**2019 C++ Programing Project Report**

- Push Box Game -

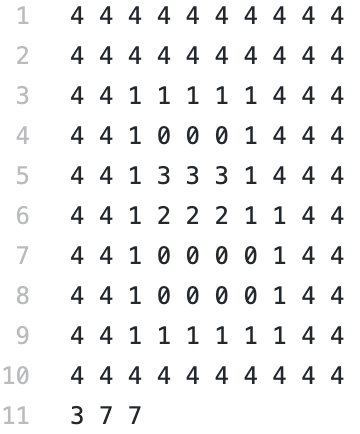
소프트웨어학부 20181569 강주성, 소프트웨어학부 20171658 윤여환

**1. 맵 구현**

우리는 총 10단계의 맵을 구현했는데, map.cpp 파일을 통해 맵을 받아오도록 구현했다. [그림 1]은 우리가 구현한 맵의 1단계 “level1.txt” 이다. 각각의 맵은 이렇게 txt 파일로 저장되어 [그림 2]의 소스코드와 같이 map.cpp 파일에 구현된 mapArray class의 set\_map 멤버 함수를 통해 2차원 배열로 저장이 된다. 단계별로 맵의 사이즈가 다 달랐기 때문에 ncurses 로 출력할 때 일관성있게 출력하기 위해 사이즈를 10\*10으로 통일하고, 맵 바깥의 공백을 4로 통일했다. 나머지 부분은 프로젝트 설명에 주어진 것처럼 벽은 1, 박스는 2, 골은 3, 공백은 0으로 설정해서 2차원 배열에 저장했다. 그리고 [그림 1]의 마지막 line 11은 골 개수와 시작 좌표이다. 각 단계별로 골 개수와 시작 좌표가 다르기 때문에 맵을 불러올 때 골 개수와 시작 좌표도 초기화해주었다.

map.cpp 파일을 통해 받아온 2차원 배열은 game.cpp 파일의 PushBoxGame class의 newGame 멤버 함수를 통해 터미널 창에 보여진다. 2차원 배열 int map[][]을 newGame 멤버함수의 인자로 받아서 ncurses library를 통해 터미널 창에 출력했다. 벽에 해당하는 1은 “#”로 하고, ncurses library의 color 함수들을 사용하여 문자의 색과 배경을 색의 똑같이 만들어서 벽처럼 보이게 구현했다. 박스에 해당하는 2는 “\*”, 골에 해당하는 3은 “@”, 공백에 해당하는 0과 4는 “ “로 공백처리했다. 그리고 [그림 3]의 line 77과 같이 level.txt에서 받아온 시작 좌표 (a, b)에 “O”를 캐릭터로 출력했다.

[그림 1] “level1.txt” 파일



[그림 2] map.cpp 파일의 mapArray class의 set\_map 멤버 함수의 일부



[그림 3] game.cpp 파일의 PushBoxGame class의 newGame 멤버 함수



**2. 움직임 구현**

캐릭터의 움직임은 캐릭터의 이동 방향의 한 칸 앞과 두 칸 앞을 비교하여 아래와 같이 총 7가지의 케이스로 나누어 구현했다.

1. 캐릭터 => 공간
2. 캐릭터 => 박스 => 공간
3. 캐릭터 => 박스 => 골
4. 캐릭터 => 박스 => 벽
5. 캐릭터 => 박스 => 박스
6. 캐릭터 => 벽
7. 캐릭터 => 골

1, 2, 3, 7번 케이스는 캐릭터가 앞을 움직이는 경우이고, 나머지 4, 5, 6번 케이스는 앞으로 움직이지 못하는 케이스이다. 케이스를 나누기 이전에 [그림 4]와 같이 enum 함수를 사용해서 2차원 배열에 게임의 구조물들을 숫자로 저장했던 것을 헷갈리지 않도록 문자로 표현했다. 이것들을 사용해서 조건을 나누었다.

[그림 5]와 같이 main 함수 내부에서는 while(true) 안에서 break가 걸릴 때까지 키를 입력받는다. 보고서에서는 위로 올라가는 케이스를 예시로 보여주겠다. main.cpp 파일의 main 함수에서 “위쪽 화살표”를 입력 받았을 때 PushBoxGame class인 g의 method인 moveUP 을 호출한다. game.cpp 파일에 정의된 moveUP method는 현재 단계의 맵을 인자로 받아온다. 캐릭터의 한 칸 앞과 두 칸 앞의 좌표를 보기 쉽게 정의하기 위해 [그림 6]의 line 82, 83처럼 int up, up2를 한 칸 앞, 두 칸 앞의 좌표를 대입해준다. 현재의 경우는 위로 가는 경우이기 때문에 비교하는 map[][]의 x좌표는 b-3으로 고정되어 있고 y좌표만 up, up2를 사용하여 바꿔준다.

