서경대학교

객체지향프로그래밍 (CS1149 - 03)

소스 코드

2019316034 정지홍

-------------pushpush.cpp-------------

```
#include <iostream>
using namespace std;
#define MAP_X1 5
#define MAP_Y1 5
#include "View.h"
#include "Goal.h"
#include "Maps.h"
#include "Information.h"
#include "Sound.h"
int main() {
	int cntMap = 1;//현재 내가 몇번째 맵인지 저장하는 변수
	int goal = 0;
	int posx = 2, posy = 4;//현재 캐릭터의 위치
	int oldx, oldy;
	_setcursortype(_NOCURSOR);
	textbackground(WHITE);
	clrscr();
	View::drawMap();
	View::drawCharacter(posx, posy);
	int move = 0;//캐릭터의 이동 횟수를 나타내는 변수
	int* pMove = &move;//이동횟수 증가를 위해 포인터를 매개변수로 넘길 예정
	Inform::InforInit();//오른쪽 게임 내용을 알리는 창을 그린다.
	while (1) {

		Inform::mapNum(cntMap);//오른쪽에 몇 번째 맵인지 알리는 창을 그린다..
		Inform::move(pMove, cntMap);//캐력터 이동 횟수를 처리하는 함수

		Play::move();//캐릭터 이동 사운드 효과 처리 함수
		textcolor(CYAN);

		int key = View::getkey();
		int nposx = oldx = posx;
		int nposy = oldy = posy;
		//case 2에서 사용할 위치 포인터(loadMove)
		int* pPosx = &posx;
		int* pPosy = &posy;

		switch (key) {
		case M_LEFTKEY:nposx --; break;
		case M_UPKEY: nposy--; break;
		case M_DOWNKEY: nposy++; break;
		case M_RIGHTKEY:nposx ++; break;
		}
		switch (mapData[nposy][nposx])//위에서 감소시킨 맵 데이터의 좌표기준으로 캐릭터이동 계산
		{
		case 0://path
			posx = nposx; posy = nposy;
			break;
		case 4://Goal
			break;
		case 5://왼쪽으로만 이동 가능
			if ((nposx < posx)) {
				posx = nposx; posy = nposy;
			}
			break;
		case 6://오른쪽으로만 이동 가능
			if (nposx > posx) {
				posx = nposx; posy = nposy;
			}
			break;
		case 3://*
		case 1://==
			break;
/*----------------------------------bomb----------------------------------*/
		case 7:
			if (nposx < posx) {//왼쪽 이동
```

```
                //폭탄 왼쪽좌표에 벽,단방향 이동, 공 인 경우
                if ((mapData[nposy][ nposx-1] == 1) || (mapData[nposy][nposx - 1] == 5) ||
                    (mapData[nposy][nposx - 1] == 6) || (mapData[nposy][nposx - 1] == 2)) {
                    Boom::delWallL(nposx-1 , nposy);//해당 좌표 데이터 변수를 0으로 변경
                    View::drawCell(nposx - 1, nposy);//원래 폭탄 위치 및 파괴한 장해물 좌표 셀을 다시
그린다
                    View::drawCell(nposx, nposy);
                    posx = nposx; posy = nposy;//캐릭터 이동
                    Play::bomb();//폭탄 사운드 효과
                    break;
                }
                //폭탄 왼쪽이 목표물 혹은 완료된 목표물인 경우
                if (mapData[nposy][nposx - 1] == 3 || mapData[nposy][nposx - 1] == 4) {
                    clrscr();
                    Inform::goalDestroyed(cntMap);//게임 종료
                    break;
                }
                /*폭탄 이동 처리*/
                mapData[nposy][nposx - 1] = 7;
                mapData[nposy][nposx] = 0;
                View::drawCell(nposx - 1, nposy);
                View::drawCell(nposx, nposy);
                posx = nposx; posy = nposy;
                break;
            }
            else if (nposx > posx) {//오른쪽 이동
                //폭탄 오른쪽좌표에 벽,단방향 이동, 공 인 경우
                if ((mapData[nposy][nposx+1] == 1)|| (mapData[nposy][nposx + 1] == 6) ||
                    (mapData[nposy][nposx + 1] == 5) || (mapData[nposy][nposx + 1] == 2)) {

                    Boom::delWallR(nposx + 1, nposy);//해당 좌표 데이터 변수를 0으로 변경
                    View::drawCell(nposx +1, nposy);//원래 폭탄 위치 및 파괴한 장해물 좌표 셀을 다시
그린다
                    View::drawCell(nposx, nposy);
                    posx = nposx; posy = nposy;//캐릭터 이동
