8주차 결과 보고서

전공: 컴퓨터공학 학년: 2학년 학번: 20221559 이름: 박준우

1. checkToMove

예비보고서에서는 x, y에서 4x4가 판 안에 들어가는지를 확인했는데, 실제로는 블럭 중 대부분이(12칸) 빈칸이기 때문에, 4x4중 실제로 블럭인 부분이 판 안에 들어가는지를 확인하도록 바뀌었다.

그럼에도 4x4 전체에 대해 4번씩 for loop가 두번 돌아가서 시간복잡도는 정해진 시간, O(1)이다.

공간복잡도 또한 기존 선언된 배열 참조, for loop를 위한 변수선언 정도밖에 이루어지지 않아 O(1)이다.

2. DrawChange

이전 위치, 방향을 판단하는 코드는 맞게 작성하였다. 그러나 DrawBlock 함수 이용해서는 배경 없이 .만 찍는것이 불가능해 해당 함수 내의 for loop를 가지고 와서 작성했다.

이전 위치와 방향을 판단하는것도 상수시간, 지우고 다시 그리는 것도 2중 for문이나, 16번 반복되는 것이 정해져 있어 마찬가지로 O(1)이다.

공간복잡도도 마찬가지로 O(1)이다.

3. BlockDown

강의자료의 flow chart를 보고 작성해서 순서에 틀린 것은 없고, timed\_out을 호출시 0으로 바꿔서, game()함수의 while문 안에서 1초 후 호출 하도록 바꿔주는 부분이 누락되었다.

CheckToMove(), DrawChange()는 항상 실행되고, AddBlockToField(), DeleteLine(), DrawBlock() 함수는 블럭이 아래로 이동 가능하면 실행된다. 다른 함수들은 모두 O(1)이나 DeleteLine 함수가 O(height\*width) 시간이 걸리기 때문에 O(H\*W) 시간이 된다.

공간복잡도도 DeleteLine() 함수에서 게임판을 복사하도록 작성했기 때문에 O(R\*C)이다.

4. AddBlockToField

블럭이 있는 자리에서만 해당하도록 각 for loop을 바꾼 것 말고는 변화가 없다.

이 위치는 valid하다는 것이 blockDown에서 보장하고 있다.

마찬가지로 O(1) 시간복잡도, O(1) 공간복잡도를 가진다.

5. DeleteLine

구현하고자 하던 대로 구현하였다. 마지막에 DrawField()를 호출해주는 부분 제외하고는 의사코드와 결이 동일하나, 지워진 줄 수를 반환하는 부분은 의사코드에 작성하지 않았었다.

게임 판의 모든 곳을 탐색해야 하므로 O(height \* width) 시간복잡도를 가지고, 함수 내에서 원래 게임 판을 전체 복사하여서 O(height \* width) 공간복잡도를 가진다.

6. 그림자 기능을 구현하기 위한 함수들, 기존 함수의 변형

DrawShadow() 함수는 해당 위치의 해당 블럭이 아래로 어디까지 내려갈 수 있는지를 확인한 후, 그곳에 블럭을 그리면 된다.

기존에 사용하던 DrawBlock() 함수를 DrawBlockWithFeatures()로 바꾸고, 이 함수 내에서 DrawBlock을 호출하도록 변경한다.

이때 이전의 그림자를 지우도록, DrawChange에서의 변경도 필요하다.

DrawShadow

while(CheckToMove(Under))

y++;

DrawBlock(y, x, block)

CheckToMove의 실행횟수는 height에 bound 되므로, O(height) 시간복잡도를 가지고, 별도의 변수 선언이 없으므로, O(1) 공간복잡도를 가짐을 알 수 있다.

7. 다음 블록의 개수를 3개로 늘려라

NextBlock의 개수를 정의하고 있던 헤더파일의 BlockNum을 3으로 변경하고, init시 셋 다 설정해주고, 테두리와 다음 블럭 모양을 그려주는 함수, 블록 위치를 고정하고 다음 블럭을 설정하는 BlockDown 함수를 수정하면 된다.

Init과, drawoutline, drawNextBlock은 기존 코드의 반복이므로, 위치를 잘 맞춰서 바꿔주면 된다.

BlockDown

…

nextBlock[0] = nextBlock[1];

nextBlock[1] = nextBlock[2];

nextBlock[2] = rand() % 7;

로 변경이 끝이다.

시간 복잡도 및 공간 복잡도는 상수가 커진 것에 불과해 그대로이다.

8. 점수에 매번 놓는 블록이 아래로 맞닿은 부분의 개수 \* 10을 더해주어라

AddBlockToField 함수를 조금 수정해 주면 된다. 기존은 아무것도 return 하지 않았는데, Block을 추가할 때, block이 있는 칸에 대해 바로 아래 칸이 field를 벗어나거나 (맨 아래 줄), 이 칸에 원래 있었다면 세어 준 후, 이것의 10배를 return 한다.

BlockDown에서 return 받아서 score에 더해주면 된다.

For each element in block

If below is out of range || below == 1 : ret ++;

Return ret;

시간복잡도나, 공간복잡도 상의 변화는 없다. AddBlockField는 마찬가지로 O(1), O(1)을 가진다.