6주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20191619 이름: 이동석

**1. Pseudo code 비교**

/\* timed\_out, gameOver, score are global variables \*/

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

/\* return TRUE if block can move \*/

int w,h

for h=0 to 3

for w=0 to 3

if block[currentBlock][blockRotate][h][w]==1

if the block is already there return FALSE

else if out of height and width return FALSE

endif

Endfor

Endfor

return TRUE

}

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

int w,h;

for ( h=0; h<4; h++)

{

for (w=0; w<4; w++)

{

if (block[currentBlock][blockRotate][h][w])

{

if (f[h+blockY][w+blockX]) return 0;

else if ( h+blockY >= HEIGHT || w+blockX < 0 || w+blockX >= WIDTH ) return 0;

}

}

}

return 1;

}

예비보고서에서 작성한 의사코드로 함수를 실제로 구현했다. Peseudo code와 달라진 점은 없다. 조건을 코드로 구현했다.CheckToMOVE 함수의 시간복잡도는, 블록의 크기 h\*w 만큼 반복 수행하고, 각 명령의 시간복잡도는 상수이므로, O(h\*w) 라 할 수 있다. 상수에 가깝기 때문에 사실상 O(1)로 봐도 무방하다. 공간복잡도는 전역변수 f에 영향을 받는다고 볼 수 있다. 공간복잡도는 O(HEIGHT\*WIDHT) 라 할 수 있다.

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

/\* update the field by command \*/

int w,h, preRotate

CASE OF command

KEY\_UP :

KEY\_DOWN :

KEY\_RIGHT :

KEY\_LEFT : delete the previous block

OTHER : break;

ENDCASE

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

}

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

int w,h;

int preRotate=(blockRotate+3)%4;

switch (command)

{

case KEY\_UP :

for ( h=0; h<4; h++)

{

for ( w=0; w<4; w++)

{

if (block[currentBlock][preRotate][h][w]) {

move(blockY+h+1,blockX+w+1);

printw(".");

}

}

}

break;

case KEY\_DOWN :

for (h=0; h<4; h++)

{

for (w=0; w<4; w++)

{

if (block[currentBlock][blockRotate][h][w])

{

move(blockY+h,blockX+w+1);

printw(".");

}

}

}

break;

case KEY\_RIGHT :

for (h=0; h<4; h++)

{

for (w=0; w<4; w++)

{

if (block[currentBlock][blockRotate][h][w])

{

move(blockY+h+1, blockX+w);

printw(".");

}

}

}

break;

case KEY\_LEFT :

for (h=0; h<4; h++)

{

for (w=0; w<4; w++)

{

if (block[currentBlock][blockRotate][h][w])

{

move(blockY+h+1, blockX+w+2);

printw(".");

}

}

}

break;

default :

break;

}

DrawBlock(blockY, blockX, currentBlock, blockRotate,' ');

}

DrawChange 함수 역시 Peseudo code와 달라진 점은 거의 없다. 커맨드에 따른 이전 블록의 필드를 업데이트하는 부분을 코드로 작성했다.이 함수의 시간복잡도는 switch 문에서 반복문과 DrawBlock에서의 반복문 모두 블록의 크기 h\*w 만큼 반복 수행하고, 각 명령의 시간복잡도는 상수이므로, O(h\*w)이다. 공간 복잡도 역시 블록의 크기만큼인 O(h\*w)이다.

void BlockDown(int sig){

/\* blocs fall every second \*/

timed\_out=0 // To make it fall automatically

if the block can move down

blockY++

DrawChange(field, KEY\_DOWN, nextBlock[0], blockRotate, blockY, blockX)

else if the block touch the top

gameOver=TRUE

return

else

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

score+=DeleteLine(field)

nextBlock[0]=nextBlock[1]

nextBlock[1]=rand()%7 // To make next block

Initialize the variables blockRotate, blockY, blockX

DrawNextBlock(nextBlock)

DrawField()

PrintScore(score)

endif

}

void BlockDown(int sig){

timed\_out=0;

if (CheckToMove(field, nextBlock[0], blockRotate, blockY+1, blockX))

{

blockY++;

DrawChange(field, KEY\_DOWN, nextBlock[0], blockRotate, blockY, blockX);

}

else if (blockY==-1)

{

gameOver=TRUE;

return;

}

else

{

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

score+=DeleteLine(field);

nextBlock[0]=nextBlock[1];

nextBlock[1]=rand()%7;

blockRotate=0; blockY=-1; blockX=WIDTH/2 -2;

DrawNextBlock(nextBlock);

DrawField();

PrintScore(score);

}

}

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

/\* add block in field \*/

int w,h

for h=0 to 3

for w=0 to 3

if block[currentBlock][blockRotate][h][w]==1

f[blockY+h][blockX+w]=1

endif

Endfor

Endfor

}

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

/\* Delete the line and raise the score \*/

int w,h,temp,flag,line=0

for h= HEIGHT-1 to 0

flag=1

for w=0 to WIDTH-1

if the line is not full

flag=0

break

Endfor

if flag==1

line++

for temp=h-1 to 0

for w=0 to WIDTH-1

f[temp+1][w]=f[temp][w]

Endfor

Endfor

for w=0 to WIDTH-1

f[0][w]=0

Endfor

h++

endif

Endfor

return line\*line\*100

}

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

int w,h;

for (h=0; h<4; h++)

{

for (w=0; w<4; w++)

{

if (block[currentBlock][blockRotate][h][w]) // 블록을 쌓는다

{

f[blockY+h][blockX+w]=1;

}

}

}

//Block이 추가된 영역의 필드값을 바꾼다.