                    Play::bomb();//폭탄 사운드 효과
                    break;
                }
                //폭탄 오른쪽이 목표물 혹은 완료된 목표물인 경우
                if (mapData[nposy][nposx + 1] == 3 || mapData[nposy][nposx + 1] == 4) {
                    clrscr();
                    Inform::goalDestroyed(cntMap);//게임 종료
                    break;
                }
                /*폭탄 이동 처리*/
                mapData[nposy][nposx + 1] = 7;
                mapData[nposy][nposx] = 0;
                View::drawCell(nposx + 1, nposy);
                View::drawCell(nposx, nposy);
                posx = nposx; posy = nposy;
                break;
            }
            //위로 이동
            else if (nposy < posy) {

                if (mapData[nposy-1][nposx ] == 1 || (mapData[nposy-1][nposx ] == 6) ||
                    (mapData[nposy-1][nposx ] == 5) || (mapData[nposy - 1][nposx] == 2)) {
                    Boom::delWallUp(nposx , nposy-1);
                    View::drawCell(nposx , nposy-1);
                    View::drawCell(nposx, nposy);
                    posx = nposx; posy = nposy;
                    Play::bomb();
                    break;
                }
                if (mapData[nposy-1][nposx ] == 3 || mapData[nposy-1][nposx ] == 4) {
                    clrscr();
                    Inform::goalDestroyed(cntMap);
                    break;
                }
```

```
				mapData[nposy-1][nposx ] = 7;
				mapData[nposy][nposx] = 0;
				View::drawCell(nposx , nposy-1);
				View::drawCell(nposx, nposy);
				posx = nposx; posy = nposy;
				break;
			}
			//아래로 이동
			else if (nposy > posy) {

				if (mapData[nposy + 1][nposx] == 1 || (mapData[nposy+1][nposx ] == 6) ||
					(mapData[nposy+1][nposx ] == 5) || (mapData[nposy + 1][nposx] == 2)) {
					Boom::delWallDown(nposx, nposy + 1);
					View::drawCell(nposx, nposy + 1);
					View::drawCell(nposx, nposy);
					posx = nposx; posy = nposy;
					Play::bomb();
					break;
				}
				if (mapData[nposy + 1][nposx] == 3 || mapData[nposy + 1][nposx] == 4) {
					clrscr();
					Inform::goalDestroyed(cntMap);
					break;
				}
				mapData[nposy + 1][nposx] = 7;
				mapData[nposy][nposx] = 0;
				View::drawCell(nposx, nposy + 1);
				View::drawCell(nposx, nposy);
				posx = nposx; posy = nposy;
				break;
			}
/*-----------------------load---------------------------------------*/
		case 2://load
			textbackground(YELLOW);
			//왼쪽이동
			if (nposx < posx) {
				//공끼리 충돌, 단방향 이동 구간 충돌
				if (Crush::crushBallL(nposy, nposx) == true) {
					break;
				}
				View::loadMoveL(nposx, nposy, pPosx, pPosy);

			}
			//오른쪽 이동
			else if (nposx > posx) {
				//공끼리 충돌, 단방향 이동 구간 충돌
				if (Crush::crushBallR(nposy, nposx) == true) {
					break;
				}
				View::loadMoveR(nposx, nposy, pPosx, pPosy);
			}
			//위로 이동
			else if (nposy < posy) {
				//공끼리 충돌, 단방향 이동 구간 충돌
				if (Crush::crushBallUp(nposy, nposx) == true) {
					break;
				}
				View::loadMoveUp(nposx, nposy, pPosx, pPosy);
			}
			//아래로 이동
			else if (nposy > posy) {
				//공끼리 충돌, 단방향 이동 구간 충돌
				if (Crush::crushBallDown(nposy, nposx) == true) {
					break;
				};
				View::loadMoveDown(nposx, nposy, pPosx, pPosy);
			};

			break;
		}

		if (oldx != posx || oldy != posy) {
			View::drawCell(oldx, oldy);//원래 위치의 심볼을 mapData에서 가져와서 그린다.
