8주차 결과보고서

전공: 기계공학과 학년: 3학년 학번: 20191820 이름: 김형준

**1.**

1) CheckToMove()

예비보고서의 pseudo code를 그대로 c언어의 문법에 맞게 구현했기 때문에 달라진 점은 따로 없다.

테트리스 블록의 크기는 4(BLOCK\_HEIGHT)\*4(BLOCK\_WIDTH)로 항상 일정하므로 이중 for문으로 반복되나, 시간복잡도는 input의 크기와 상관없이 항상 일정하다.

따라서 이 함수의 시간복잡도는 O(1)이다.

함수 내부에서 f[HEIGHT][WIDTH]를 사용하나, C언어는 배열을 함수에 전달할 때 배열 전체가 아닌 배열의 주소만을 전달하므로 공간복잡도는 O(1)이다.

2) DrawChange()

다른 함수와 마찬가지로 예비보고서의 pseudo code를 그대로 c언어의 문법에 맞게 구현했기 때문에 달라진 점은 따로 없다.

CheckToMove함수와 마찬가지로, 테트리스 블록의 크기가 상수이므로 이중 for문으로 반복 되고 있지만, 시간복잡도는 O(1)이다.

단순히 배열 f의 주소를 함수에 전달받을 뿐이므로 이 함수의 공간복잡도는 O(1)이다.

3) AddBlockToField()

다른 함수와 마찬가지로 예비보고서의 pseudo code를 그대로 c언어의 문법에 맞게 구현했기 때문에 달라진 점은 따로 없다.

테트리스 블록의 크기가 상수이므로 이중 for문으로 반복되고 있지만, 시간복잡도는 O(1)이다. 함수 내부에서 f[HEIGHT][WIDTH]를 사용하나, 배열 f 전체가 아닌 주소를 함수에 전달받으므로 이 함수의 공간복잡도는 O(1)이다.

4) DeleteLine()

다른 함수와 마찬가지로 예비보고서의 pseudo code를 그대로 c언어의 문법에 맞게 구현했기 때문에 달라진 점은 따로 없다.

모든 열마다 한 줄이 꽉 차있는지 확인하고, 꽉 찬 줄이 있다면 그 줄보다 위에 있는 모든 줄을 한 칸씩 내린다. 그러나 테트리스 게임에서 한 번에 4줄까지만 꽉 채울 수 있으므로

(I자 블록이 4줄이 세로로 한 칸씩 빈 자리에 세로로 떨어지는 경우)

이 함수의 시간복잡도는 O(HEIGHT\*WIDTH)이다. (지운 줄 위의 모든 줄을 한 줄씩 내리는데 걸리는 시간복잡도는 O(HEIGHT\*WIDTH)지만, 줄이 지워지는 총 횟수는 O(1)이므로 총 시간복잡도는 O(HEIGHT\*WIDTH)가 된다.)

함수 내부에서 f[HEIGHT][WIDTH]를 사용하나, 배열 f 전체가 아닌 주소를 함수에 전달받으므로 이 함수의 공간복잡도는 O(1)이다.

5) BlockDown()

다른 함수와 마찬가지로 예비보고서의 pseudo code를 그대로 c언어의 문법에 맞게 구현해서 크게 달라진 점은 없으나, 함수 호출 후 커서의 위치를 필드 바깥으로 옮기기 위해

DrawField() 호출 후 move(HEIGHT,WIDTH+10);를 추가하였다.

함수 내부에서 CheckToMove(), DrawChange(), AddBlockToField(), DeleteLine(),

DrawField(), PrintScore(), DrawNextBlock()을 호출하므로 이 함수의 시간복잡도는 해당 함수들의 시간복잡도 중 가장 큰 값인 O(HEIGHT\*WIDTH)를 가지게 된다. (BlockDown()의 코드 중 함수 호출 부분을 제외한 부분에서 O(HEIGHT\*WIDTH)를 넘어가는 코드는 존재하지 않음)

이 함수의 공간복잡도는 O(1)이다.

