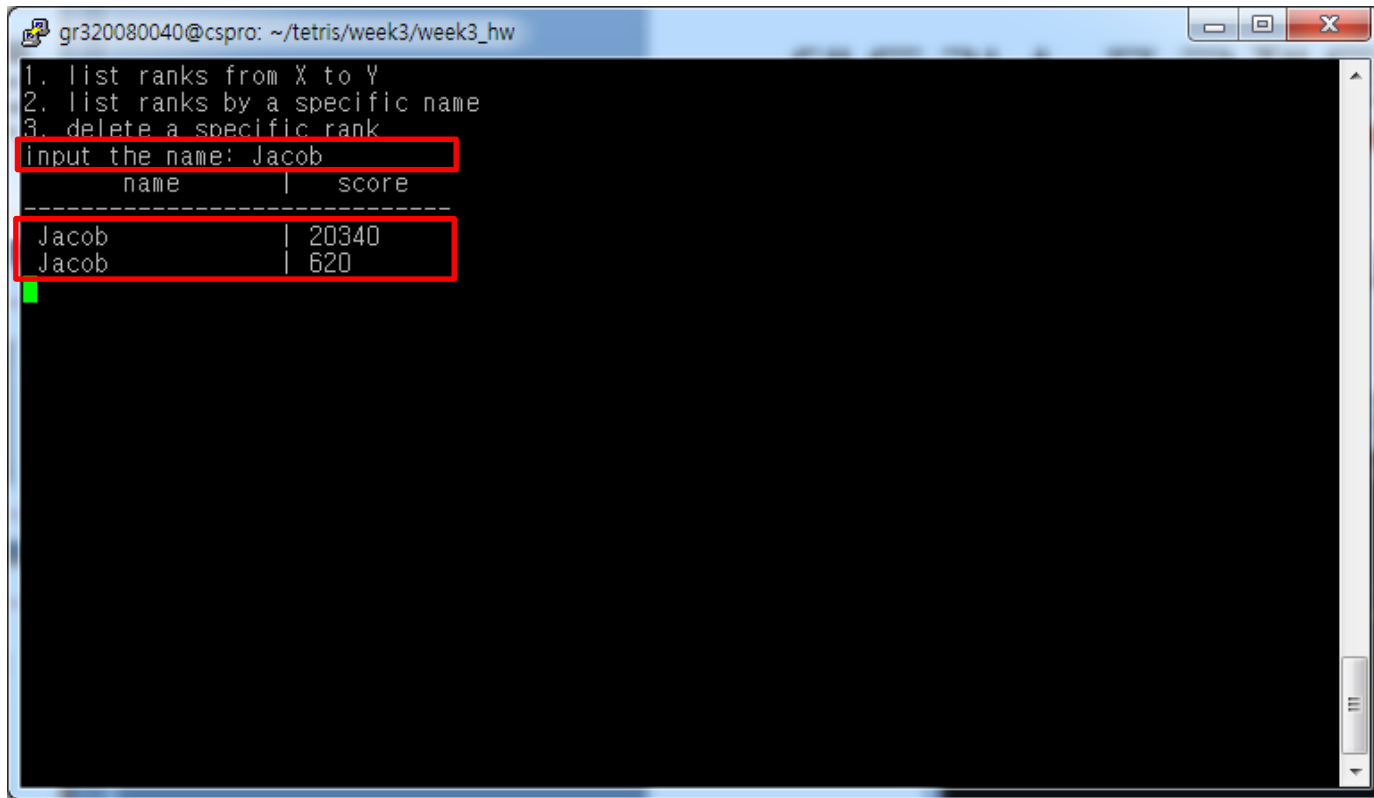
- rank.txt

[illegible]

# 테트리스 프로젝트 2주차 숙제 – 모드 1(2/3)

## □ 출력 예제 1

- Jacob을 입력하여 Jacob의 모든 랭킹 정보들을 찾는 경우



A terminal window titled 'gr320080040@cspro: ~/tetris/week3/week3\_hw' displays the following text:

```
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
input the name: Jacob
```

