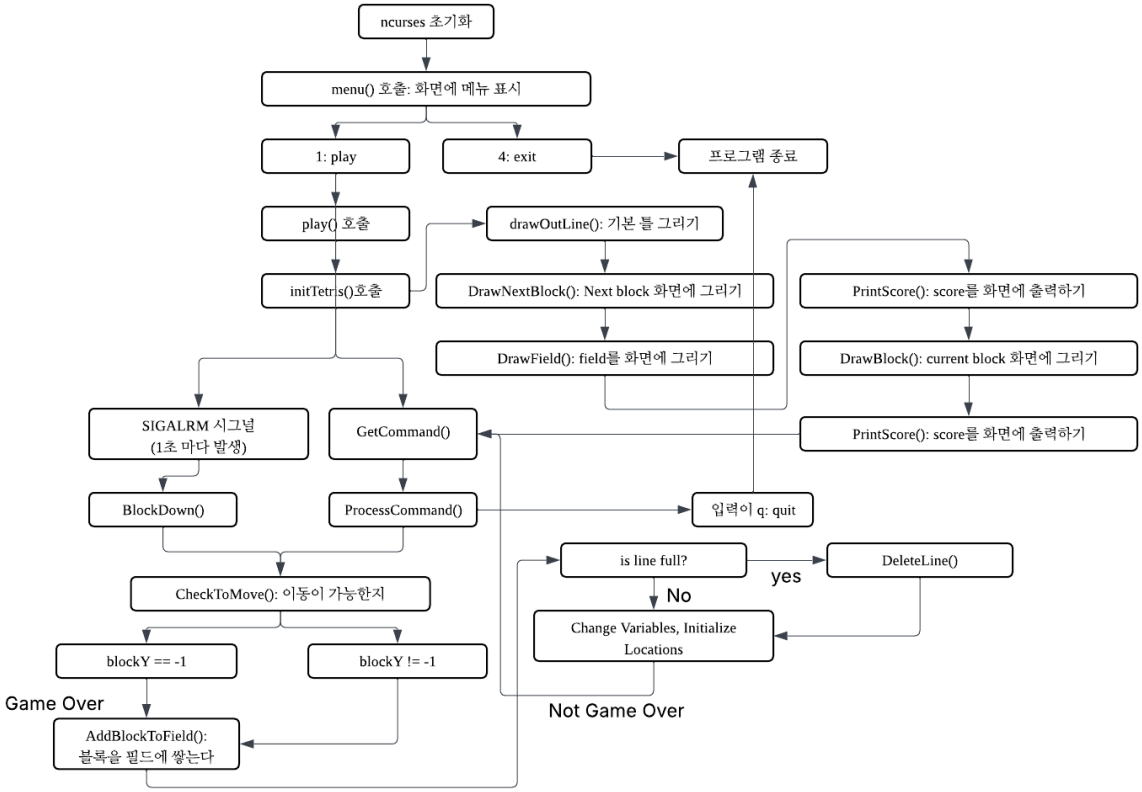
8주차 예비보고서

전공 : 경영학과 학년 : 4학년 학번 : 20190808 이름 : 방지혁

1. 테트리스 frame 프로그램 파일을 미리 읽어보고(부록1 ncurses 라이브러리 포함), 테트리스 게임의 flow chart를 자세히 작성하시오. 그리고 테트리스 게임을 구성하는 각 함수의 기능에 대해서 설명하시오. 1번 문항은 테트리스_1주차_강의슬라이드.pdf 45pg의 모든 함수에 대한 설명이 있어야합니다.



void initTetris(); 테트리스 프로그램의 모든 전역 변수들을 초기화하고 그림 그리는 함수들을 호출합니다.

void DrawOutline(); 블록이 떨어지는 공간, nextblock 공간, score 공간의 테두리를 모두 그려줍니다.

void DrawField(); 테트리스 field를 그립니다

void PrintScore(int score); 점수를 표시합니다.

void DrawNextBlock(int* nextBlock); 다음에 나올 block을 표시합니다.

void DrawBox(int y, int x, int height, int width); 인자로 들어간 좌표에 선으로 된 직사각형을 그립니다.

void DrawBlock(int y, int x, int blockID, int blockRotate, char tile); 해당 좌표에 블럭을 그립니다.

int GetCommand(); 키보드 입력을 받아서 명령으로 바꿉니다.

int ProcessCommand(int command); 명령을 받아 이를 수행합니다.

void BlockDown(int sig); block이 1초마다 내려가도록 호출되는 함수입니다. 더 이상 내릴 수 없을 경우 해당 current block을 필드와 합치고 만약 채워진 line이 있다면 해당 line을 지웁니다. 그리고 다음 block을 사용하도록 합니다.

int CheckToMove(char f[HEIGHT][WIDTH], int currentBlock, int blockRotate, int blockY, int blockX); 현재 블록이 입력된 위치와 회전상태를 고려해서 이동할 수 있는지 확인하고 가능하면 1을 반환 충돌하면 0을 반환합니다.

void DrawChange(char f[HEIGHT][WIDTH], int command, int currentBlock, int blockRotate, int blockY, int blockX); 입력된 명령에 의해 이전 블록을 지우고, 블록이 바뀌면 새로 그립니다.

void AddBlockToField(char f[HEIGHT][WIDTH], int currentBlock, int blockRotate, int blockY, int blockX); 블록이 있는 위치만 필드에 쌓습니다.

int DeleteLine(char f[HEIGHT][WIDTH]); 채워진 가로줄을 삭제하고 점수를 계산하여 반환합니다.

void play(); 게임을 시작합니다.

char menu(); 메뉴를 그립니다.

