그룹 과제 1 - 테트리스

(3784) C++ 프로그래밍

지정희 교수님



**2조 - GRAVITY**

201011260 박진한

201511220 이지현

201611185 권혜지

201611187 김도연

201611244 한상규

**제출일: 2016.11.03.**

문제정의

테트리스

오류를 포함하여 C 언어로 작성된 테트리스 프로그램을 디버깅하고, C++ 문법으로 변환 및 추가 기능을 구현하는 과제이다. 다른 사람에 의해 이미 짜여 있는 코드를 이해하고, 버그를 잡아 내고, 객체 지향 프로그래밍 기법을 최대한 활용하여 전역 변수 및 함수를 사용하지 않고 클래스화 시키는 것이 핵심 목적이다.

오류 발견 및 해결 과정

점수 출력 오류

변수들을 초기화해 주는 함수 init()에서 변수 score가 0으로 초기화되지 않아 오류가 발생하였다.

→ score = 0으로 초기화하여 쓰레기 값이 발생하지 않도록 하였다

블록이 천장에 닿지 않았는데 게임 오버 되는 오류

게임 오버 여부를 체크하기 위해 선언된 변수 is\_gameover는 strike\_check() 함수에 의해 값이 정해진다. strike\_check() 함수는 totalBlock[y + i][x + j]이 1 값을 가질 때 block\_dat에 1을 저장한다. 하지만 이런 코드가 블록이 위에서 떨어질 때 일어나면 의도치 않은 상황에서 is\_gameover가 1을 반환하게 되어 오류가 발생한다.

→ if((y + i) < 0)인 상황에서는 block\_dat에 1을 대입하지 않도록 조건을 추가시켜 주었다.

제거한 라인 수가 음수 값을 갖고 stage가 바로 넘어가지 않는 오류

클래스 설계 이전에 main 함수 안의 for ( i = 1; true; i++)에서 라인을 얼마나 지웠는지 체크하여 한 번에 여러 개의 블록이 지워졌을 경우 삭제된 라인 수가 음수 값을 가지고 따라서 레벨 상승이 바로 바로 이루어지지 않았다.

→ moveBlock()에서 checkFullLine() 함수를 호출하여 라인이 지워졌는지를 확인하고 지운다. checkFullLine() 함수 마지막 부분에

If (lineCheck(0))

{

InitLevel();

initLine(0);

}

위 코드를 추가하여 라인 클리어가 발생할 때마다 레벨을 상승시켜 주어야 하는지 체크해 준다. 이렇게 하면 여러 블록이 지워지더라도 지운 라인 수가 음수가 되는 일이 없고 레벨이 즉시 올라가게 된다.

게임 오버 후 재실행 시 바로 게임 오버가 되는 오류

게임 오버가 되는 조건은 is\_gameover가 1 값을 가지는 경우이다. 한번 게임을 실행하여 게임 오버가 되면 is\_gameover는 1 값을 갖게 되고 단계 선택 화면으로 돌아간다. 이 과정 전에 is\_gameover를 0으로 초기화해 주지 않으면 게임 오버 여부를 체크해 주는 부분에 걸려 바로 게임이 끝나게 된다.

→ 게임 오버를 확인해 주는 부분 마지막에 is\_gameover = 0;을 추가시켜 주어 해결한다.

클래스 설계

Stage

블록이 떨어지는 속도, 막대기가 나오는 확률, 제거해야 할 라인 수 등 스테이지에 관한 변수 및 함수들을 포함한다.

Variables

게임 실행, 블록에 관련된 변수와 함수를 가지고 있다. Stage 클래스를 멤버 변수로 가진다.

주요 추가 기능

미리보기

- showPreBlock()

중력이 원래 방향일 경우(블록이 아래로 낙하하는 경우) strikeCheck() 함수의 for문을 통해 블록 다음 칸부터 바닥까지 블록이 아래 바닥과 부딪힐 수 있는지를 검사한다. 만약 부딪힌다면, 그 블록이 존재하는 위치에서 totalBlock이 0인 바닥(바닥에 블록이 있다면 블록도 바닥으로 취급)에 ‘□’을 출력한다.

중력이 반대 방향일 경우 같은 방법으로 strikeCheck()함수의 for문을 통해 블록 바로 위칸부터 천장까지 검사하여 천장과 부딪힐 수 있는지 검사한다. 부딪힌다면, 그 블록이 존재하는 위치에서 totalBlock이 0인 천장에 ‘□’을 출력한다.

- erasePreBlock()

showPreBlock()을 통해 출력한 블록 미리보기 ‘□’의 위치에 ‘ ‘(공백)을 출력하여 블록 미리보기의 잔상을 지워 준다.