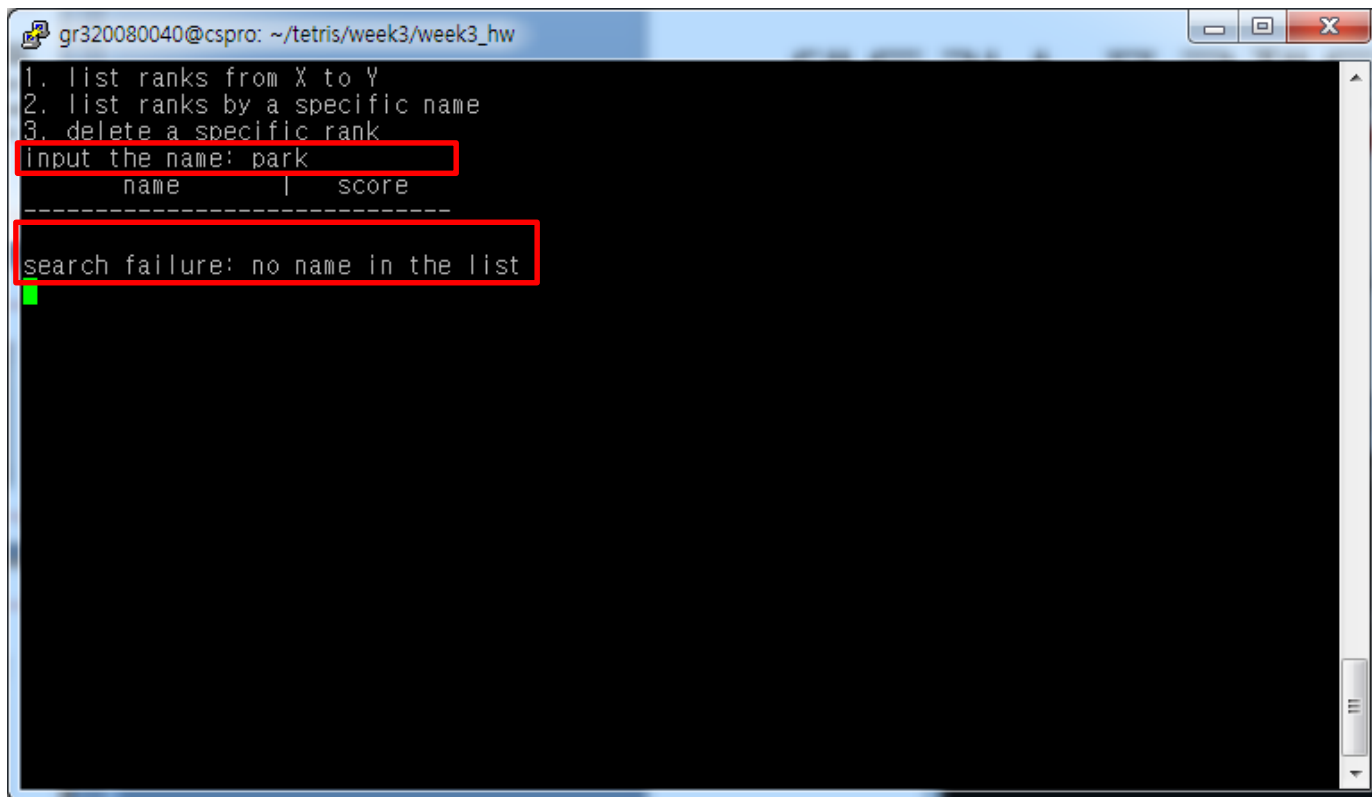
name	score
Jacob	20340
Jacob	620

The input 'Jacob' and the resulting table are highlighted with red boxes in the original image.

# 테트리스 프로젝트 2주차 숙제 – 모드 1(3/3)

## □ 출력 예제 2

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



```
gr320080040@cspro: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
input the name: park
  name      |  score
-----
search failure: no name in the list
```

The image shows a terminal window with a blue title bar. The window title is "gr320080040@cspro: ~/tetris/week3/week3\_hw". The terminal content displays a menu with three options: "1. list ranks from X to Y", "2. list ranks by a specific name", and "3. delete a specific rank". Below the menu, the prompt "input the name:" is followed by the user input "park". A table header is shown with "name" and "score" columns, separated by a vertical line and underlined. Below the header, a dashed line indicates the start of the data list. The message "search failure: no name in the list" is displayed, indicating that the name "park" was not found in the list. A green cursor is visible at the end of the message.

## 테트리스 프로젝트 2주차 숙제 - 모드 2(1/5)

## □ 입력 파일

- rank.txt
- Kim의 정보를 삭제할 예정.