**2.**

1) 그림자 기능

DrawBlock(y, x, blockID, blockRotate, tile)

for i = 0 to 3

for j = 0 to 3

if (block[blockID][blockRotate][i][j]==1 && i+y>=0)

move(i+y+1,j+x+1)

draw tile

move(HEIGHT,WIDTH+10)

: 테트리스 필드의 (y,x)에 tile문자로 블록을 그린다. 시간복잡도 : O(1), 공간복잡도 : O(1)

DrawShadow(y, x, blockID, blockRotate)

Sy=y, Sx=x

for i = y to HEIGHT

if (CheckToMove(field,blockID,blockRotate,i,x))

Sy = i

else

break

DrawBlock(Sy, Sx, blockID,blockRotate,'/')

: (y,x)에 있는 블록의 그림자를 그린다. 시간복잡도 : O(HEIGHT), 공간복잡도 : O(1)

DrawBlockWithFeatures(y, x, blockID, blockRotate)

DrawShadow(y, x, blockID, blockRotate)

DrawBlock(y, x, blockID, blockRotate,' ')

: (y,x)에 있는 블록과 그림자를 함께 그린다. 시간복잡도 : O(HEIGHT), 공간복잡도 : O(1)

2) 2개의 블록 미리 보여주기

InitTetris() : nextBlock[2]=rand()%7; 가 추가되고, (두 번째 next블록 정보 초기화)

DrawBlock()이 DrawBlockWithFeatures(blockY,blockX,nextBlock[0],blockRotate); 로 교체되었다.

시간복잡도 : O(HEIGHT\*WIDTH), 공간복잡도 : O(1)

BlockDown() :

nextBlock[0] = nextBlock[1];

nextBlock[1] = nextBlock[2];

nextBlock[2] = rand()%7;

nextBlock의 값 갱신부분이 위와 같이 변경되었다. (두 번째 next블록 랜덤 선택)

DrawBlock()이 DrawBlockWithFeatures(blockY,blockX,nextBlock[0],blockRotate); 로 교체되었다.

시간복잡도 : O(HEIGHT\*WIDTH), 공간복잡도 : O(1)

DrawNextBlock(nextBlock)

for i = 0 to 3

move(4+i,WIDTH+13)

for j = 0 to 3

if( block[nextBlock[1]][0][i][j] == 1 )

attron(A\_REVERSE)

printw(" ")

attroff(A\_REVERSE)

else printw(" ")

for i = 0 to 3

move(10+i,WIDTH+13)

for j = 0 to 3

if( block[nextBlock[2]][0][i][j] == 1 )

attron(A\_REVERSE)

printw(" ")

attroff(A\_REVERSE)

else printw(" ")

: 두 개의 next블록을 그리게 변경되었다. (nextBlock[1], nextBlock[2])

시간복잡도 : O(1), 공간복잡도 : O(1)

3) 닿은 면적만큼 score 증가하기

AddBlockToField(f , currentBlock, blockRotate, blockY, blockX)

touched=0

for i = 0 to BLOCK\_HEIGHT-1

for j = 0 to BLOCK\_WIDTH-1

if (block[currentBlock][blockRotate][i][j] == 1)

if (blockY+i+1>=HEIGHT)

touched++

else if (f[blockY+i+1][blockX+j] == 1)

touched++

for i = 0 to BLOCK\_HEIGHT-1

for j=0 to BLOCK\_WIDTH-1

if (block[currentBlock][blockRotate][i][j] == 1)

f[blockY+i][blockX+j] = 1

return touched\*10

: 블록의 밑면이 테트리스의 바닥에 닿거나, 이미 쌓인 블록 위에 닿는 경우

닿은 칸 수\*10점을 반환하게 변경되었다. 시간복잡도 : O(1), 공간복잡도 : O(1)

BlockDown() :

AddBlockToField(field,nextBlock[0],blockRotate,blockY,blockX); 에서

score += AddBlockToField(field,nextBlock[0],blockRotate,blockY,blockX); 로 변경되었다.

(AddBlockToField에서 계산된 점수를 score에 합산함)

시간복잡도 : O(HEIGHT\*WIDTH), 공간복잡도 : O(1)