8주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211522 이름: 김정환

**1.**

.................

1. CheckToMove : 탐색을 하는 블록의 크기는 4\*4로 하나씩 연산을 수행하므로 시간 복잡도는 O(1)이다. 함수 내에서 변수에 따라 새로 생성되는 부분은 없으나 인자로 전달받을 때 f가 HEIGHT와 WIDTH에 영향을 받으므로 공간 복잡도는 O(HEIGHT \* WIDTH)이다.

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

for i = 0 to 3

for j = 0 to 3

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

if(blockY + i >= HEIGHT or blockX + j >= WIDTH or blockX + j < 0)

return FALSE

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

return FALSE

return TRUE

스크린샷이(가) 표시된 사진

자동 생성된 설명

Pseudo code와 실제 구현 코드의 차이점으로는 조건들을 연결할 때 or이나 and를 ||, &&를 나타내서 연결하였고, TRUE와 FALSE를 각각 1과 0으로 나타낸 점에서 차이가 있다.

2. DrawChange : Command에 의해 변경될 칸만 변경한다. 따라서 기존의 블록의 삭제나 DrawBlock이나 4\*4만큼의 반복이므로 시간 복잡도는 O(1), 공간 복잡도는 f가 HEIGHT와 WIDTH에 영향을 받으므로 O(HEIGHT \* WIDTH)이다.

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

int x = blockX, y = blockY, z = blockRotate

switch(command)

case KEY\_UP: z = (z + 3) % 4

case KEY\_DOWN: y -= 1

case KEY\_LEFT: x += 1

case KEY\_RIGHT: x -= 1

for i = 0 to 3

for j = 0 to 3

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

print “ “ to field (y + i, x + j)

DrawBlock(blockY, blockX, currentBlock, blockRotate, ‘ ‘)

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

DrawChange에서의 구현 차이로는 실제 코드에서 블록이 그려지는 위치는 (i + y + 1, j + x + 1)이므로 기존 블록 삭제 시 move 함수로 해당 위치로 이동 후 prtinw로 “.”을 출력하는데 차이가 있다.

3. AddBlockToField : AddBlockToField는 블록의 크기만큼 field에 추가해주므로 4\*4만큼의 반복 횟수로 f array의 값을 변경한다. 따라서 시간 복잡도는 O(1), 공간 복잡도는 O(HEIGHT \* WIDTH)이다.

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

for i = 0 to 3

for j = 0 to 3

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

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

스크린샷, 텍스트, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

실제 구현은 Pseudo Code와 큰 차이 없으나 각각의 좌표에 대해서 범위 판정만 추가하였다.

4. DeleteLine : for loop를 HEIGHT \* WIDTH만큼 도는 것을 확인할 수 있다. 따라서 시간 복잡도는 O(HEIGHT \* WIDTH)이고, 새롭게 저장하는 값도 배열의 형태는 아니지만 f를 인자로 받으므로 공간 복잡도는 O(HEIGHT \* WIDTH)이다.

int DeleteLine(char f[HEIGHT][WIDTH])

cnt = 0

for i = 0 to HEIGHT – 1

flag = TRUE

for j = 0 to WIDTH – 1

if f[i][j] == 0

flag = FALSE

break

if flag == TRUE

cnt += 1

for p = i to 1(downward)

for q = 0 to WIDTH – 1

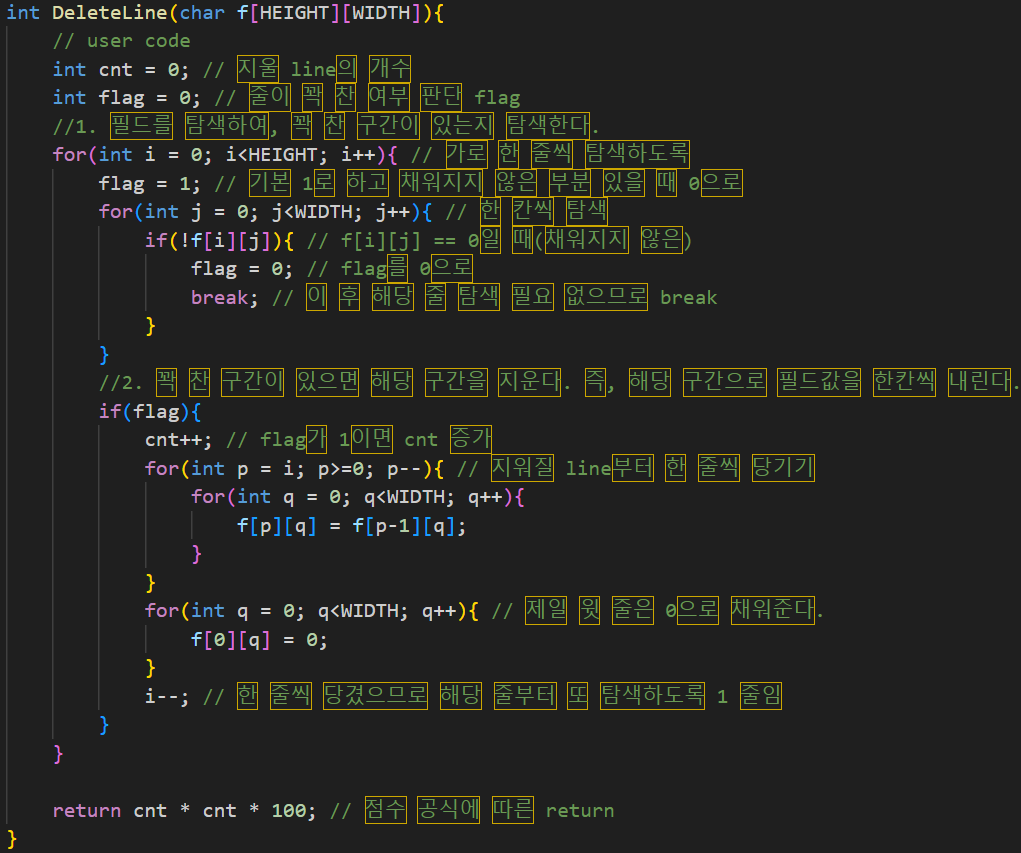
f[p][q] = f[p – 1][q]

