8주차 예비보고서

전공: 컴퓨터공학과 학년: 4학년 학번: 20192135 이름: 윤영인

**1.**

**1.1**

1) CheckToMove

블록을 움직일 수 있는지 확인하는 함수로, 움직일 수 없으면 0, 있으면 1을 반환한다. 블록을 놓으려고 하는 필드에 이미 블록이 쌓여있거나, 좌표가 출력 영역을 벗어날 경우 움직일 수 없다.

2) DrawChange

이전 블록을 삭제하고 새로운 블록을 그리는 함수로, command를 입력 받아 블록의 회전수, 좌표를 변경하여 블록의 모양을 바꿔 다시 그린다. 블록을 그릴 때는 DrawBlock 함수를 이용하여 새 블록을 그리기 전에는 이전 블록을 지워야 한다.

3) BlockDown

현재 블록을 한 칸씩 아래로 떨어뜨리는 함수로, CheckToMove 함수를 이용하여 블록을 한 칸 내릴 수 있는지 확인한다. 만약 누적된 블록이 영역에 모두 차서 블록을 내릴 수 없다면 game over 변수를 1로 갱신하고 갱신된 정보를 다시 그린다.

4) AddBlockToField

주어진 필드 상의 위치에 현재 블록을 쌓는 함수로, field 배열을 1로 설정하여 블록이 쌓여있음을 표시한다.

5) DeleteLine

완전히 채워진 한 줄을 지우고, 지운 줄에 대한 점수를 계산하는 함수이다. 현재 블록이 쌓인 영역을 확인하여 한 줄이 가득 채워진 줄이 있으면 해당 줄을 삭제하고, 줄 위의 필드 정보를 한 줄씩 내려 갱신한다. 지워진 줄에 대한 점수를 계산하여 반환한다.

6) Menu

게임을 처음 실행할 때 실행되는 함수로, 1은 게임 시작, 4는 게임 종료를 의미한다.

7) Play

게임을 실행하는 함수로, 사용자에게 command를 입력 받아 프로세스를 수행하도록ProcessCommand를 수행하고 매초마다 블록을 떨어뜨리도록 BlockDown 함수를 실행한다.

8) DrawField

테트리스의 필드를 그리는 함수이다.

9) DrawNextBlock

다음 블록의 정보를 바탕으로 다음 블록을 확인하는 상자에 다음 블록을 그리는 함수이다.

10) PrintScore

게임 진행 시 완전히 채워진 라인을 삭제하여 반환된 점수를 계산하여 화면에 출력하는 함수이다.

11) DrawOutLine

테트리스 필드의 테두리를 그리는 함수이다.

12) GetCommand

사용자가 입력한 command를 입력받아 처리하는 함수로, command에는 KEY\_RIGHT, KEY\_LEFT, KEY\_DOWN, QUIT이 있으며 이 외의 것들은 명령으로 취급하지 않는다.

13) ProcessCommand

입력받은 명령의 적절한 동작을 취하는 함수로, KEY\_RIGHT과 KEY\_LEFT는 각각 오른쪽, 왼쪽으로 이동하며 KEY\_DOWN은 아래로 내려가도록 한다.

14) DrawBox

인자로 받은 가로와 세로 크기의 사각형을 그리는 함수이다.

15) DrawBlock

화면에 테트리스 블록을 그리는 함수로, 입력 받은 (x,y) 좌표에 tile 형식으로 블록을 그린다.

**1.2 플로우 차트**

텍스트, 도표, 기술 도면, 평행이(가) 표시된 사진

자동 생성된 설명

menu 함수를 실행하여 사용자가 1을 입력하면 게임을 시작하고, 4를 입력하면 종료한다. 1을 입력하여 게임을 시작하면 InitTetris 함수로 테트리스 필드를 초기화하고, GetCommand() 함수로 사용자에게 command를 입력받는다. 입력받은 command에 대한 동작을 ProcessCommand로 처리하는데 Quit이 아니면 블록을 그릴 수 있는지 확인하여 테트리스 블록을 그린다. 만약 command가 QUIT 일 경우 게임을 종료한다. 필드가 다 채워져서 Game Over이 되면 게임 수행을 종료한다.

**2.**

1) CheckToMove

for (i= 0 ~ 4) {

for (j = 0 ~ 4) {

if 이미 블록이 쌓여있으면

return 0

if 블록의 좌표가 필드를 벗어났으면

return 1

}

}

return 1

2) DrawChange

Switch(command) {

KEY\_UP: 이전 방향으로 돌림 (0 → 3 → 2 → 1 →0 순)

KEY\_LEFT: 이전 블록의 x좌표 = 현재 x좌표 + 1

KEY\_RIGHT: 이전 블록의 x좌표 = 현재 x좌표 – 1

KEY\_DOWN: 이전 블록의 y좌표 = 현재 y좌표 – 1

}

for (i = 0 ~ 4) {

for (j = 0 ~ 4) {

if 이전 좌표의 위치에 블록이 그려져 있으면

then 좌표로 이동하여 ‘.’ 으로 변경

}

}

DrawBlock으로 블록을 그림

커서를 밖으로 이동

3) BlockDown

if CheckToMove == 1

then {

현재 블록의 Y좌표를 1증가

DrawChange

}

else

then {

if 현재 블록의 Y좌표 == -1

then {

gameOver = 1

}

else

then {

AddBlockToField

DeleteLine 함수를 호출하여 score 갱신

현재 블록에 다음 블록을 넣음

다음 블록을 rand()로 초기화

현재 블록의 좌표 및 회전수를 초기화

DrawNextBlock을 호출하여 다음 블록을 그림

score 출력

현재 블록과 필드를 그림 (DrawBlock, DrawField)

}

}

time\_out = 0 // 매 초마다 한 칸 아래로 떨어지도록 하기 위해 설정

4) AddBlockToField

for (i = 0 ~ 4) {

for (j = 0 ~ 4) {

if 이미 블록이 쌓여있으면

then {

field[현재 블록의 Y 좌표 + i][현재 블록의 X좌표 + 1] = 1

}

}

5) DeleteLine

for (i= 0 ~ HEIGHT) {

lineflag = 1

for (j = 0 ~ WIDTH) {

if field[i][j] == 0

then {

lineflag = 0

반복문 종료

}

}

if lineflag == 1

then

for (m = 0 ~ i) {

for (n = 0 ~ WIDTH) {

field[m][n] = field[m – 1][n] // 삭제되는 라인의 윗 블록들을 한 줄 아래로 내림

}

}

}

line\_count 1 증가

return line^2 \* 100

}