먼저 line 84은 바로 앞에 벽이 있는 경우이니깐 아무 변화 없이 바로 return 해서 입력을 다시 받는다. 마찬가지로 line 86, 87, 104, 105도 움직이지 못하는 경우이니 변화 없이 return 해준다.

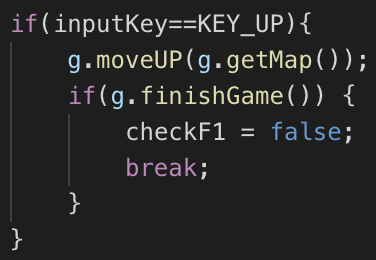
Line 85부터 102까지는 박스를 미는 경우이다. Line 88의 else문이 박스가 밀리는 경우인데 박스가 가게 될 두 칸 앞이 골일 경우에는 두 칸 앞으로 박스가 올라간 골(BoxOnGoal)로 바꿔주고 박스가 있던 위치는 공백(Space)으로 바꿔준다. 그렇지 않은 경우는 두 칸 앞을 박스, 한 칸 앞을 공백으로 바꿔준다.

이런 식으로 조건을 다 확인한 다음 이동하는 방향(설명하는 경우 a가 1 작아지는 경우)으로 값을 변경한다. 그리고 위의 조건에 따라 int map[][] 의 값을 바꿔준 것을 토대로 맵 화면을 갱신해준다. 이런 과정을 상하좌우의 움직임을 구현했다.

[그림 4] PushBoxGame.h 에 구현된 enum 함수



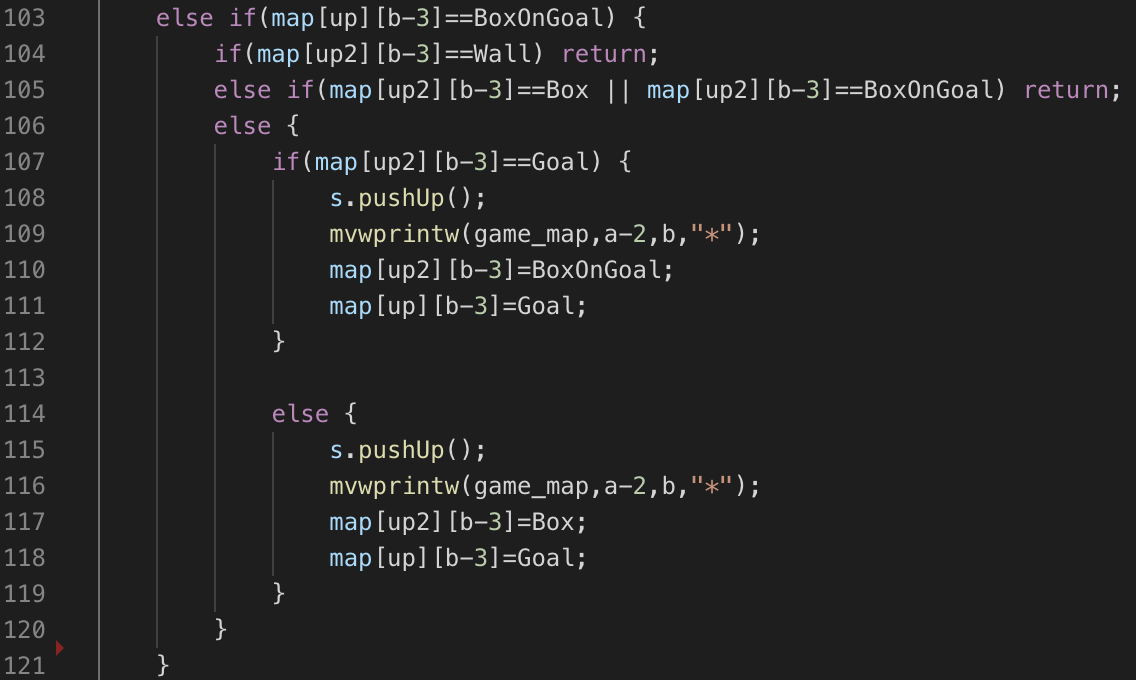
[그림 5] main.cpp 파일의 main 함수 중 “위쪽 화살표”를 키로 입력 받는 부분



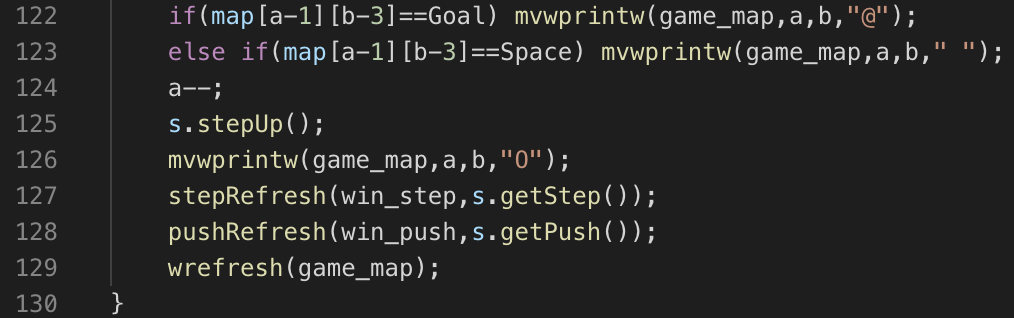
[그림 6] game.cpp 파일의 PushBoxGame의 moveUP 멤버 함수의 일부 (1)