			View::drawCharacter(posx, posy);//움작일 위치에 가서 캐릭터를 그린다
		}
/*------------------맵 완료 되었는지 체크한다-------------------------*/
		if ((IsGoal::allGoal(mapData[1][1], mapData[18][8], mapData[2][17],
			mapData[13][14]) == true)&&cntMap==1) {
			MapUpdate::update2();
			View::drawMap();
			posx = 2, posy = 4;
			View::drawCharacter(posx, posy);
			cntMap += 1;
			Play::nextStage();
		};
		if ((IsGoal::allGoal(mapData[1][1], mapData[18][8], mapData[2][17],
			mapData[13][14]) == true)&&cntMap==2) {
			MapUpdate::update3();
			View::drawMap();
			posx = 2, posy = 4;
			View::drawCharacter(posx, posy);
			cntMap += 1;
			Play::nextStage();
		};
		if ((IsGoal::allGoal(mapData[1][1], mapData[18][8], mapData[2][17],
			mapData[13][14]) == true) && cntMap == 3) {
			MapUpdate::update4();
			View::drawMap();
			posx = 2, posy = 4;
			View::drawCharacter(posx, posy);
			cntMap += 1;
			Play::nextStage();
		};
		if ((IsGoal::allGoal(mapData[1][1], mapData[18][8], mapData[2][17],
			mapData[13][14]) == true) && cntMap == 4) {
			MapUpdate::update5();
			View::drawMap();
			posx = 2, posy = 4;
			View::drawCharacter(posx, posy);
			cntMap += 1;
			Play::nextStage();
		};
		if ((IsGoal::allGoal(mapData[1][1], mapData[18][8], mapData[2][17],
			mapData[13][14]) == true) && cntMap == 5) {
			Inform::clearGame();
		}
	}

}
```

-----------------View.h-----------------

```
#pragma once
#include <iostream>
using namespace std;
#include <conio.h>
#include "keycode.h"
#include "Consola.h"
#include "CrushCheck.h"
#include "Goal.h"
#include "Maps.h"
#define MAP_WIDTH 20
#define MAP_HEIGHT 20
class View {

public:
	static int getkey() {//키 입력
```

```cpp
		int ch = _getch();
		return (ch == 0xe0) ? (0xe000 | _getch()) : ch;
}
static void xyputstr(int x, int y, const char* str) {
		gotoxy(x * 2, y);//정사각형 좌표계
		cout << str;
}
static void drawBox(int x1, int y1, int x2, int y2, int color) {
		textcolor(color);
		for (int i = x1; i < x2; ++i) {
				xyputstr(i, y1, "-------");
				xyputstr(i, y2, "-------");
		}
		for (int i = y1; i < y2; ++i) {
				xyputstr(x1, i, "|");
				xyputstr(x2, i, "|");
		}
		xyputstr(x1, y1, " ┌");
		xyputstr(x2, y1, "┐ ");
		xyputstr(x1, y2, " └");
		xyputstr(x2, y2, "┘ ");
}
static void drawCharacter(int col, int row) {
		gotoxy((MAP_X1 + col + 1) * 2, MAP_Y1 + row + 1);
		puts("Ω");
}
static void drawCell(int col, int row) {
		const char* cellSymbol[] = { "  ","==","o","* ","Π","<<",">>","@ "};
		int cell = mapData[row][col];
		textbackground((cell == 1) ? BROWN : YELLOW);
		textcolor((cell == 1) ? MAGENTA : CYAN);
		//textcolor(CYAN);
		gotoxy((MAP_X1 + col + 1) * 2, MAP_Y1 + row + 1);
		puts(cellSymbol[cell]);
}
static void drawMap() {
		for (int row = 0; row < MAP_HEIGHT; ++row)
				for (int col = 0; col < MAP_WIDTH; ++col)
						drawCell(col, row);
}

static void loadMoveL(int oldLX , int oldLY,int *posx,int *posy ) {
		if (oldLX==1) {//이동할 곳이 벽인 경우(게임창 안벗어나기 위해서)
				return;
		}
		//벽과 충돌하는지 검사한다.