A screenshot of a Windows command prompt window. The title bar shows the user 'gr320080040@cspro' and the current directory '~/tetris/week3/week3\_hw'. The terminal output displays the contents of a file named 'rank.txt':  

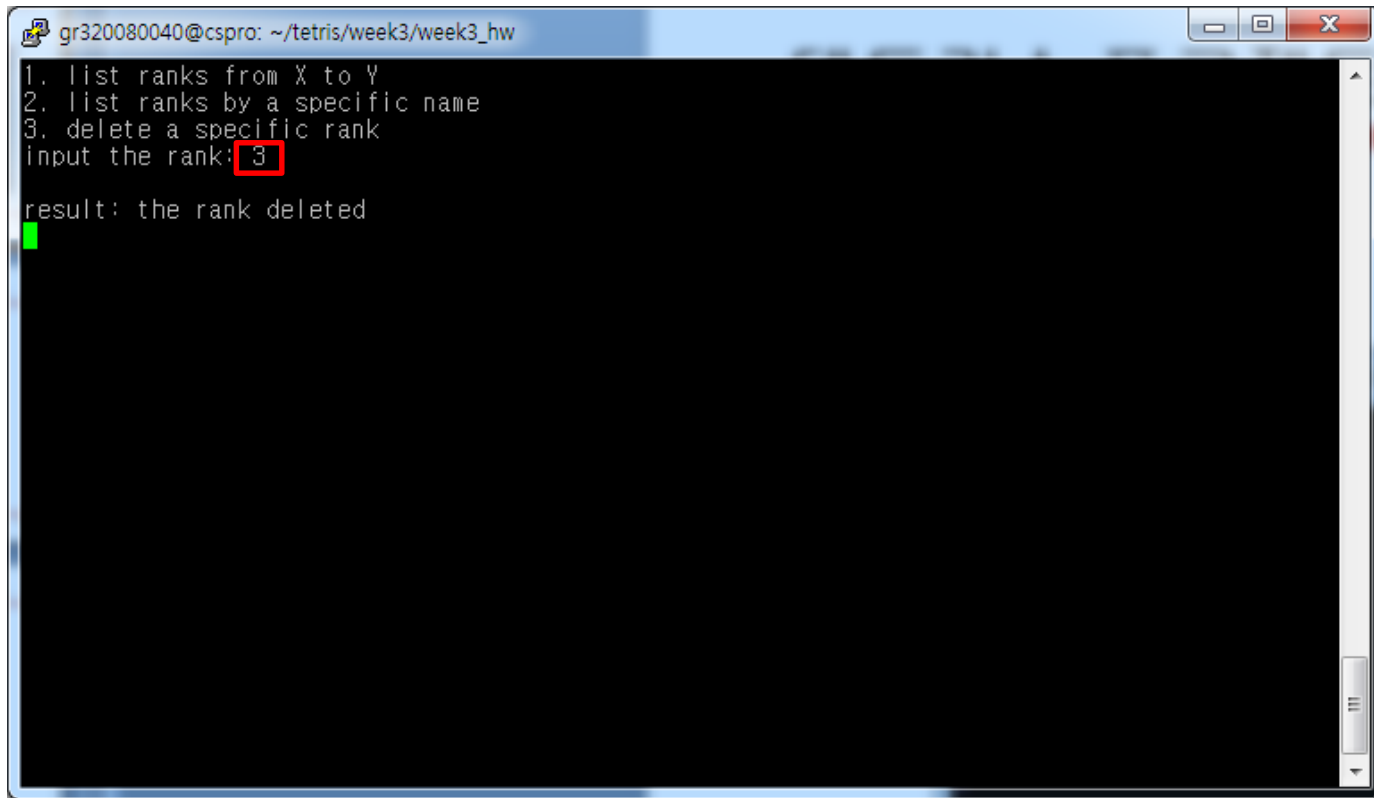
```
Jacob 20340  
Park 17990  
Kim 7430  
Jacob 620  
Chris 450
```

  
The line 'Kim 7430' is highlighted with a red rectangular box. Below the names and scores, there are several tilde (~) characters representing empty lines in the file. At the bottom of the window, a status bar indicates '"rank.txt" 6L, 54C' (6 lines, 54 columns), the cursor position '1,1', and the encoding '모두' (All).