[그림 7] game.cpp 파일의 PushBoxGame의 moveUP 멤버 함수의 일부 (2)



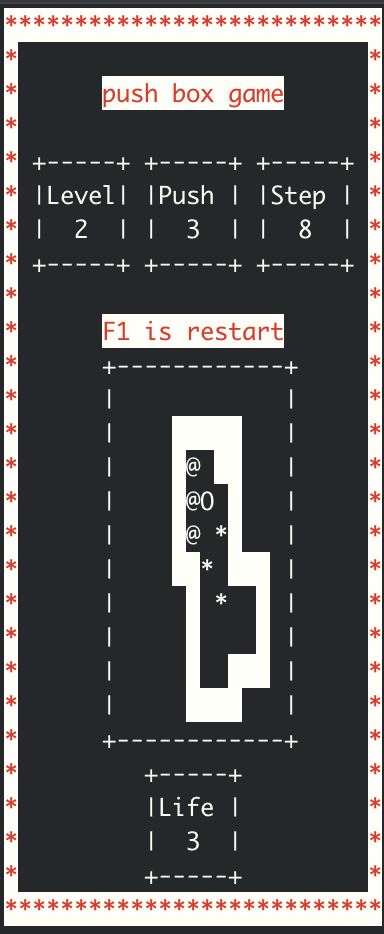
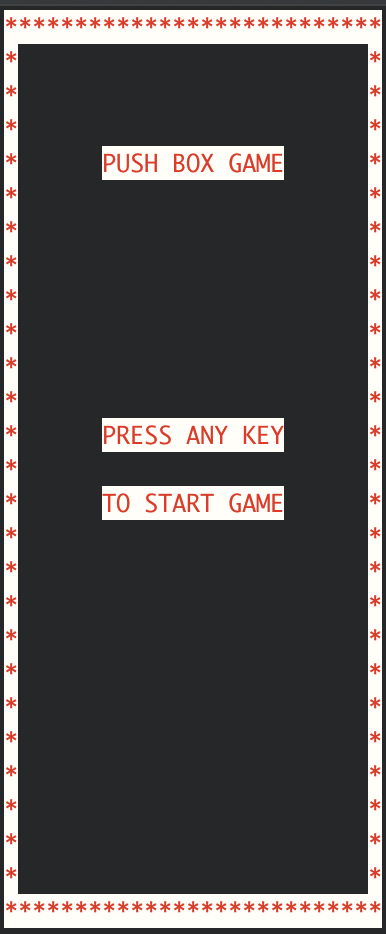
[그림 8] game.cpp 파일의 PushBoxGame의 moveUP 멤버 함수의 일부 (3)



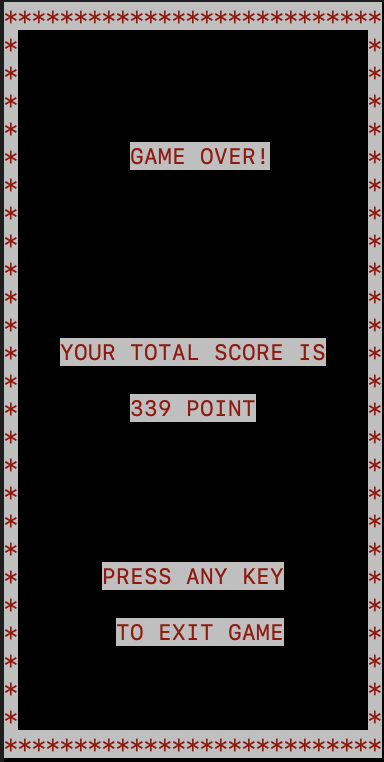
**3. 레이아웃**

게임 맵의 레이아웃은 맵 구현에서 이야기했으니 나머지 부분의 레이아웃을 설명하도록 하겠다. [그림 9]는 게임 시작 화면이다. PushBoxGame class가 처음 호출될 때 하나의 new\_win이라는 이름의 윈도우로 나오도록 구현했다. 게임 실행 화면은 win\_level, win\_push, win\_step, game\_map, win\_life 이렇게 기능 별로 나눠서 한 화면에 나오도록 했다. 그리고 단계 클리어 시 [그림 11]처럼 클리어 화면이 나오고 아무 키나 입력시 다시 게임 실행 화면의 구성 윈도우를 호출해서 [그림 10]과 같이 다시 나오게 구현했다.