나가기

- Exit()

esc 키를 입력할 경우(start 함수 - esc의 아스키 코드 값 27을 입력) gotoxy() 함수를 통해 커서 위치를 이동시킨 뒤 게임 종료 안내 메시지를 출력하고, '예'를 선택할 시 system("cls") 함수로 화면을 지우고 게임을 종료, 레벨 선택 화면으로 돌아가고, '아니오' 선택 시 esc 키를 누르기 전 화면을 출력한 뒤 게임을 이어가도록 한다. 문자는 \_getch() 함수를 통해 읽어들인다.

중력 방향 바꾸기

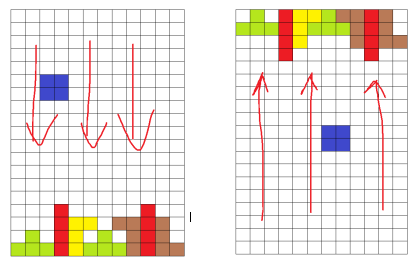
1. 시간 초

이 기능에는 변수 setTime, time1, time2가 필요하다. setTime은 몇 초마다 중력 방향이 바뀔 것인지를 저장한다. 그리고 게임이 시작되면 time1에 time(NULL)을 대입하여 현재 시간 정보를 저장한다. 그리고 for 문을 돌 때마다 time2에 현재 시간을 대입하여 setTime-(time2-time1)==0인지 확인한다. 만약 시간이 다 됐다면 eraseCurBlock()로 현재 내려오고 있는 블록을 지워 준다. 그리고 blockStart()를 호출하여 블록이 다시 천장에서부터 내려오도록 한다. 다시 time1에 현재 시간 정보를 저장하고 reverse 변수를 변경한다.

1. 프레임 체인지

Reverse가 0이었다면 totalblock[0][j]에 있는 블록들을 지워 주고 totalblock[20][j]에 1을 채워 준다. 반대로 Reverse가 1이었다면 totalblock[20][j]에 있는 블록들을 지워주고 totalblock[0][j]에 1을 채워 준다.

1. 블록 정보를 그대로 가지고 중력 방향이 바뀌어야 한다



떨어질 때 당시의 블록 모습을 그대로 기억한 채, 최대한 위 방향으로 옮길 수 있어야 한다. 또, 그 반대의 방향으로도 마찬가지 방식으로 옮길 수 있어야 한다.

기존의 코드는 함수 merge\_bloc이 쌓인 블록이 저장되는 변수 total\_block에다가 빈칸이면 0, 존재하면 1을 저장한다. 이와 같은 방법으로는 merge\_block으로 합쳐지기 전 블록의 모습을 저장할 수 없다. 따라서 C++ 언어로 변경하면서 새로운 변수 blockCounter를 만들어 mergeBlock 함수가 totalBlock에 1 대신 blockCounter를 저장하고 blockCounter를 증가시키도록 만들었다.

쌓여 있는 블록들은 blockCounter로 구분되므로 blockCounter의 시작인 2부터 현 blockCounter까지 반복하여 모든 totalBlock을 for문을 이용해 비교하도록 하고, 특정 값을 가진 모든 칸이 자신의 위에 있는 칸이 이거나 자신과 똑같은 값을 가질 때만 해당 블록이 움직일 수 있으므로 이를 if문을 이용해 검사한다. 만일 if문을 만족하지 못한다면 continue를 이용해 다음 blockCounter를 검사하도록 만들고, continue 뒤엔 해당 블록을 한 칸 위로 올리는 작업을 진행한다. 마지막으로 이 모든 과정은 while문을 통해 더 이상 움직일 수 있는 블록이 없을 때까지 진행한다.

움직일 수 있는 블록이 존재하는가는 앞에서 언급한 if문과 동시에 해당 blockCounter 값을 가진 블록이 현재 존재하는지를 검사하여야 한다. for문에서는 변수 counter를 이용해 해당 blockCounter값을 가지는 값이 있는지 검사하고 만일 없다면 continue문을 이용해 counter를 증가시킨 뒤 다시 검사하고, 있을 시에만 앞서 말한 if문을 통해 해당 블록의 움직임 가능 여부를 판단해야 한다(해당 블록이 없으면 if문에서는 참이 되므로). 현 blockCounter는 언제나 현재 쌓여 있는 블록의 totalBlock의 최대값보다 1 클 수 밖에 없으므로 이 검사에서 얻어진 counter값이 blockCounter와 같아진 채 끝난다면 while문의 조건이 거짓이 되어 반복이 끝나게 만들면 된다.