# 테트리스 프로젝트 2주차 숙제 – 모드 2(2/5)

## □ 출력 예제 1

- Kim의 랭킹 정보를 삭제하기 위해, 3을 입력

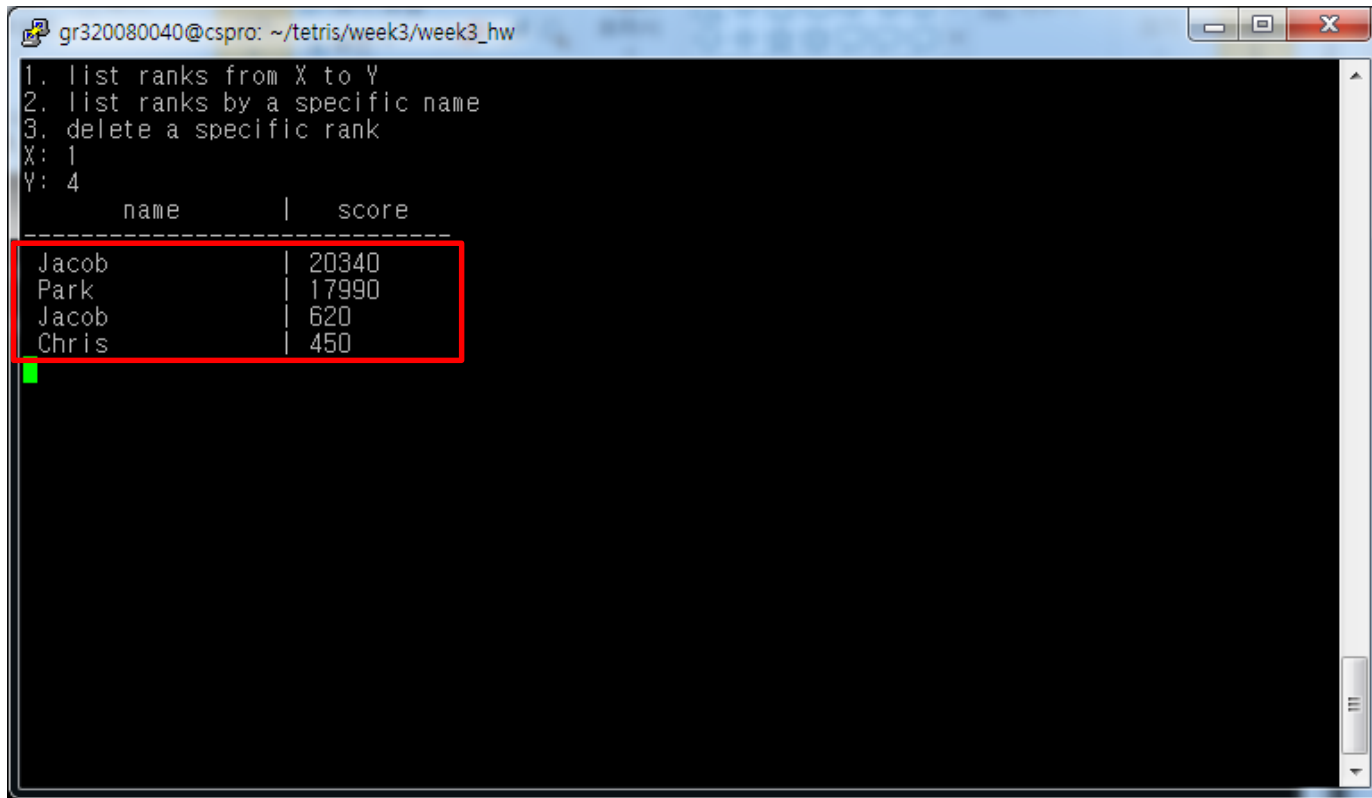


```
gr320080040@cspro: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
input the rank: 3
result: the rank deleted
```

# 테트리스 프로젝트 2주차 숙제 – 모드 2(3/5)

## □ 출력 예제 2

- Kim의 랭킹 정보가 삭제된 것을 확인.



```
gr320080040@csp: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
X: 1
Y: 4
name | score
-----
Jacob | 20340
Park  | 17990
Jacob | 620
Chris | 450
```