		if (Crush::crushWallL(oldLX,oldLY,mapData) == 0) {
				return;

		}
		*posx = oldLX; *posy = oldLY;//캐릭터 이동을 위해서 포인터로 받은 값을 변경
		mapData[oldLY][oldLX] = 0;//공이 있던 자리를 path로
		mapData[oldLY][oldLX-1] = 2;//공을 왼쪽으로 이동 처리
		gotoxy((MAP_X1 + oldLX ) * 2, MAP_Y1 + oldLY + 1);//공을 그릴 위치로 이동
		puts("o");//공 그린다
		if (IsGoal::goalCheck(mapData[18][8]) == true) {//골인
				mapData[18][8] = 3;//*로 변경
				drawCell(8, 18);//해당 셀을 다시 그린다
		}
		else if (IsGoal::goalCheck(mapData[2][17]) == true) {//골
				mapData[2][17] = 3;//*로 변경
				drawCell(17, 2);//해당 셀을 다시 그린다
		}
		else if (IsGoal::goalCheck(mapData[13][14]) == true) {//골
				mapData[13][14] = 3;//*로 변경
				drawCell(14, 13);
		}
		else if ((IsGoal::goalCheck(mapData[1][1]) == true)) {//골
				mapData[1][1] = 3;//*로 변경
				drawCell(1, 1);//해당 셀을 다시 그린다
		}

}

static void loadMoveR(int oldLX, int oldLY, int* posx, int* posy) {
		if (oldLX == 18 ) {//이동할 곳이 벽인 경우(게임창 안벗어나기 위해서)
				return;
		}
		//벽과 충돌하는지 검사한다.
		if (Crush::crushWallR(oldLX, oldLY,mapData) == 0) {
				return;
		}
		*posx = oldLX; *posy = oldLY;
		mapData[oldLY][oldLX] = 0;
		mapData[oldLY][oldLX + 1] = 2;
		gotoxy((MAP_X1 + oldLX+2) * 2, MAP_Y1 + oldLY + 1);
		puts("o");
		if (IsGoal::goalCheck(mapData[18][8]) == true) {
				mapData[18][8] = 3;
				drawCell(8, 18);
		}
		else if (IsGoal::goalCheck(mapData[2][17]) == true) {
				mapData[2][17] = 3;
				drawCell(17, 2);
		}
		else if (IsGoal::goalCheck(mapData[13][14]) == true) {
				mapData[13][14] = 3;
				drawCell(14, 13);
		}
		else if ((IsGoal::goalCheck(mapData[1][1]) == true)) {
				mapData[1][1] = 3;
				drawCell(1, 1);
		}
}

static void loadMoveUp(int oldLX, int oldLY, int* posx, int* posy) {
		if (oldLY == 1) {//이동할 곳이 벽인 경우(게임창 안벗어나기 위해서)
				return;
		}
		//벽과 충돌하는지 검사한다.
		if (Crush::crushWallUp(oldLX, oldLY,mapData) == 0) {
				return;
		}
		*posx = oldLX; *posy = oldLY;
		mapData[oldLY][oldLX] = 0;
		mapData[oldLY-1][oldLX ] = 2;
		gotoxy((MAP_X1 + oldLX+1) * 2, MAP_Y1 + oldLY );
		puts("o");
		if (IsGoal::goalCheck(mapData[18][8]) == true) {
				mapData[18][8] = 3;
				drawCell(8, 18);
		}
		else if (IsGoal::goalCheck(mapData[2][17]) == true) {
				mapData[2][17] = 3;
				drawCell(17, 2);
		}
		else if (IsGoal::goalCheck(mapData[13][14]) == true) {
				mapData[13][14] = 3;
				drawCell(14, 13);
		}
		else if ((IsGoal::goalCheck(mapData[1][1]) == true)) {
				mapData[1][1] = 3;
				drawCell(1, 1);
		}
}

static void loadMoveDown(int oldLX, int oldLY, int* posx, int* posy) {
		if (oldLY == 18) {//이동할 곳이 벽인 경우(게임창 안벗어나기 위해서)
```

```
				return;
			}
			//벽과 충돌하는지 검사한다.