[그림 9] 게임 시작 화면 [그림 10] 게임 화면



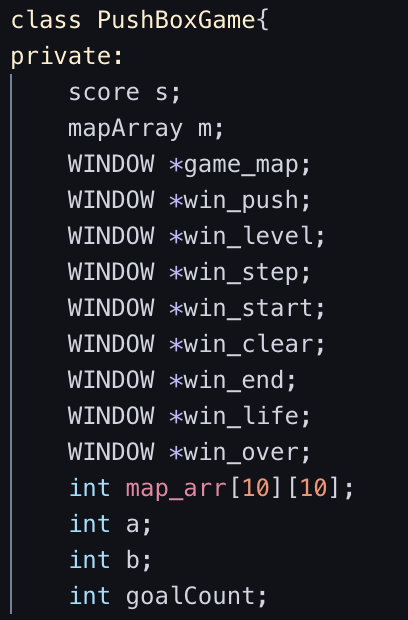
[그림 11] 단계 클리어 화면 [그림 12] 게임 종료 화면



**4. 클래스 나누기와 makefile 만들기**

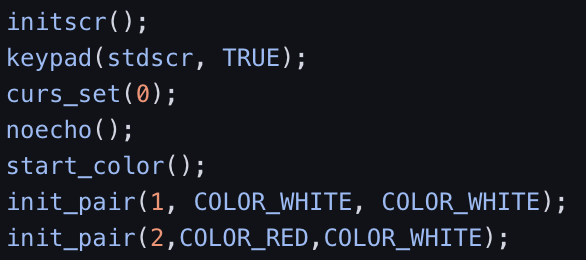
1. 게임에서 기본 레이아웃을 만들고 각종 엔딩화면과 클리어 화면, push,step,life화면을 보여줄 객체들과 함수를 가지고 캐릭터의 시작 좌표와 상자가 목적지에 모두 도착했을 때 게임이 끝난다는 것을 알려주는 변수, 맵을 표시해줄 맵 배열을 가지는 PushBoxGame Class가 있습니다.

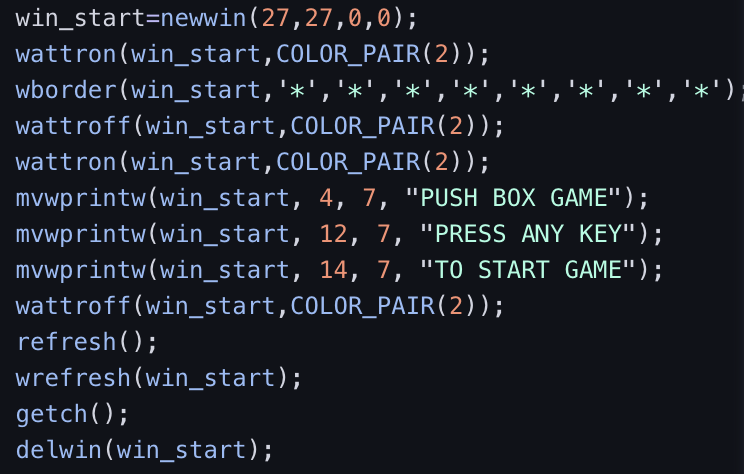
[그림 13] PushBoxGame class 멤버변수

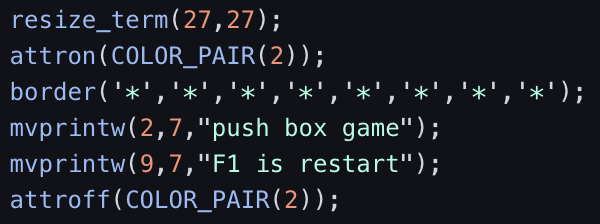


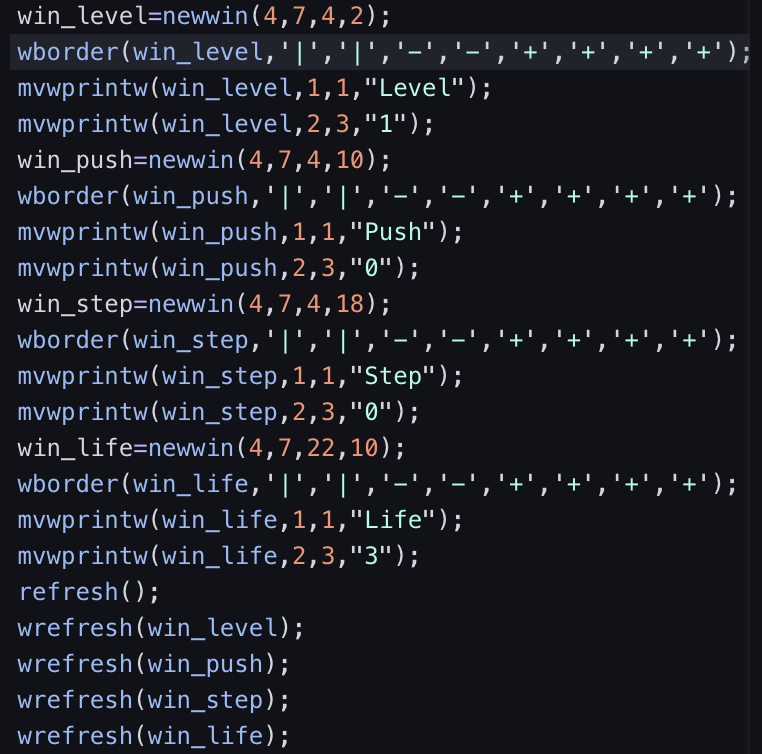
게임이 시작 할 때 나오는 시작 화면과 push,step,level,life화면은 모두 PushBoxGame 생성자에서 만들어준다.