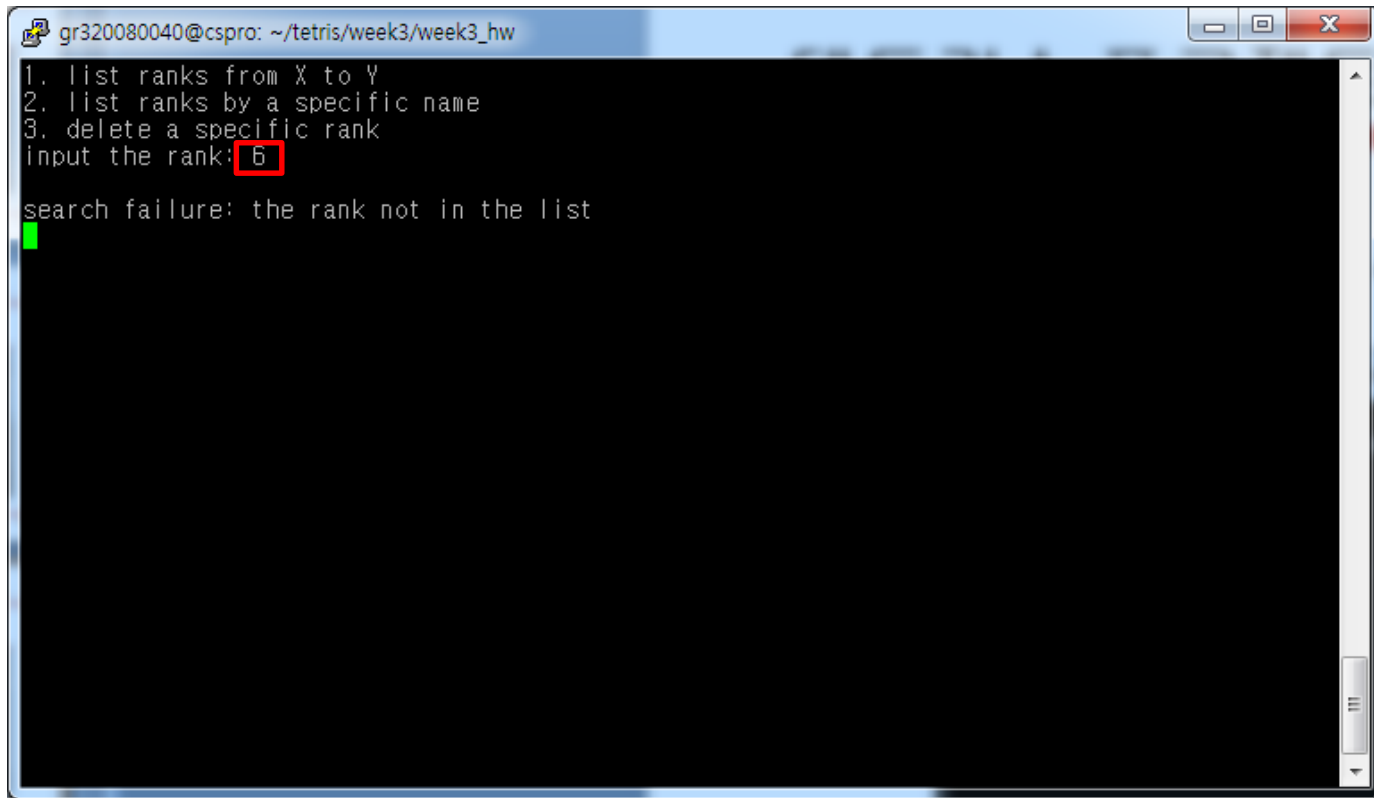
## 테트리스 프로젝트 2주차 숙제 - 모드 2(4/5)

- rank.txt에서 Kim의 랭킹 정보가 삭제되었는지 확인.

# 테트리스 프로젝트 2주차 숙제 – 모드 2(5/5)

## □ 출력 예제 4

- 존재하지 않는 랭킹을 삭제하려고 할 때는 다음 메시지를 출력한다.