1. 아래에서 위로 블록 올라가고 블록 쌓기

Reverse가 0이었을 때 블록 y++를 해 주었는데 Reverse가 1이었다면 y—로 변경하여 블록이 위로 올라가게 해 주고, y > 19일 때 1을 리턴하여 게임 오버가 되도록 한다.

커서 숨기기

CursorView() 함수를 이용하여 커서가 불필요하게 깜빡거리지 않도록 커서를 숨긴다.

BGM

PlaySound(TEXT("C:\\tetrix\\gameover.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

게임 시작 화면, 초기 실행 시, 리버스 시 각각 다른 분위기의 배경 음악이 재생되도록 한다.

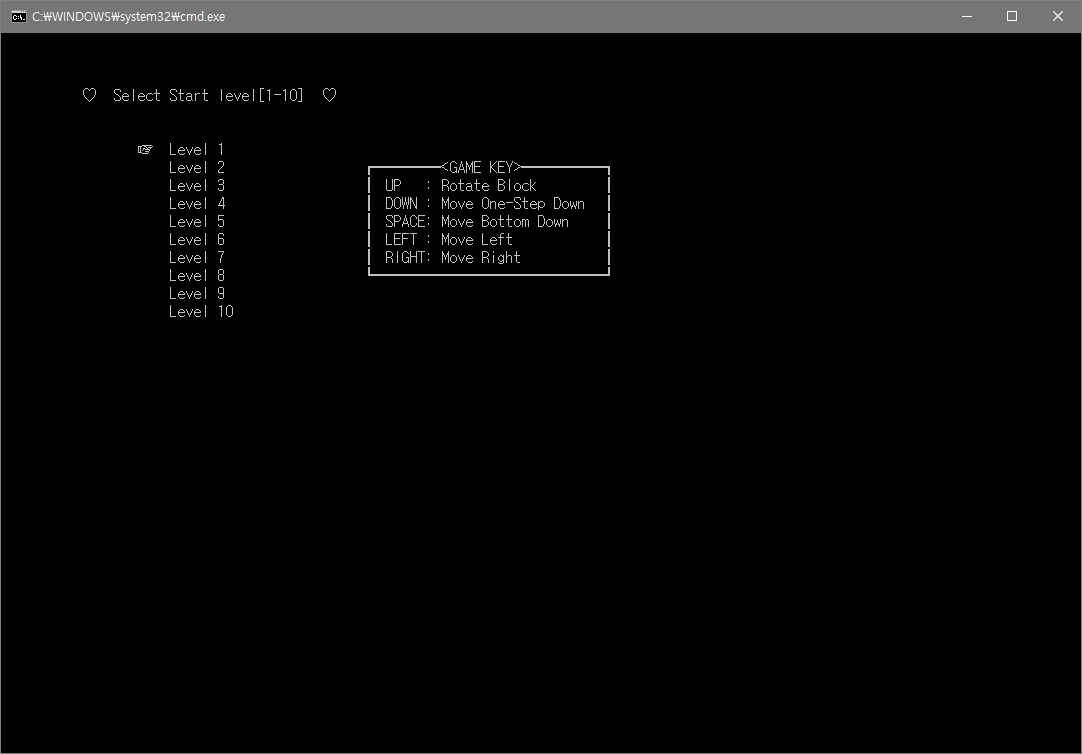
레벨 선택

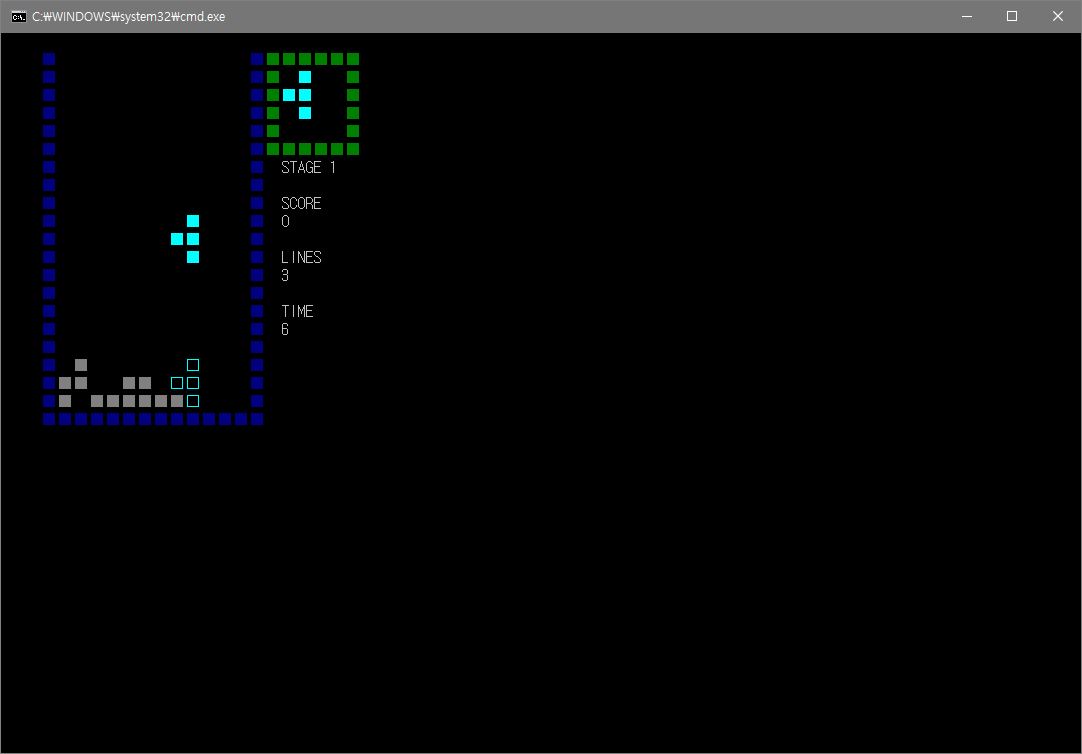
레벨을 키보드로 직접 입력하지 않고 방향 키로 선택할 수 있도록 하였다.