			if (Crush::crushWallDown(oldLX, oldLY,mapData) == 0) {
				return;
			}
			*posx = oldLX; *posy = oldLY;
			mapData[oldLY][oldLX] = 0;
			mapData[oldLY+1][oldLX ] = 2;
			gotoxy((MAP_X1 + oldLX + 1) * 2, MAP_Y1 + oldLY+2 );
			puts("o");
			if (IsGoal::goalCheck(mapData[18][8]) == true) {
				mapData[18][8] = 3;
				drawCell(8, 18);
			}
			else if (IsGoal::goalCheck(mapData[2][17]) == true) {
				mapData[2][17] = 3;
				drawCell(17, 2);
			}
			else if (IsGoal::goalCheck(mapData[13][14]) == true) {
				mapData[13][14] = 3;
				drawCell(14, 13);
			}
			else if ((IsGoal::goalCheck(mapData[1][1]) == true)) {
				mapData[1][1] = 3;
				drawCell(1, 1);
			}

		}
};
```

---------------CrushCheck.h---------------

```
#pragma once
#include <iostream>
using namespace std;
#include "Maps.h"
class Crush {
public:

		static int crushWallL(int x, int y,int mapData[][20]) {
			if (mapData[y][x - 1] == 1) {
				return 0;
			}
		}

		static int crushWallR(int x, int y,int mapData[][20]) {
			if (mapData[y][x + 1] == 1) {
				return 0;
			}
		}

		static int crushWallUp(int x, int y, int mapData[][20]) {
			if (mapData[y - 1][x] == 1) {
				return 0;
			}
		}

		static int crushWallDown(int x, int y, int mapData[][20]) {
			if (mapData[y + 1][x]== 1) {
				return 0;
			}
		}

		static bool crushBallDown(int x, int y) {
			if (mapData[x + 1][y] == 2) {
				return true;//공끼리 충돌
			}
			//단방향 충돌
```

```cpp
			if ((mapData[x + 1][y] == 6))return true;
			if ((mapData[x + 1][y] == 5))return true;
			else
				return false;
		}

		static bool crushBallUp(int x, int y) {
			if (mapData[x - 1][y] == 2) {
				return true;//공끼리 충돌
			}
			//단방향 충돌
			if ((mapData[x-1][y ] == 6))return true;
			if ((mapData[x-1][y] == 5))return true;
			else
				return false;
		}

		static bool crushBallR(int x, int y) {
			if (mapData[x ][y+1] == 2) {
				return true;//공끼리 충돌
			}
			//단방향 충돌
			if ((mapData[x][y + 1] == 6 ))return true;
			if ((mapData[x][y + 1] == 5))return true;
			else
				return false;
		}

		static bool crushBallL(int x, int y) {
			if (mapData[x][y - 1] == 2) {
				return true;//공끼리 충돌
			}
			//단방향 충돌
			if ((mapData[x][y - 1] == 5) )return true;
			if ((mapData[x][y - 1] == 6))return true;
			else
				return false;
		}

};
```

-----------------Sound.h-----------------

```cpp
#pragma once
#include <windows.h>
#include <mmsystem.h>
#pragma comment(lib, "winmm.lib")
#include <conio.h>
#include <thread>
class Play {
public:
		static void move() {
			PlaySound(TEXT("move.wav"), NULL, SND_FILENAME | SND_ASYNC );
			while (!_kbhit());
		}
		static void bomb() {
			int a = 0;
			PlaySound(TEXT("bomb.wav"), NULL, SND_FILENAME | SND_ASYNC);
			while (a==0) { //폭탄 소리가 다 출력되기 위해서 O(n^2)으로
				for (int i = 0; i < 100000; i++) {
					for (int j = 0; j < 10000; j++) {
						a++;
					}
				}
			};
		}
		static void nextStage() {
			int a = 0;
			PlaySound(TEXT("nextStage.wav"), NULL, SND_FILENAME | SND_ASYNC);
		while (a == 0) {
			for (int i = 0; i < 100000; i++) {
				for (int j = 0; j < 10000; j++) {
					a++;
				}
			}
		};
	}
};
```

--------------Information.h---------------

```
#pragma once
using namespace std;
#include <iostream>
#include "Consola.h"
class Inform {
public:
	static void InforInit() {//오른쪽 게임 내용을 알리는 창을 그린다.