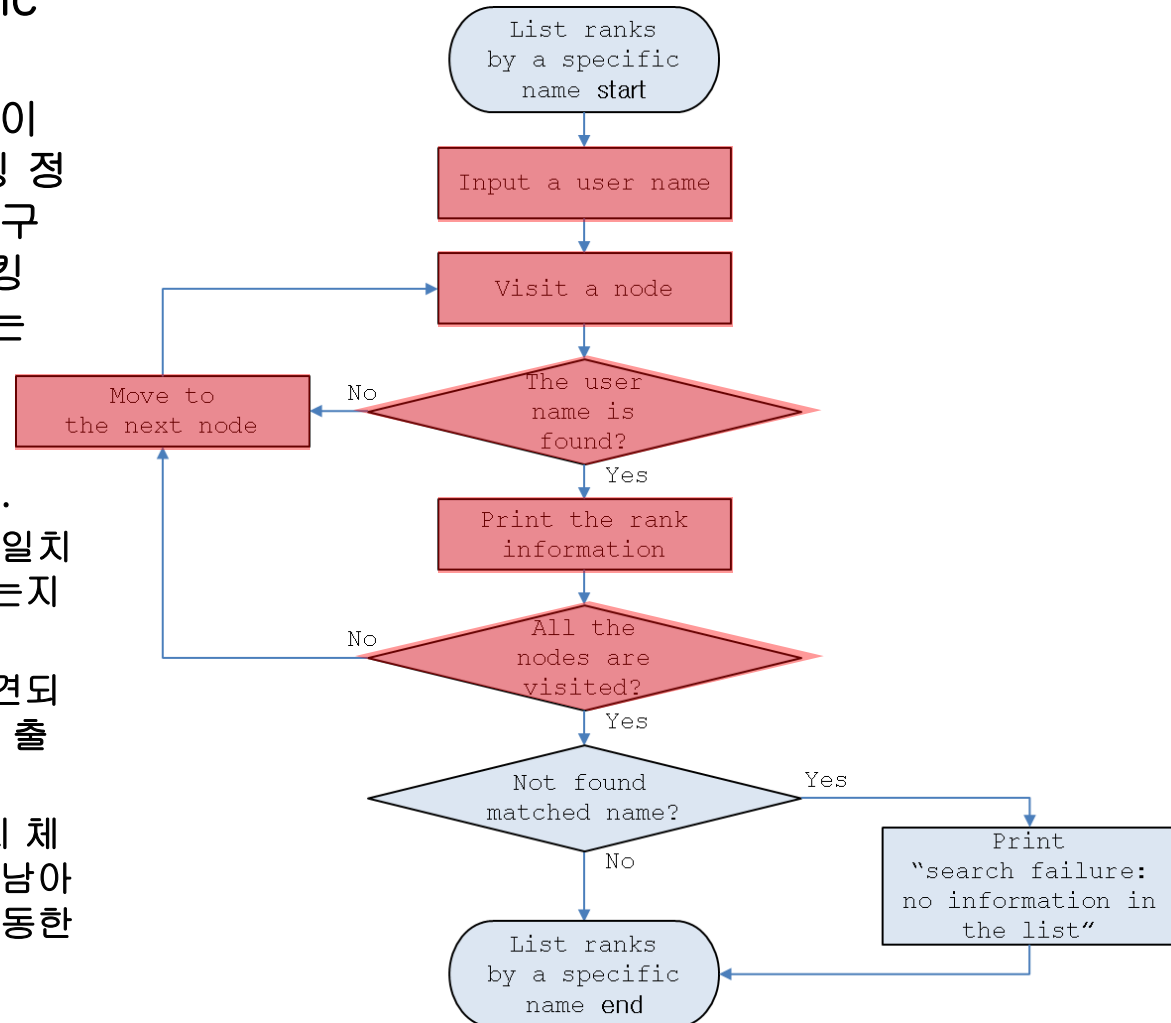
```
gr320080040@cspro: ~/tetris/week3/week3_hw
1. list ranks from X to Y
2. list ranks by a specific name
3. delete a specific rank
input the rank: 6
search failure: the rank not in the list
```