}

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

int w,h;

int flag; // flag

int line=0; // 지울라인 개수

for ( h= HEIGHT-1; h>=0; h--) // 아래에서부터 탐색하므로 거꾸로

{

flag=1; // 초기화

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

{

if (f[h][w]!=1)

{

flag=0;

break; // 하나라도 1이아니면 break

}

}

if (flag) // 현재 h에 지울라인이 있음

{

line++; // 라인개수 세주고 필드값 내리기

for (int temp=h-1; temp>=0; temp--) // 맨 윗줄은 따로 0으로 초기화

{

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

f[temp+1][w]=f[temp][w];

}

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

f[0][w]=0;

h++; // 그 줄부터 다시 검사해야함

}

}

return line\*line\*100;

}

BlockDown함수는 일정시간마다 블록을 아래로 한칸 씩 내린다. 내릴 수 없을 경우 밑에 나올 AddBlockToField 함수를 이용한다. 시간복잡도는 CheckToMove, AddBlockToField, DrawBlock,DeleteLIne, DrawNextBlock, DrawField 함수들과 관련이 있다. 이때, DrawField와 DeleteLIne에서 시간복잡도 O(HEIGHT\*WIDTH) 이고 나머지 함수는 블록의 크기 O(h\*w)이다. 공간복잡도는 필드의 크기인 O(HEIGHT\*WIDHT) 와 블록의 크기 O(h\*w)이다.

DeleteLine는 줄을 체크하고 삭제하여 점수를 반환하며, AddBlockToField는 더 이상 움직일 수 없는 블록을 필드에 반영한다. 시간복잡도와 공간복잡도는 eleteLIne에서 시간복잡도와 공간복잡도 O(HEIGHT\*WIDTH) 이고, AddBlockToFiled 에서 O(h\*w)를 가진다.

**2. 과제 Pseudo code**

과제에서 구현하고 수정해야하는 함수는 다음과 같다.

1. AddBlockToField 에서 점수 계산방법을 수정한다.

2. BlcokDown에서 AddBlockToField 에서 계산한 점수를 누적한다.

3. 그림자 기능을 구현하기 위해, DrawShadow 함수를 구현한다.

4. DrawBlockWithFeatures 함수를 통해 그림자와 블록을 갱신한다.

5. 2개의 블록을 그리기 위해, DrawOutline과 InitTetris, DrawNextBlock,

BlockDown 함수를 수정한다.

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

/\* add block in field \*/

int w,h

for h=0 to 3

for w=0 to 3

if block[currentBlock][blockRotate][h][w]==1

if (f[blockY+h+1][blockX+w] || h+blockY+1==HEIGHT) touched++;

f[blockY+h][blockX+w]=1

endif

Endfor

Endfor

Return touched \* 10;

}

시간복잡도와 공간복잡도의 변화는 없다.

void BlockDown(int sig){

/\* blocs fall every second \*/

timed\_out=0 // To make it fall automatically

if the block can move down

blockY++

DrawChange(field, KEY\_DOWN, nextBlock[0], blockRotate, blockY, blockX)

else if the block touch the top

gameOver=TRUE

return

else

score+=AddBlockToField(field, nextBlock[0], blockRotate, blockY, blockX)

score+=DeleteLine(field)

nextBlock[0]=nextBlock[1]

nextBlock[1]=nextBlock[2] // To make next block

nextBlock[2]=rand()%7 // To make second block

Initialize the variables blockRotate, blockY, blockX

DrawNextBlock(nextBlock)

DrawField()

PrintScore(score)

endif

}

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

/\* update the field by command \*/

int w,h, preRotate

CASE OF command

KEY\_UP :

KEY\_DOWN :

KEY\_RIGHT :

KEY\_LEFT : delete the previous block

OTHER : break;

ENDCASE

DrawField();

DrawBlockWithFeatures(blockY,blockX,currentBlock,blockRotate);

}

DrawNextBlock에서 그려야 하는 블록이 하나 추가되어 O(h\*w\*2) 만큼의 시간복잡도를 가지게 된다. 공간복잡도의 경우, nextBlock이 하나 더 추가되어 역시 O(h\*w\*2) 이다. DrawChange의 경우 그림자를 반영하기 위해 DrawField와 DrawBlockWithFeatures 함수가 추가되어 공간복잡도와 시간복잡도가 필드의 크기인 O(HEIGHT\*WIDHT) 가 된다.

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

DrawShadow(y,x,blockID,blockRotate);

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

}

그림자와 블록을 갱신해준다. DrawShadow함수와 DrawBlock 함수를 호출한다. 시간복잡도와 공간복잡도는 DrawShadow 함수와 관련있다.

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

int count=y;

while the block can move

count++;

DrawBlock(count,x,blockID,blockRotate,'/');

}

DrawBlock 함수는 O(h\*w) 이므로, DrawShadow의 경우 시간복잡도는 CheckToMove의 시간복잡도\*count만큼 추가된다. CheckToMove 의 경우 시간복잡도는 O(h\*w) 이며 count는 최대 HEIGHT라 볼 수 있으므로, O(h\*w\*HEIGHT) 가 된다.

void DrawNextBlock(int \*nextBlock){

int i, j;

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

move(4+i,WIDTH+13);

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

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

attron(A\_REVERSE);

printw(" ");

attroff(A\_REVERSE);

}

else printw(" ");

}

}

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

move(11+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(" ");

}

}

}

DrawNextBlock 함수의 경우 위치에 맞게 블록을 하나 더 그려준다. 앞서 말했듯이 그려야 하는 블록이 하나 추가되어 O(h\*w\*2) 만큼의 시간복잡도와 공간복잡도를 가지게 된다.