for q = 0 to WIDTH – 1

f[0][q] = 0

i -= 1

return cnt \* cnt \* 100



Pseudo Code와 구현에 있어 큰 차이가 없다.

5. BlockDown : 호출되는 함수들의 시간 복잡도를 확인하면 CheckToMove가 O(1), DrawChange가 O(1), DrawField의 시간 복잡도가 O(HEIGHT \* WIDTH)이고 그 외에는 반복문 등의 실행은 없으므로 시간 복잡도는 O(HEIGHT \* WIDTH)이다. 공간 복잡도는 새롭게 변경하는 부분이 blockY++을 제외하고는 특별히 없으므로 O(1)이다.

void BlockDown(int sig)

if CheckToMove(f, currentBlock, blockRotate, blockY + 1, blockX)

blockY += 1

DrawChange(f, command, currentBlock, blockRotate, blockY, blockX)

else

if blockY == -1

gameOver = TRUE

else

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

score += DeleteLine(f)

nextBlock[0] = nextBlock[1]

nextBlock[1] = random integer(0 to 6)

blockY = -1

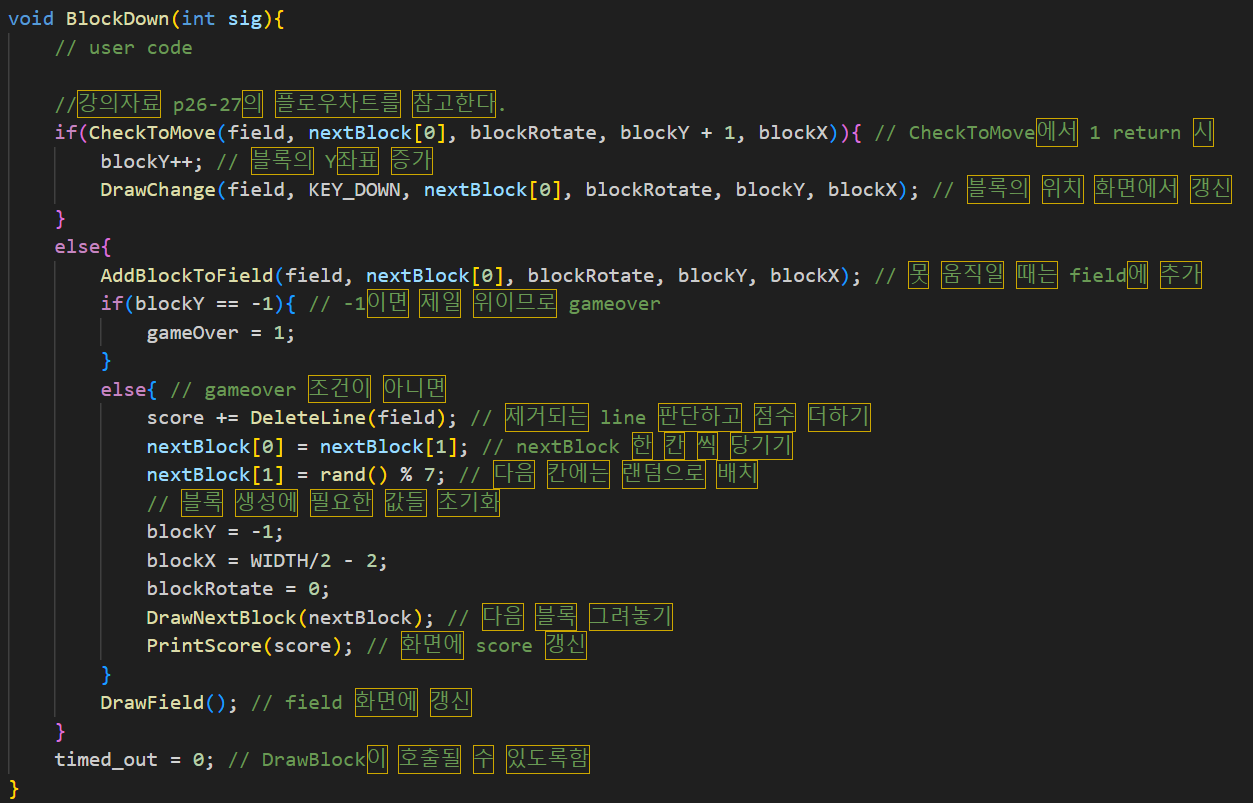
blockX = WIDTH/2 – 2

blockRotate = 0

DrawNextBlock(nextBlock)

PrintScore(score)

DrawField()



구현에 있어서 차이점으로는 if 문 뒤에 DrawBlock을 실행할 수 있도록 하는 timed\_out을 0으로 초기화해주는 부분 등에 대한 차이가 있다.

................

**2.**

.......................

void DrawBlock(int y, int, x, int blockID, int blockRotate, char tile)

for i = 0 to 3

for j = 0 to 3

if block[blockID][blockRotate][i][j] == 1 and i+y >= 0

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

print tile

move cursor to (HEIGHT, WIDTH + 10)

4 \* 4만큼의 반복에서 연산 및 출력한다. 시간 복잡도는 O(1)이고, 공간 복잡도는 O(1)이다.

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

temp\_y = y

while CheckToMove(field, nextBlock[0], blockRotate, y + 1, x)

temp\_y += 1

DrawBlock(temp\_y, x, blockID, blockRotate, ‘/’)

while loop가 최대 HEIGHT만큼 계산되므로 시간 복잡도는 O(HEIGHT), 공간 복잡도는 저장되는 데이터의 양의 변화가 temp\_y 밖에 없으므로 O(1)이다.

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

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

DrawShadow(y, x, blockID, blockRotate)

DrawBlock의 시간 복잡도가 O(1)이고, DrawShadow의 시간 복잡도가 O(HEIGHT)이므로 시간 복잡도는 O(HEIGHT)이고, 공간 복잡도는 둘 다ㅏ O(1)이므로 O(1)이다.

void InitTetris()