## 2주차 숙제 구현 함수 - rank () (1/2)

### □ List ranks by a specific name

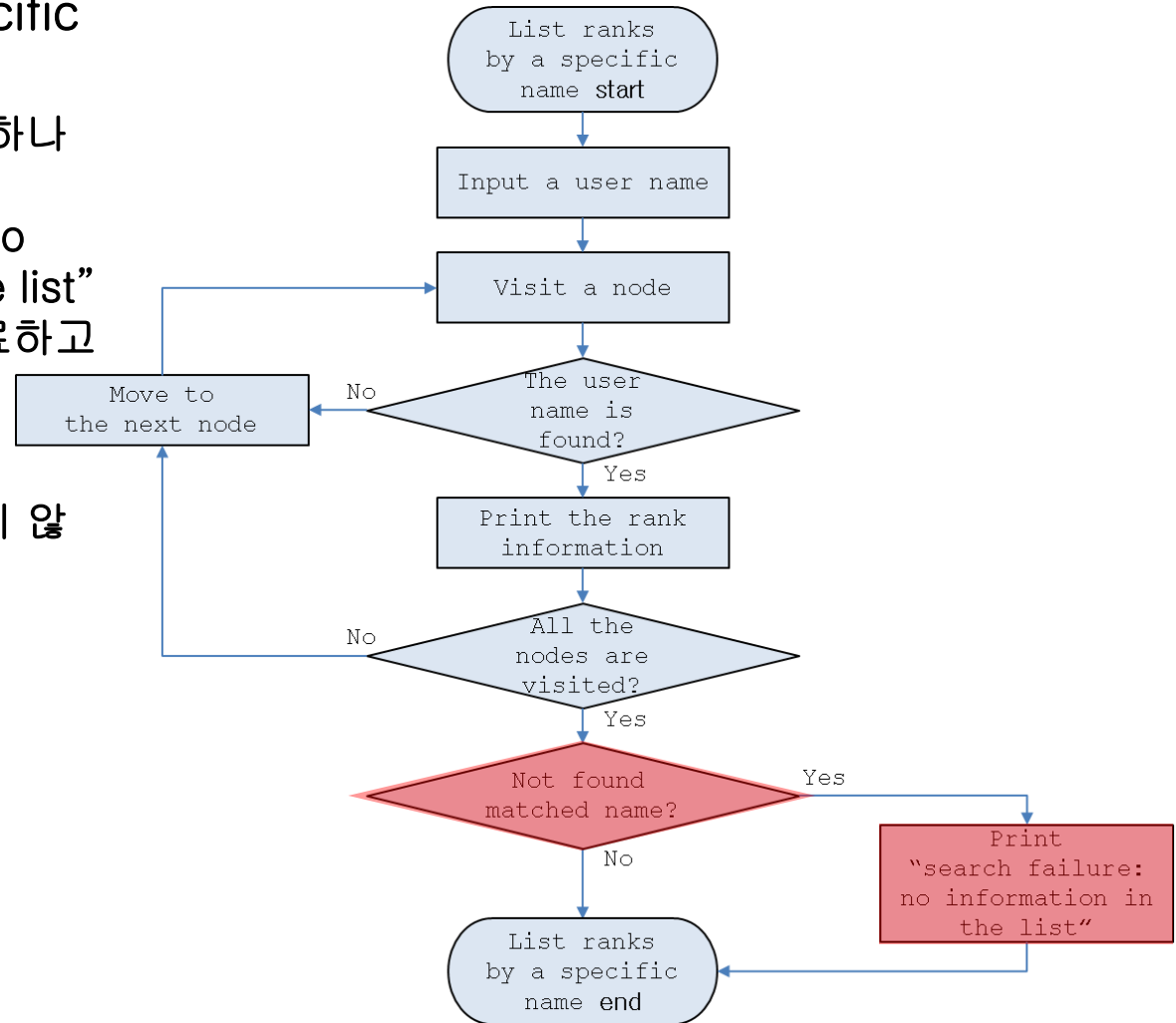
- 찾고자 하는 사용자 이름을 입력 받고, 랭킹 정보를 갖고 있는 자료구조에서 일치하는 랭킹 정보를 찾고 출력하는 함수.
- 사용자이름을 입력한다.
- 각 노드를 방문하면서, 일치하는 사용자 이름이 있는지 확인한다.
- 찾는 사용자 이름이 발견되면, 랭킹 정보를 화면에 출력한다.
- 모든 노드를 방문했는지 체크해서, 방문할 노드가 남아있으면, 다음 노드로 이동한다.