[그림 14] 생성자 함수 내부



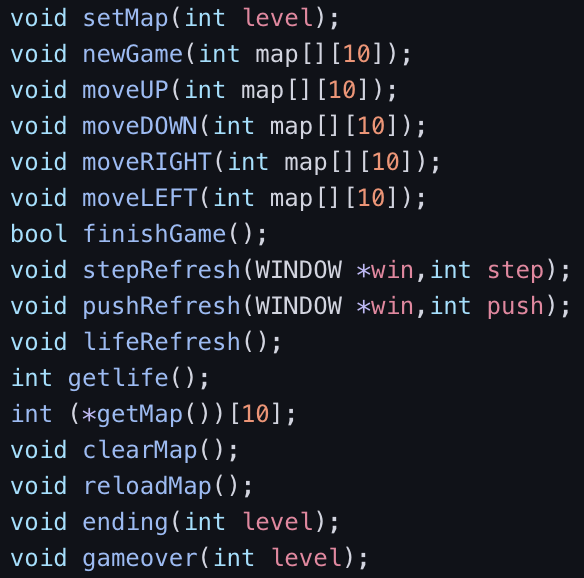






게임을 구현할 때 필요한 함수들이다.

[그림 15] PushBoxGame 함수



- setMap은 level을 넣으면 해당 레벨에 대한 맵 배열을 게임에 이용할 map\_arr에 값을 모두 넣어준다. map class의 객체 m을 이용하여 txt파일로 저장된 값들을 배열에 넣어주는 함수를 이용한다. 또한 a,b에 시작좌표를 넣어주고 goalCount에 목적지의 개수를 넣어준다. Level화면의 level이 바뀌어야 하므로 여기서 setMap이 실행될 때 바뀌도록 해준다.

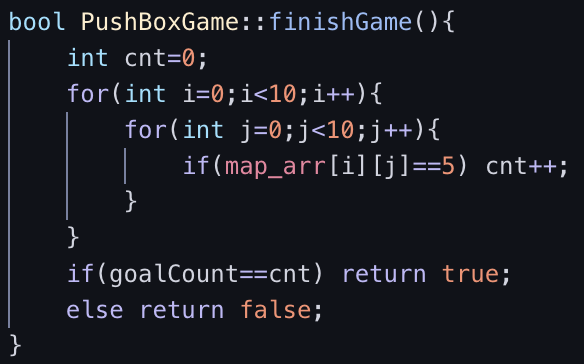
[그림 16] setMap함수 내부



- move함수들은 위에서 움직임을 구현한 코드를 넣은 함수다.

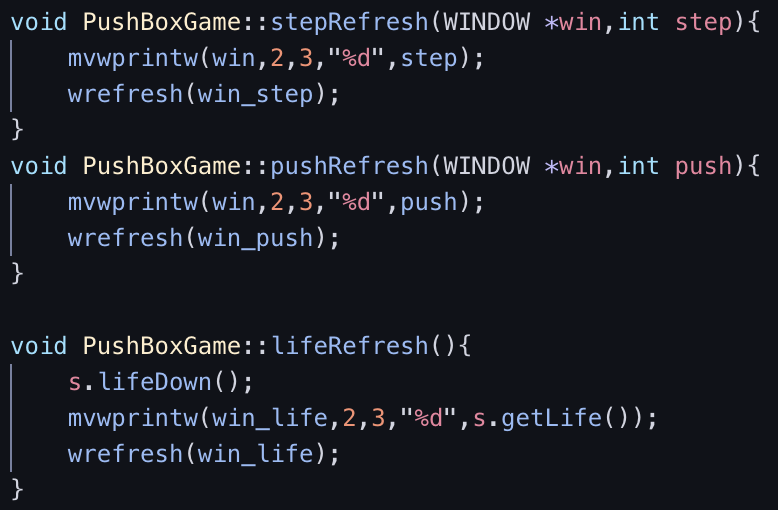
- finishGame은 게임은 각 level의 목적지 개수와 현재 맵 내부에서 box가 goal위에 있는 값이 5이므로 배열에 5가 있는 개수를 검사해서 같다면 해당 ture를 리턴해준다.

