**학과: 컴퓨터공학과 학년: 2 학번: 20191657 이름:최세은**

1. **실습 시간에 작성한 프로그램의 함수들이 예비보고서에서 작성한, 각 구현 함수들의 pseudo code와 어떻게 달라졌는지 설명하고, 시간 및 공간 복잡도를 보이시오.**

우선 전체적으로 예비보고서에서 작성한 pseudo code와 실습 때 작성한 실제 code와 크게 다르지 않았다.

1. CheckToMove

pseudo code와 c 언어의 문법 차이로 인해 달라진 것 빼고는 변한 게 없었다. 그저 해당 블록이 정해진 위치로 이동하는게 가능한지 검사하는 함수이고, 이를 위해 따져야 할 항목들도 적어서 차이가 없었다. 이 함수의 시간 복잡도는 이중 반복문이 변수와 상관없이 무조건 최대 16번까지 실행되기 때문에 O(1)이다. 또한 이미 주어진 2차원 배열로 함수가 실행되기 때문에 공간 복잡도는 O(n^2)이다.

1. DrawChange

이 함수도 예비보고서에서 작성한 pseudo code와는 큰 차이가 없지만, 우선 pseudo code에서는 switch문으로 command를 구분했던 것과 달리, 함수에서는 if문으로 command를 구분했다. 그리고 이전의 블록을 지우는 코드가 함수에서 추가되었다. 이 함수는 과제로 인해 함수 내에서 DrawBlockWithFeatures 함수도 사용하고 있기 때문에 그것까지 고려했을 때 시간복잡도는 O(n^2) 이다. 또한 공간복잡도는 위 CheckToMove 함수랑 같이 미리 생성된 2차원 배열을 이용하므로 O(n^2)이다.

1. AddBlockToField

pseudo code와 실제 구현한 함수 둘 다 이중 반복문을 사용하며 그 안에 if문이 있는 것 까지 같지만, if문으로 따지는 조건이 늘어났다. 우선 현재 블록의 정보를 배열에서 찾아낸 후, 블록의 위치가 필드 내에 있을 때 블록이 추가된 영역의 필드값을 바꿔준다. 이 함수의 시간복잡도는 위 CheckToMove 함수와 마찬가지로 변수의 값과 관계없이 최대 16번의 반복문이 실행되기 때문에 O(1)이며, 미리 선언된 2차원 배열을 사용하므로 공간복잡도도 O(n^2)이다.

1. DeleteLine

반복문으로 맨 마지막 줄이 꽉 차있을 때, 그 줄을 지우고 점수를 추가하는 등 문법을 제외하면 모든 내용이 같으나, 실제 코드에서는 그 점수를 return까지 해주었다. 이 함수에서도 이중 반복문을 사용하고 있으므로 시간복잡도는 O(n^2) 이고 공간복잡도도 위 함수들과 같이 O(n^2)이다.

1. BlockDown

다른 함수들과 마찬가지로 pseudo code와 거의 일치하나 다음 블록을 만드는 nextblock 배열을 더 구체적으로 코드를 짰고, 마지막에 time\_out 변수로 시간이 계속 흐르게 해주었다. 이 함수는 다른 함수들과는 다르게 반복문을 사용하지 않고, 다른 함수들을 함수 내에서 사용하였다. 이때 시간 복잡도는 함수 내에 쓰여진 다른 함수들을 고려하여 O(n^2)가 되고, 공간복잡도는 다른 함수와 마찬가지로 미리 선언된 2차원 배열을 사용하므로 O(n^2)가 된다.

1. **테트리스 프로젝트 1주차 숙제 문제를 해결하기 위한 pseudo code를 기술하고, 시간 및 공간 복잡도를 보이시오.**
2. 그림자 기능

void DrawShadow(int y, int x, int blockID,int blockRotate)

while y<HEIGHT and CheckToMove(field,blockID,blockRotate,y+1,x)==1

y++;

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

반복문을 사용하므로 이 함수의 시간복잡도는 O(n)이다. 또한 위의 다른 함수들과 마찬가지로 공간복잡도는 O(n^2)가 된다.

1. 두 개의 블록 미리 보여주기

void DrawNextBlock(int \*nextBlock)

for i = 0; i < 4; i++

move(10+i,WIDTH+13)

for j = 0; j < 4; j++

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

attron(A\_REVERSE)

printw(" ")

attroff(A\_REVERSE)

else printw(" ");

우선 이미 구현된 함수 DrawNextBlock을 수정하여 next 블록을 한 개가 아닌 두개를 보여주게 했다. 이 때 시간복잡도는 이중 반복문을 사용했지만 변수의 값과 관계없이 최대 16번 반복하므로 O(1), 공간복잡도는 타 함수와 마찬가지로 O(n^2)이다.

1. 닿은 면적만큼 score 증가시키기

void AddBlockToField(char f[HEIGHT][WIDTH],int currentBlock,int blockRotate, int blockY, int blockX)

int s = 0

for int i = 0;i < 4;i++

for int j = 0;j < 4;j++

if block[currentBlock][blockRotate][i][j]

if 0 <= blockY + i and blockY + i <= HEIGHT

if 0 <= blockX + j and blockX + j <= WIDTH

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

if f[blockY + i][blockX + j] or blockY + i + 1 == HEIGHT

s++

score += s \* 10

미리 구현된 AddBlockToField 함수를 수정하여 블록이 바닥에 닿았을 때 닿은 면적만큼 score을 증가시킨다. 따라서 따로 변수를 추가하여 그 변수에 닿은 면적의 크기를 기록한다. 이 때 시간복잡도는 원래 함수와 같이 O(1), 공간복잡도도 O(n^2)이다.