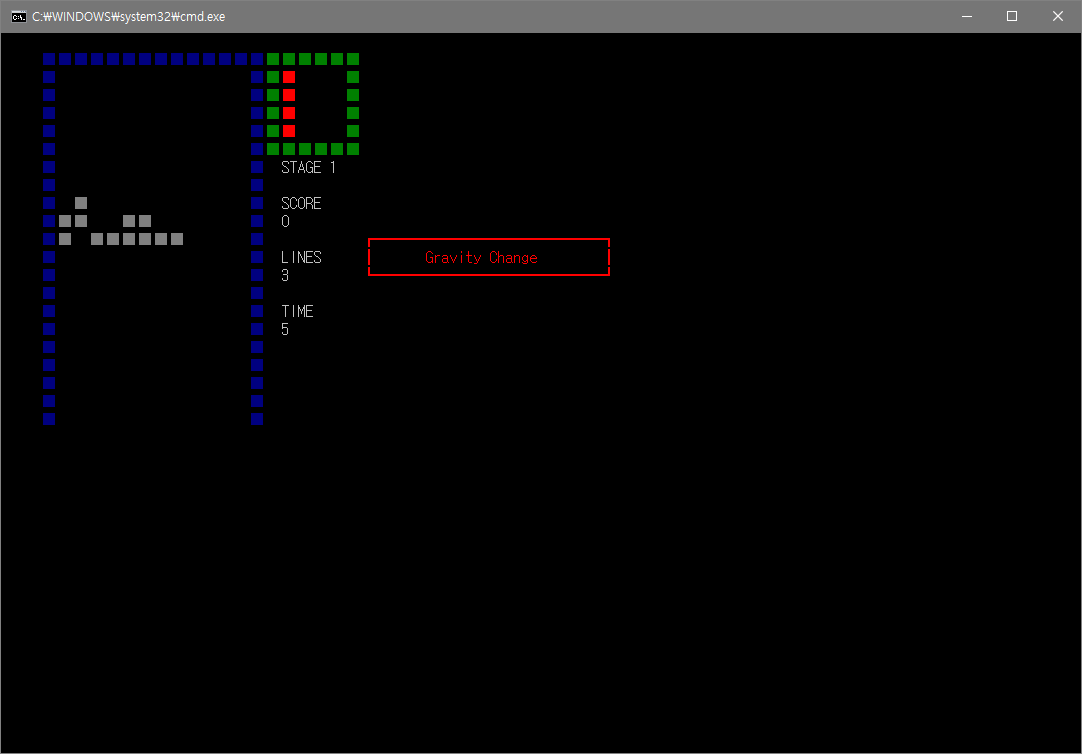
토의사항

‘테트리스’는 누구에게나 익숙한 게임이다. 평소에는 단순히 게임을 즐기기만 했다. 이번 그룹 과제를 수행하면서 별 생각없이 하던 게임의 구동 원리를 이해하면서 단순하게 보이는 기능 하나에도 세심한 코드가 들어간다는 것을 크게 느꼈다. 그리고 테트리스의 기본적인 틀에 추가기능들을 추가하면서 어떻게 하면 새롭고 재미있는 게임을 만들 수 있을지 고민하는 과정에서 게임을 플레이하는 사람들에게 재미를 주는 것이 결코 쉽지 않은 일이라는 것을 알게 되었다.

실행 화면











인증샷