[그림 17] finishGame함수 내부



- stepRefresh와 pushRefresh 그리고 lifeRefresh함수는 step값과 push값 그리고 life값이 바뀔 때 마다 화면에 refresh해주는 함수다.

[그림 18] Refresh함수 내부



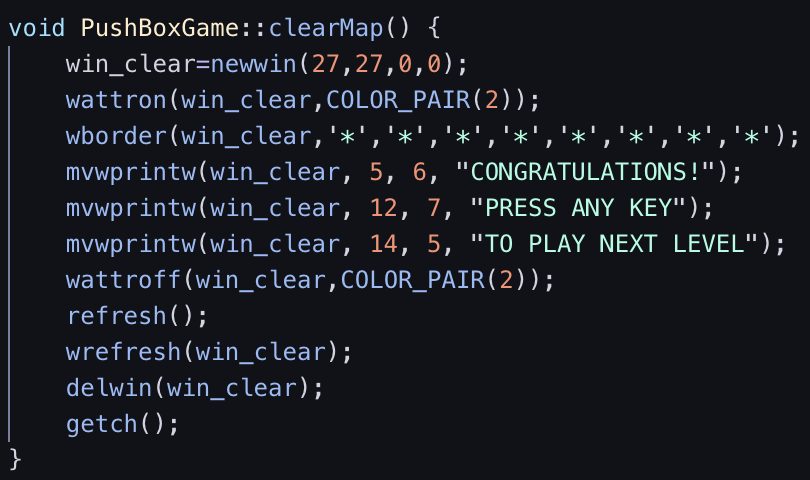
- life,step,push는 score class에서 다룬다. lifeRefresh함수는 함수가 호출 됐을 때 함수 자체에서 life값을 1 감소시키고 출력해준다.

- getlife함수는 life를 보기 위해서는 score class에 있는 private멤버변수에 있는 life변수를 확인해야 하는데 private이므로 getlife함수를 만들어 값을 리턴해 오게 하는 함수다.

- getMap은 map배열을 리턴값으로 하는 함수로 현재 맵을 리턴해온다.

- clearMap은 한단계가 끝날 때 마다 나오는 화면을 구현한 함수다.

[그림 19] clearMpa함수 내부



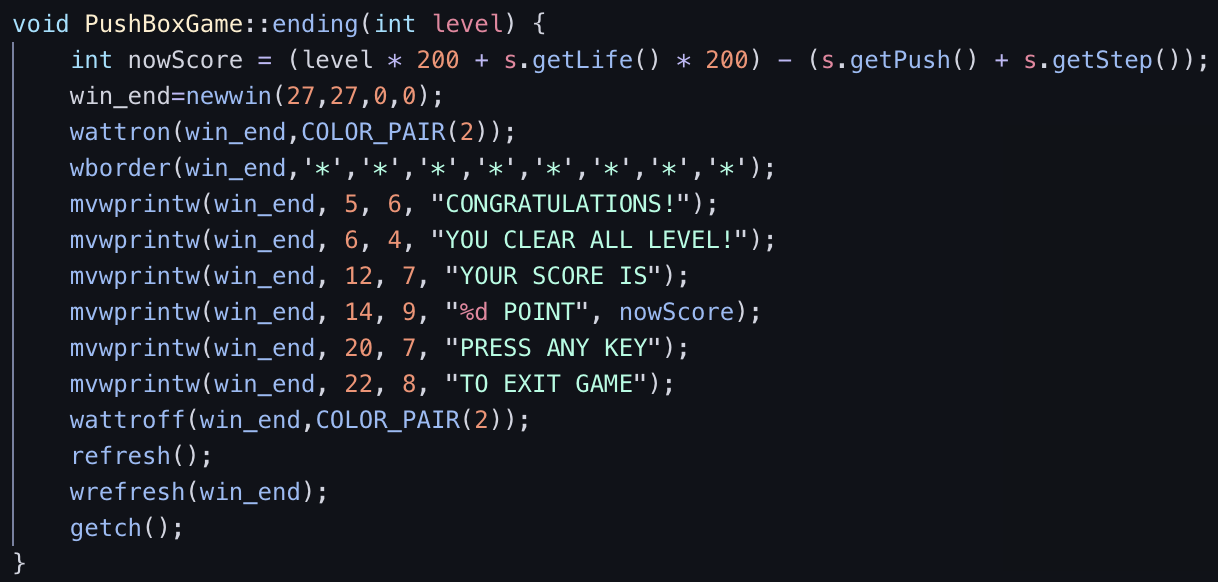
- reloadMap함수는 claerMap함수 이후에 다시 게임 화면을 보여줘야 하므로 만든 함수다.