## 2주차 숙제 구현 함수 - rank () (2/2)

### □ List ranks by a specific name

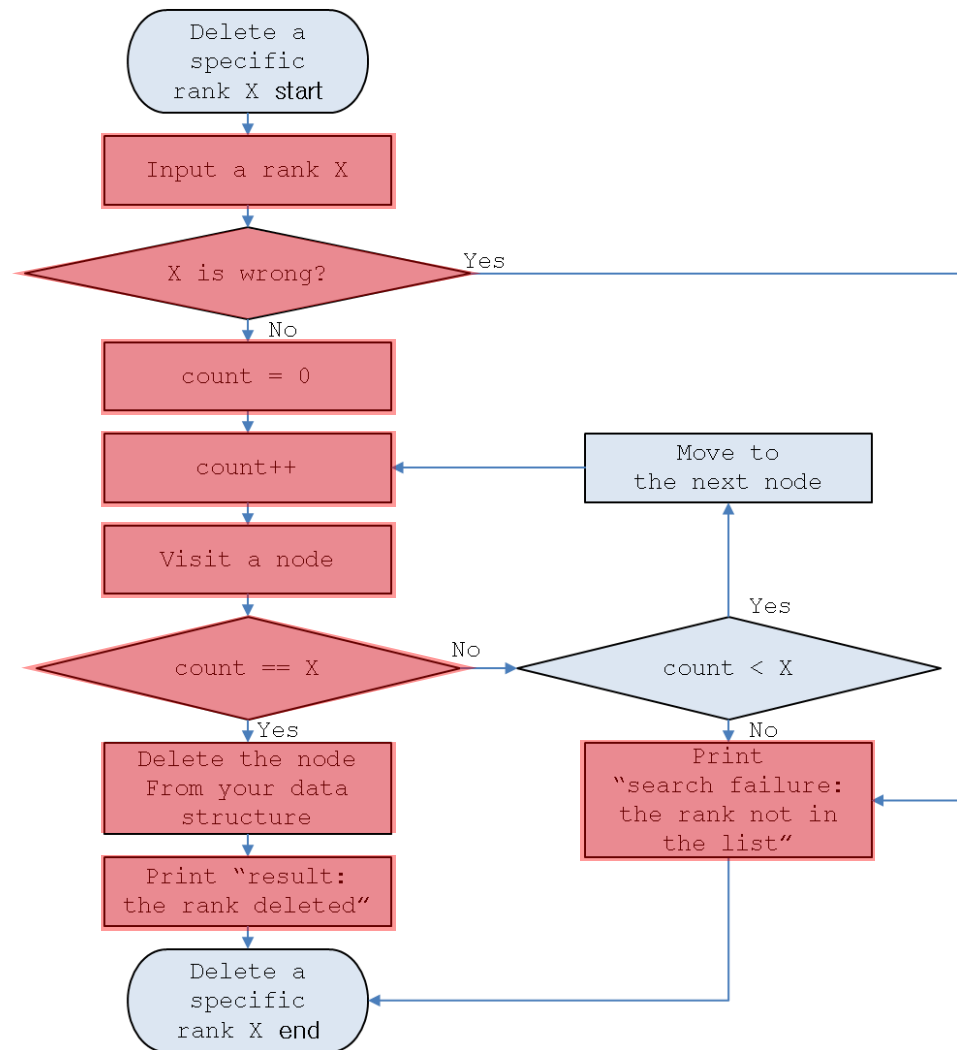
- 일치하는 이름을 하나도 찾지 못했다면, “search failure: no information in the list”를 출력한 후, 종료하고
- 일치하는 이름을 찾았다면, 아무런 메시지를 출력하지 않고 종료한다.



## 2주차 숙제 구현 함수 - rank () (1/2)

### □ Delete a specific rank X flow chart

- 삭제하길 원하는 랭킹(정수)을 입력 받고, 랭킹 정보를 저장하고 있는 자료 구조에서 해당 랭킹에 해당하는 랭킹 정보(노드)를 삭제하는 함수.
- 삭제하길 원하는 랭킹(정수)를 입력한다.
- 입력받은 랭킹이 잘못되었는지 체크해서 잘못되었다면, 메시지를 출력하고 종료한다.
- 변수 count를 0으로 초기화한다.
- 변수 count를 1증가한다.
- 노드를 방문해서 X번째인지 확인해서, 그렇다면 노드를 삭제한다.
- 화면에 “result: the rank deleted”를 출력한다.



## 2주차 숙제 구현 함수 - rank () (2/2)

### □ Delete a specific rank X flow chart

- 점수에 대해서 내림차순으로 정렬된 리스트 상에서 현재 방문한 노드가 X번째 노드인지 체크해서, X번째 노드가 아니라면, **count** 값이 X보다 작은지 확인한다.
- **count** 값이 X보다 작다면, 다음 노드로 이동한다.
- **count** 값이 X보다 크다면, “search failure: the rank not in the list”를 출력하고 종료한다(앞에서 X가 잘못된 입력인지 체크하므로 이 경우는 발생하지 않으나, 조건문이 모든 경우 고려한다는 것을 보여주기 위해 플로우 차트 상에서만 명시).

