8주차 결과보고서

전공: 경영학과 학년: 3학년 학번: 20190963 이름: 한다현

**1.**

CheckToMove() 함수는 pseudo 코드와 같다. 이중 for loop으로 구성되어 있으며 i와 j는 0부터 3까지 반복한다. block[currentBlock][blockRotate][i][j]가 1이면 조건문을 실행한다. x좌표가 0보다 작거나 WIDTH를 넘어가는 경우, y좌표가 HEIGHT를 넘어가는 경우, 그리고 field[i+blockY+1][j+blockX]가 1인 경우에는 0을 반환하고 그 외의 경우에는 1을 반환한다. 이 함수는 시간복잡도가 입력값에 의존하지 않고, 반복문의 실행횟수가 정해져 있으므로 항상 동일한 시간복잡도를 갖는다. 따라서 CheckToMove() 함수의 시간 복잡도는 O(1)이다. 또한 이 함수에서는 고정 공간만 사용하고 있으므로 공간 복잡도는 O(1)이다.

DrawChange() 함수는 switch 문을 사용하여 command의 값을 파악한다. 그 후 이중 for loop을 통해 이전 블록을 지운다. pseudo 코드에서는

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

field[j + blockY][i + blockX] = 0;

라고 작성하였지만 실습에서는 다음과 같이 수정하였다.

*for* (i = 0; i < 4; i++)

*for* (j = 0; j < 4; j++)

            {

*if* (block[*currentBlock*][*blockRotate*][i][j] == 1 && i + *blockY* >= 0)

                {

                    move(i + *blockY* + 1, j + *blockX* + 1);

                    printw("%c", '.');

                }

            }

pseudo 코드의 경우 이미 쌓여있는 블록까지 지워지는 오류가 있었기 때문에 DrawBlock 함수의 코드를 참조하여 수정하였다. switch 문이 종료된 후에는 DrawField() 함수를 통해 필드를 그리고, DrawBlock() 함수를 통해 블록을 새로 그렸다. 이 함수도 for loop에서 i와 j가 고정된 값을 가지므로 항상 동일한 횟수의 반복을 실행하기 때문에 시간복잡도는 O(1)이다. 또한 고정공간만 사용하고 있기 때문에 공간 복잡도는 O(1)이다.

BlockDown() 함수의 pseudo 코드에서는 블록을 아래로 이동할 수 있는 경우의 코드를 다음과 같이 작성하였다.

if CheckToMove() == 1 then blockY++

실습시간에 작성한 코드에서는 blockY의 값을 1 증가시킨 후 DrawChange() 함수를 호출하는 코드를 추가하였다. 또한 pseudo 코드에서는 DeleteLine() 함수를 호출하기만 하였는데, 실습시간에 작성한 코드에서는 score = DeleteLine()와 같이 작성하여 DeleteLine 함수에서 반환하는 점수는 score 변수에 저장하였으며, DrawField() 함수로 필드를 그린 후 DrawBlock() 함수를 통해 새로운 블록을 그리도록 코드를 작성하였다. 마지막으로 함수의 마지막 부분에 timed\_out 변수를 0으로 초기화하는 코드를 추가하였다. 이 함수는 특정 변수에 의존하고 있지 않으므로 시간 복잡도는 O(1)이고, 고정 공간을 사용하기 때문에 공간 복잡도는 O(1)이다.

AddBlockToField() 함수는 pseudo 코드와 동일하다. 이중 for loop으로 구성되어 있고, i와 j는 0부터 3까지 반복한다. 안쪽 for loop에서는 block[currentBlock][blockRotate][i][j]가 1인 경우 field[blockY+i][blockX+j]를 1로 변경하는 작업을 실행한다. 이 함수는 특정 변수에 의존하고 있지 않으므로 시간 복잡도는 O(1)이고, 고정 공간을 사용하기 때문에 공간 복잡도는 O(1)이다.

DeleteLine 함수도 pseudo 코드와 동일하다. 두 변수 flag와 line을 0으로 초기화한다. 이 후 이중 for loop이 실행되는데, i는 0부터 HEIGHT-1, j는 0부터 WIDTH-1까지 반복한다. field[i][j]가 0이면 두 번째 for loop을 중지하고, 1인 경우에는 flag를 1 증가시킨다. 두 번째 for loop이 종료되면 flag의 값이 WIDTH와 같은지 확인하고, 같다면 line을 1 증가시킨 후 다시 이중 for loop을 통해 위의 블록들을 한칸씩 밑으로 내리는 작업을 실행한다. 모든 for loop이 종료되면 line\*line\*100을 반환한다. 이 함수는 i와 j에 의존하고 있으므로 시간 복잡도는 O(n^2)이고, 이차원 배열에 대한 데이터를 수정하기 때문에 공간복잡도는 O(n^2)이다.

**2.**

void DrawShadow()

while CheckToMove() == 1 then y++

DrawBlock(y, x, blockID, blockRotate, ‘/’)

반복문이 변수 y에 의존하므로 시간 복잡도는 O(n)이고, 고정공간만 사용하므로 공간복잡도는 O(1)이다.

void DrawBlockWithFeatures()

DrawBlock()

DrawShadow()

특정변수에 의존하지 않으므로 시간복잡도는 O(1)이고, 고정공간만 사용하므로 공간복잡도는 O(1)이다.

void InitTetris()

nextBlock[2] = rand % 7 추가

이 함수는 반복문이 i와 j에 의존하고 있으므로 시간 복잡도는 O(n^2)이고, 고정 공간을 사용하기 때문에 공간 복잡도는 O(1)이다.

void DrawNextBlock()

for i=0 to 4, i++

move to 11+I, WIDTH+13

for j=0 to 4, j++

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

then “ “ reverse print

else “ “ print

추가

특정 변수에 의존하고 있지 않으므로 시간 복잡도는 O(1)이고, 고정 공간을 사용하기 때문에 공간 복잡도는 O(1)이다.

void BlockDown()

nextBlock[0] = nextBlock[1]

nextBlock[1] = nextBlock[2]

nextBlock[2] = rand() % 7

로 수정,

AddBlockToField()를 score += AddBlockToField()로 수정

특정 변수에 의존하고 있지 않으므로 시간 복잡도는 O(1)이고, 고정 공간을 사용하기 때문에 공간 복잡도는 O(1)이다.

int AddBlockToField()

int touched = 0

for i=0 to 4, i++

for j=0 to j++

if block[currentBlock][blockRotate][i][j] == 1 then

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

then touched++

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

return touched\*10

특정 변수에 의존하고 있지 않으므로 시간 복잡도는 O(1)이고, 4차원 배열을 사용하기 때문에 공간 복잡도는 O(n^4)이다.