[그림 20] reloadMap함수 내부



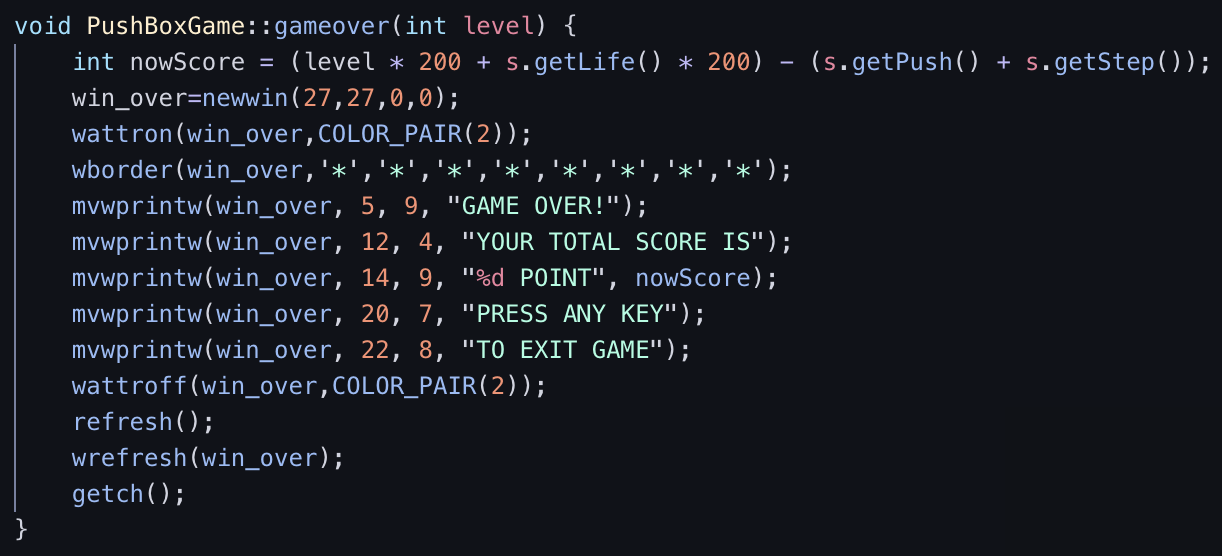
- ending함수는 모든 level을 클리어 했을 때 나오는 화면을 보여주는 함수로 마지막으로 점수를 계산해서 보여준다.

[그림 21] ending함수 내부



- gameover함수는 life를 모두 소진했을 때 나오는 화면을 보여주는 함수로 마지막 점수를 계산해서 보여준다.

[그림 22] gameover함수 내부



2. map을 fstream을 통해 txt파일에서 값을 가져와 저장해주는 함수를 가지는 map class가 있다.

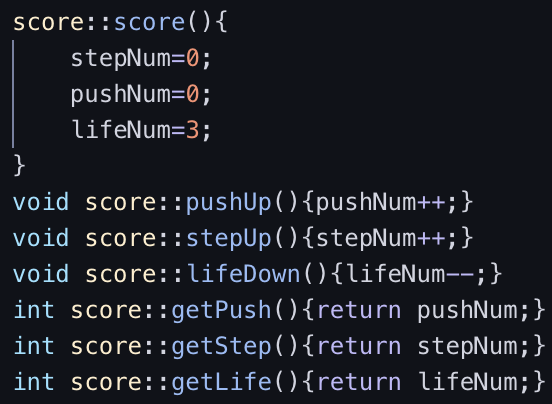
[그림 23] 맵을 불러오는 함수



3. life와 step,push를 관리하는 score class가 있다. 이 class에서는 값들을 리턴해주거나 값을 증가, 감소시켜주는 함수가 있다.

- 생성자에서는 멤버변수를 초기화 해주고 get함수는 값을 리턴, up 또는 down함수는 멤버변수 값을 증가 또는 감소시키는 함수다.