		textbackground(WHITE);
		textcolor(GREEN);
		gotoxy(70, 5);
		cout << "___________________________________";
		gotoxy(70, 6);
		cout << "|                                 |";
		gotoxy(70, 11);
		cout << "|\t                          |";
		gotoxy(70, 12);
		cout << "|\t캐릭터: Ω                 |";
		gotoxy(70, 13);
		cout << "|\t                          |";
		gotoxy(70, 14);
		cout << "|\t목표물: Π                 |";
		gotoxy(70, 15);
		cout << "|\t                          |";
		gotoxy(70, 16);
		cout << "|\t폭탄: @                    |";
		gotoxy(70, 17);
		cout << "|\t                          |";
		gotoxy(70, 18);
		cout << "|\t오른쪽 이동 가능: >>       |";
		gotoxy(70, 19);
		cout << "|\t                          |";
		gotoxy(70, 20);
		cout << "|\t왼쪽 이동 가능: <<         |";
		gotoxy(70, 21);
		cout << "|\t                          |";
		gotoxy(70, 22);
		cout << "|\t이동 완료시 :  *           |";
		gotoxy(70, 23);
		cout << "|\t                          |";
		gotoxy(70, 24);
		cout << "|  이동가능횟수는 1234번 입니다. |";
		gotoxy(70, 25);
		cout << "|\t                          |";
		gotoxy(70, 26);
		cout << "|_________________________________|";
	}

	static void move(int *x,int y) {//이동 횟수 처리
		textbackground(WHITE);
		textcolor(GREEN);
		gotoxy(70, 7);
		cout << "|\t                          |";
		gotoxy(70, 8);
		if (*x < 10) {
			cout << "|\t나의 이동 횟수: " << *x << "번              |";
			(*x) += 1;//포인터사용
		}
```

```
        else if (*x < 100) {
            cout << "|\t나의 이동 횟수: " << *x << "번          |";
            (*x) += 1;//포인터사용
        }
        else if (*x <1000) {
            cout << "|\t나의 이동 횟수:" << *x << "번          |";
            (*x) += 1;//포인터사용
        }
        else if (*x < 1234) {
            textcolor(RED);
            cout << "|\t나의 이동 횟수:" << *x << "번         |";
            (*x) += 1;//포인터사용
        }
        else {//이동횟수 초과하여 게임 종료를 처리한다.
            clrscr();
            textcolor(BLUE);
            gotoxy(25, 10);
            cout << "이동횟수를 다 소비하여 종료되었습니다.";
            gotoxy(25, 12);
            cout << y - 1 << "맵까지 성공하였으며 " << y << "맵에서 실패하였습니다.";
            gotoxy(25, 14);
            cout << "오른쪽 상단 x버튼을 눌러서 나가실 수 있습니다.";
        }
    }

    static void mapNum(int x) {//맵 번호를 알려주는 함수
        textbackground(WHITE);
        textcolor(GREEN);
        gotoxy(70, 9);
        cout << "|\t                              |";
        gotoxy(70, 10);
        cout << "|\t맵 번호: " << x << "/5번                  |";
    }
    static void goalDestroyed(int a) {//목표물 파괴시 처리 함수
        clrscr();
        textcolor(BLUE);
        gotoxy(25, 10);
        cout << "목표물을 파괴하여 종료되었습니다.";
        gotoxy(25, 12);
        cout << a - 1 << "맵까지 성공하였으며 " << a << "맵에서 실패하였습니다.";
        gotoxy(25, 14);
        cout << "종료를 위하여 아무키를 입력해주시길 바랍니다..\n\n\n\n\n\n\n";
        exit(1);
    }
    static void clearGame() {//게임 클리어 처리 함수
        clrscr();
        textcolor(BLUE);
        gotoxy(25, 10);
        cout << "게임을 클리어하셨습니다!!!!!";
        gotoxy(25, 14);
        cout << "오른쪽 상단 x버튼을 눌러서 나가실 수 있습니다.";
    }
};
```

-----------------Maps.h------------------

```
#pragma once
int mapData[20][20] = {
    {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