2. 실습시간에 구현할 5가지 함수들에 대한 간단한 pseudo code를 제시하시오.

int CheckToMove(char field[HEIGHT][WIDTH], int currentBlock, int blockRotate, int blockY, int blockX); 블록이 움직일 수 있는지 확인

```
함수 CheckToMove(...):
        FOR i = 0 to 3:
                FOR j = 0 to 3:
                        IF block[현재블록][블록회전][i][j] == 1:
                                // 블록이 화면 위쪽에 있는 경우 스킵
                                IF i + blockY < 0:
                                        continue
                                // 필드 경계 확인
                                // 아래쪽 경계 넘음
                                IF i + blockY >= HEIGHT:
                                        Return 0
                                // 왼쪽 경계 넘음
                                IF j + blockX < 0:
                                        Return 0
                                // 오른쪽 경계 넘음
                                IF j = block X >= WIDTH:
                                        Return 0
                                // 다른 블록과 겹치는지 확인
                                IF field[i + blockY][j + blockX] == 1
                                        Return 0
        Return 1 // 이동 가능
```

void DrawChange(char field[HEIGHT][WIDTH], int command, int currentBlock, int blockRotate, int blockY, int blockX); 예전 블록을 지우고 다시 그려줌

```
함수 DrawChange(...)
        // 이전 블록 정보 저장 변수
        이전회전 = 블록회전
        이전Y = blockY
        이전X = blockX

        SWITCH command:
                CASE KEY_UP: // 회전
                        이전회전 = (블록회전 + 3) % 4
                        Break
                CASE KEY_DOWN: // 아래로 이동
                        이전Y = blockY – 1
                        Break
                CASE KEY_RIGHT:
                        이전X = block X – 1
                        Break
                CASE KEY_LEFT:
                        이전X = block X + 1
                        Break
        // 이전 블록이 그려진 거 지우기
        FOR i = 0 to 3:
                FOR j = 0 to 3:
                IF block[현재블록][이전회전][i][j] == 1 AND i + 이전Y >= 0:
                        커서 이동
                        "." 출력
        // 새로운 블록 그리기
        DrawBlock(blockY, blockX, 현재블록, 블록회전, ' ')
```

void BlockDown(int sig); 매초마다 블록을 한 칸씩 내림

```
함수 BlockDown(...):
        // 타이머 리셋
        timed_out = 0

        // 우선 한 칸 아래로 이동 가능한지 확인
        IF CheckToMove(필드, 현재블록, 회전상태, blockY+1, blockX):
                blockY++
                DrawChange(field, KEY_DOWN, nextBlock[0], blockRotate, blockY, blockX):
                // 이동할 수 있기에 화면 업데이트
        // 더 이상 못 내려감
        ELSE:
                // 현재 블록을 필드에 추가
```

```
AddBlockToField(필드, 현재블록, 회전상태, blockY, blockX);
// 꽉 찬 줄 있으면 확인하여 지우고 점수 획득
Score += DeleteLine(...)
// 다음 블록
현재블록 = 다음블록
다음블록 = 랜덤블록
// 블록 위치 리셋
blockY = -1 // 화면 밖
blockX = 중앙
blockRotate = 0
IF 새 블록을 초기 위치에 놓을 수 없으면:
        gameOver = 1 // 게임 종료
// 화면 그리기
DrawField()
DrawNextBlock()
PrintScore()
```

void AddBlockToField(char field[HEIGHT][WIDTH], int currentBlcok,int blockRotate, int blockY, int blockX); 블록을 필드에 쌓음

```
함수 AddBlockToField(...):
    // 4 X 4 배열을 순회하면서 실제로 블록이 있으면(값이 1)
    // 필드의 위치에 블록이 있다고 표시해야 함
    FOR i = 0 to 3:
        FOR j = 0 to 3:
            // 블록의 해당 위치에 실제로 블록이 있다면
            IF block[현재블록][블록회전][i][j] == 1:
                // 필드의 해당 위치를 1로 설정
                field[i +. blockY][j + blockX] = 1
```

int DeleteLine(char field[HEIGHT][WIDTH]); 채워진 가로줄을 삭제

```
함수 DeleteLine(...):
    삭제된_줄_개수 = 0
    i = HEIGHT – 1 // 맨 아랫줄에서 시작해야해서
    WHILE i >= 0:
        // 현재 줄이 꽉 찼는지 확인
        j = 0
        FOR j = 0 to WIDTH – 1:
            If field[i][j]가 비어있으면:
                Break // 빈 칸 발견, 이 줄은 꽉 안 참
        // 어? 현재 줄이 꽉 차 있네
```

```
            IF j == WIDTH:
                        삭제된_줄_개수++

                        FOR k = i to 0:
                                    IF k == 0: // 맨 윗 줄인 경우
                                                k번째 줄을 0으로 초기화
                                    ELSE:
                                                k번째 줄 = k-1번째 줄
                        i는 그대로 유지
            ELSE:
                        // 꽉 안 찼음
                        i—

      반환 점수 = 삭제된_줄_개수 * 삭제된_줄_개수 * 100
```