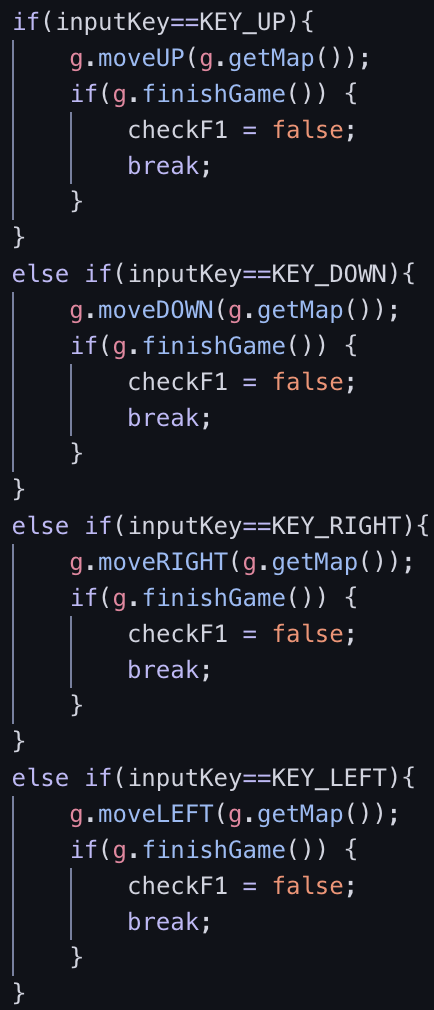
[그림 24] socre 함수

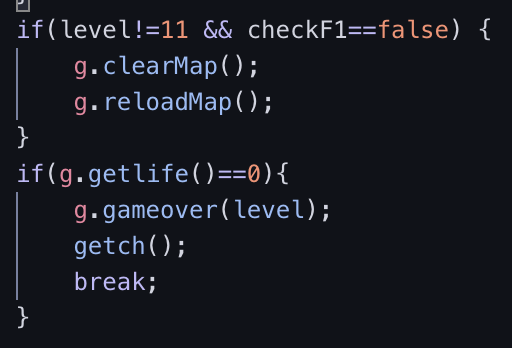


4. main함수에서는 위에서 만든 PushBoxGame 객체를 생성하고 키 입력에 따른 함수를 실행한다. 키 방향에 따른 move함수를 사용하고 F1키를 눌렀을 때 life가 하나 감소하면서 해당 레벨 게임을 다시 시작하게 된다. finshiGame함수를 계속 검사하면서 다음 레벨로 넘어갈지 판별한다. 다음 레벨로 넘어 가기 전 clear함수를 실행하고 life를 검사하면서 life가 모두 감소하면 게임이 끝난다. 모든 level을 클리어하면 ending함수를 사용하여 ending화면을 보여준다.

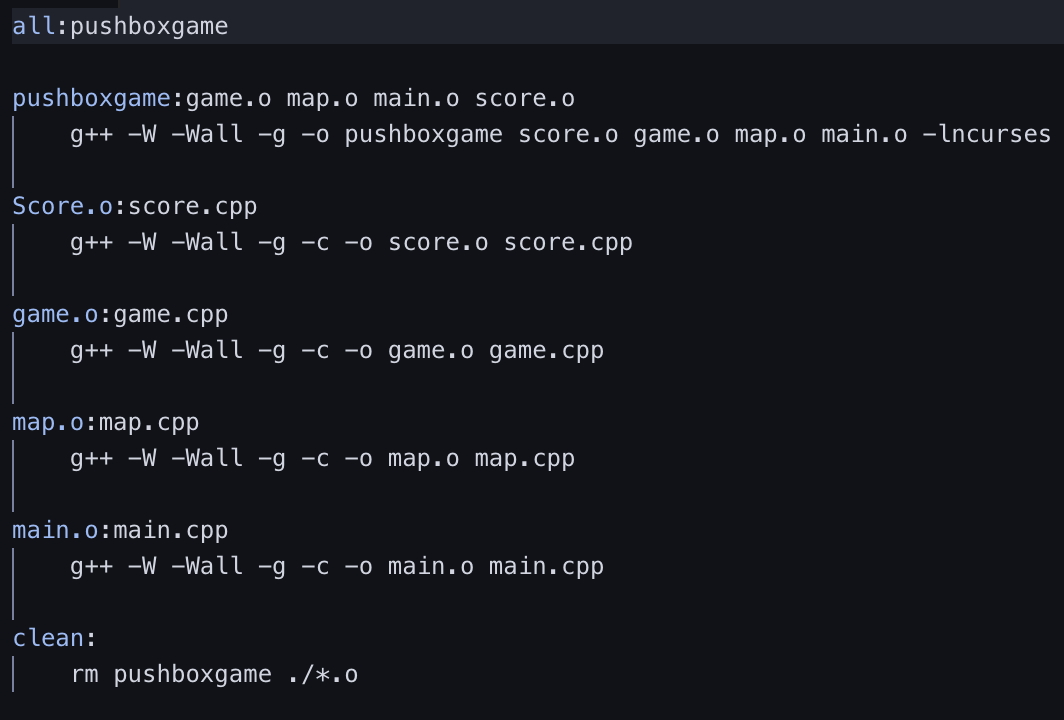
[그림 25] main 함수 내부







5. makefile



**5. 추가 기능 구현**

- F1키를 누르면 현재 레벨을 다시 시작하게 되는데 life를 1감소된다. Life가 0이 되면 게임이 종료 되면서 게임내부에서 계산된 점수를 보여준다. 점수는 level당 200점에서 step수와 push수를 뺀 값이다. 또한 모든 레벨을 클리어하면 추가점수로 life당 200점을 준다. 또한 level 10까지 만들어 더 다양한 맵을 즐길 수 있다.