    {1,4,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1,1,1},
    {1,1,1,1,1,1,0,6,0,0,0,0,0,0,0,0,1,4,1,1},
    {1,0,0,1,0,5,0,0,0,0,0,1,0,1,0,0,1,0,1,1},
    {1,0,0,1,0,1,0,0,0,0,0,1,0,2,0,0,0,0,0,1},
    {1,0,0,1,0,7,0,0,0,0,0,0,0,1,0,0,0,0,0,1},
    {1,0,0,1,1,1,0,0,2,0,1,0,0,1,0,0,0,0,1,1},
    {1,0,7,1,0,0,0,0,0,1,1,0,0,1,1,1,1,0,1,1},
    {1,0,0,1,1,1,1,0,0,0,1,1,1,1,0,7,1,0,1,1},
    {1,0,0,5,0,0,0,0,1,1,1,0,0,1,1,1,1,0,1,1},
    {1,0,0,1,0,0,0,0,1,0,1,7,7,0,0,0,0,0,0,1},
		{1,0,0,1,1,1,0,1,1,0,0,0,0,0,2,0,0,0,0,1},
		{1,0,0,6,5,0,0,1,0,0,0,0,0,1,0,1,1,0,0,1},
		{1,0,0,1,0,0,0,1,0,0,1,0,0,1,4,1,0,0,0,1},
		{1,0,1,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,1},
		{1,1,1,1,1,1,1,0,7,0,0,0,0,0,0,0,0,7,0,1},
		{1,0,0,0,6,0,1,1,1,1,1,0,7,0,1,1,0,0,0,1},
		{1,1,1,0,1,0,1,1,1,1,1,1,0,0,1,1,1,0,1,1},
		{1,0,1,0,6,2,0,0,4,1,0,0,0,0,1,0,0,1,7,1},
		{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
};
int mapData_2[20][20] = {
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
{1,4,0,2,0,6,0,0,0,1,0,0,0,0,0,0,1,1,1,1},
{1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,1,4,1,1},
{1,1,1,1,0,0,0,1,0,0,0,0,0,0,1,0,1,0,1,1},
{1,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1},
{1,0,0,0,0,1,1,1,1,1,0,0,1,1,1,1,1,0,0,1},
{1,0,0,7,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1},
{1,0,7,0,1,0,0,1,0,0,0,0,0,0,1,7,0,0,1,1},
{1,0,0,0,1,0,0,0,0,0,0,0,0,0,2,1,0,1,0,1},
{1,0,1,1,1,1,1,0,0,0,1,0,0,0,0,1,0,0,0,1},
{1,0,0,0,1,0,0,0,0,0,1,0,0,0,0,1,1,1,0,1},
{1,0,0,0,1,0,0,0,1,1,1,1,1,1,0,1,0,0,0,1},
{1,0,0,0,0,0,1,0,0,0,1,0,0,5,0,6,2,2,0,1},
{1,0,0,0,0,0,1,0,0,0,1,0,0,1,4,1,0,0,0,1},
{1,0,0,0,1,1,1,0,0,0,0,0,0,1,1,1,0,7,0,1},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7,0,0,1},
{1,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,1},
{1,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1},
{1,0,0,0,1,1,0,0,4,1,0,6,0,0,0,0,0,7,0,1},
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
};
int mapData_3[20][20] = {
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
{1,4,1,1,1,0,0,0,0,1,0,0,0,0,1,1,1,1,1,1},
{1,1,0,0,0,0,7,0,7,0,0,0,0,0,1,7,1,4,1,1},
{1,0,0,0,0,0,0,7,0,0,0,0,0,0,1,1,1,1,1,1},
{1,0,0,0,0,0,7,0,7,0,0,0,0,0,1,1,1,1,1,1},
{1,0,1,1,0,0,0,7,0,0,0,0,0,0,0,0,0,1,0,1},
{1,0,1,1,0,0,7,0,7,0,1,0,0,0,0,0,0,0,0,1},
{1,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,0,1,1,0,0,2,0,2,0,0,0,0,0,0,0,0,0,0,1},
{1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,1},
{1,0,1,1,0,0,2,0,2,0,0,0,0,1,4,1,0,0,0,1},
{1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,1},
{1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,1},
{1,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,1},
{1,0,0,0,0,0,0,0,4,1,1,1,1,0,0,0,0,0,0,1},
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
};
int mapData_4[20][20] = {
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
{1,4,1,1,0,0,0,0,0,1,0,0,0,0,0,0,1,1,1,1},
{1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,4,1,1},
{1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1,0,1,1},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1},
{1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1},
{1,1,1,1,1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,1},