int i,j;

for(j=0;j<HEIGHT;j++)

for(i=0;i<WIDTH;i++)

field[j][i]=0;

nextBlock[0]=rand()%7

nextBlock[1]=rand()%7

nextBlock[2]=rand()%7

blockRotate=0

blockY=-1

blockX=WIDTH/2-2

score=0

gameOver=0

timed\_out=0

DrawOutline()

DrawField()

DrawBlockWithFeatures (blockY,blockX,nextBlock[0],blockRotate)

DrawNextBlock(nextBlock)

PrintScore(score)

nextBlock[2]를 추가하였고, 원래는 field를 2중 for loop로 초기화하는 부분이 있으므로 시간 복잡도는 O(HEIGHT \* WIDTH), 공간 복잡도는 새로 정의하는 부분은 없으므로 O(1)이다.

void DrawNextBlock(int \*nextBlock)

for i = 0 to 3

move cursor to (4 + i, WIDTH+13)

for j = 0 to 3

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

print tile

for i = 0 to 3

move cursor to (second box + 1 + i, WIDTH + 13)

for j = 0 to 3

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

print tile

그리는 과정 모두 3 \* 3 형태의 반복문이므로 시간 복잡도는 O(1), 공간 복잡도도 새로 할당하거나 인자로 전달받는 부분이 없으므로 O(1)이다.

void BlockDown(int sig)

if CheckToMove(f, currentBlock, blockRotate, blockY + 1, blockX)

blockY += 1

DrawChange(f, command, currentBlock, blockRotate, blockY, blockX)

else

if blockY == -1

gameOver = TRUE

else

score += AddBlockToField(f, currentBlock, blockRotate, blockY, blockX)

score += DeleteLine(f)

nextBlock[0] = nextBlock[1]

nextBlock[1] = nextBlock[2]

nextBlock[2] = random integer(0 to 6)

blockY = -1

blockX = WIDTH/2 – 2

blockRotate = 0

DrawNextBlock(nextBlock)

PrintScore(score)

DrawField()

기존의 함수에서 시간 복잡도와 공간 복잡도가 달라질 변화는 없었으므로 시간 복잡도는 O(HEIGHT \* WIDTH), 공간 복잡도는 O(1)이다.

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

cnt = 0

for i = 0 to 3

for j = 0 to 3

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

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

if blockY + i == HEIGHT – 1

cnt += 1

return cnt \* 10

cnt를 연산하는 과정이 포함되었지만 변수에 따라 달라지는 부분은 아니므로 시간 복잡도는 O(1), 공간 복잡도는 O(HEIGHT \* WIDTH)이다.

.........................