10주차 결과보고서

전공 : 컴퓨터공학과 학년 : 2학년 학번 : 20211558 이름 : 윤준서

**1. 알고리즘과 자료구조**

텍스트, 스크린샷, 문서, 폰트이(가) 표시된 사진

자동 생성된 설명

X, Y, R 변수를 통해 추천 Block의 x좌표, y좌표, 회전 상태를 저장한다. max는 최대 얻을 수 있는 점수를, child는 자식 노드의 인덱스를 저장한다. root의 level이 BLACK\_NUM이 되면 함수를 종료한다.

for(R = 0; R < 4; R++)와 for(X = -2; X < 13; X++)을 통해 가능한 Block의 회전 상태와 x좌표를 탐색하여 더 이상 Block이 이동할 수 없을 때까지 진행한다.

이후 새로운 자식 노드를 생성한다. 위와 마찬가지로 Block이 더는 이동할 수 없을 때까지 y좌표를 증가시킨다.

이후 DeleteLine()와 Block이 땅에 닿았을 경우를 통해 점수를 추가한다. 이 때 recommend()를 재귀 호출하여 최대 점수를 구한다. 최종적으로 Block의 좌표와 회전상태, 그리고 최대 점수를 반환한다.

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 대수학이(가) 표시된 사진

자동 생성된 설명

modified\_recommend() 함수는 각 Block마다 회전할 수 있는 경우를 지정하여 탐색하는 시간을 줄인다. 그리고 free()를 통해 불필요한 자식 노드를 해제하여 메모리를 최적화한다. 또한 wall을 추가하여 벽 카운트를 설정해 더 나은 점수 결정이 가능해진다.

기존 recommend()의 시간복잡도는 O(4 \* (WIDTH + 5))이지만 modified\_recommend()의 경우 Block마다 회전할 수 있는 경우의 수를 지정했기에 O((각 블럭의 가능한 회전 수)\* (WIDTH + 5))이다. 공간복잡도는 둘 모두 O(BLOCK\_NUM)이다.

**2. 향상된 점, 그렇지 못한 점**

각 Block마다 회전 경우의 수를 지정하였기에 일일이 반복문을 통해 모든 종류의 Block을 탐색할 필요가 없어 시간 효율성이 향상되었다.

그러나 자식 노드의 level을 3까지 모두 탐색하는 것은 동일하기에 여전히 많은 계산이 요구된다. 이는 일정 max의 크기를 지정하여 해당 임계치를 넘는 자식 노드가 생기는 경우 이후 계산을 모두 생략하고 해당 Block을 선택하는 pruning방식을 통해 해결할 수 있다.

**3. 이번 Tetris 프로젝트를 수행하면서**

Linked List, Tree등 교내에서 배운 알고리즘 개념들을 적극적으로 활용하면서 해당 개념에 대해 더 잘 이해하고 알맞게 활용하는 방법을 습득했다. 처음엔 C언어 만으로 어떻게 게임을 구현할 수 있는지 궁금했었는데, 리눅스 내의 ncurses 라이브러리를 통해 배울 수 있었다. 처음 접하는 라이브러리이다 보니 함수 하나하나가 어색하고 어떻게 써야 할지 몰랐지만, 여러 주차에 걸쳐 계속 사용하면서 익숙해지게 되었다. 앞으로 다른 라이브러리들을 통해 프로젝트를 만들 때 전보다 더 능숙하게 다룰 수 있을 것이다.