{1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1},
{1,0,0,2,1,0,1,0,0,0,1,0,0,0,1,1,1,0,0,1},
{1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,0,0,0,2,0,1,0,0,0,0,0,0,0,0,1},
{1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},
{1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
{1,1,1,1,0,6,6,6,6,6,6,6,6,2,4,1,1,1,1,1},
{1,0,1,0,1,1,1,1,1,1,1,1,1,7,1,1,1,1,1,1},
{1,1,0,1,1,1,7,7,7,7,7,7,1,1,0,0,0,7,1,1},
{1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1},
{1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1},
{1,0,1,0,1,1,1,1,4,2,5,5,5,5,5,5,5,5,0,1},
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
};
int mapData_5[20][20] = {
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
{1,4,0,0,0,0,0,0,0,1,0,6,5,0,0,0,1,1,1,1},
{1,1,1,1,1,1,0,7,0,0,7,1,1,0,0,0,0,4,1,1},
{1,0,0,0,1,1,0,7,0,0,7,1,1,0,0,0,1,1,1,1},
{1,0,0,7,1,1,0,7,0,0,7,1,1,0,0,0,1,1,1,1},
{1,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,1},
{1,1,1,1,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,1,0,2,2,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,1,1,2,2,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,0,0,0,0,0,1,0,0,1,1,1,0,0,0,1},
{1,1,1,1,0,0,0,0,0,0,0,0,0,1,4,0,0,0,0,1},
{1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,1},
{1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},
{1,1,1,1,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,1},
{1,1,1,1,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,1},
{1,1,1,1,0,0,0,0,4,1,0,0,0,0,0,0,0,0,0,1},
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
};
class MapUpdate {

public:
    static void update2() {
            for (int i = 0; i < 20; i++) {
                for (int j = 0; j < 20; j++) {
                    mapData[i][j] = mapData_2[i][j];
                }
            }
    }
    static void update3() {
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                mapData[i][j] = mapData_3[i][j];


            }
        }
    }
    static void update4() {
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                mapData[i][j] = mapData_4[i][j];
            }
        }
    }
    static void update5() {
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                mapData[i][j] = mapData_5[i][j];
            }
        }
    }
};

class Boom {
public:
    static void delWallL(int x, int y) {
        mapData[y][x] = 0;
        mapData[y][x + 1] = 0;
```

```
	}
	static void delWallR(int x, int y) {
		mapData[y][x] = 0;
		mapData[y][x - 1] = 0;
	}
	static void delWallUp(int x, int y) {
		mapData[y + 1][x] = 0;
		mapData[y][x] = 0;
	}
	static void delWallDown(int x, int y) {
		mapData[y - 1][x] = 0;
		mapData[y][x] = 0;
	}
};
```

--------------Goal.h---------------

```
#pragma once
using namespace std;
class IsGoal {
public:

	static bool goalCheck(int x ) {
		if (x == 2) {
			return true;
		}
		else
			return false;
	}

	static bool allGoal(int a, int b, int c, int d) {
		if (a == 3 && b == 3 && c == 3 && d == 3) {
			return true;
		}
		else
			return false;

	}
};
```

--------------keycode.h---------------

```
#pragma once
#define M_UPKEY 0xe048
#define M_DOWNKEY  0xe050
#define M_RIGHTKEY 0xe04d
#define M_LEFTKEY 0xe04b
```