**C++ 팀프로젝트 #3**

-테트리스 스토리 구성하기 보고서

201810754 조윤직

201812617 송재원

201911175 박진영

201911192 양석준

제출일: 2019. 12.07

문제정의 :

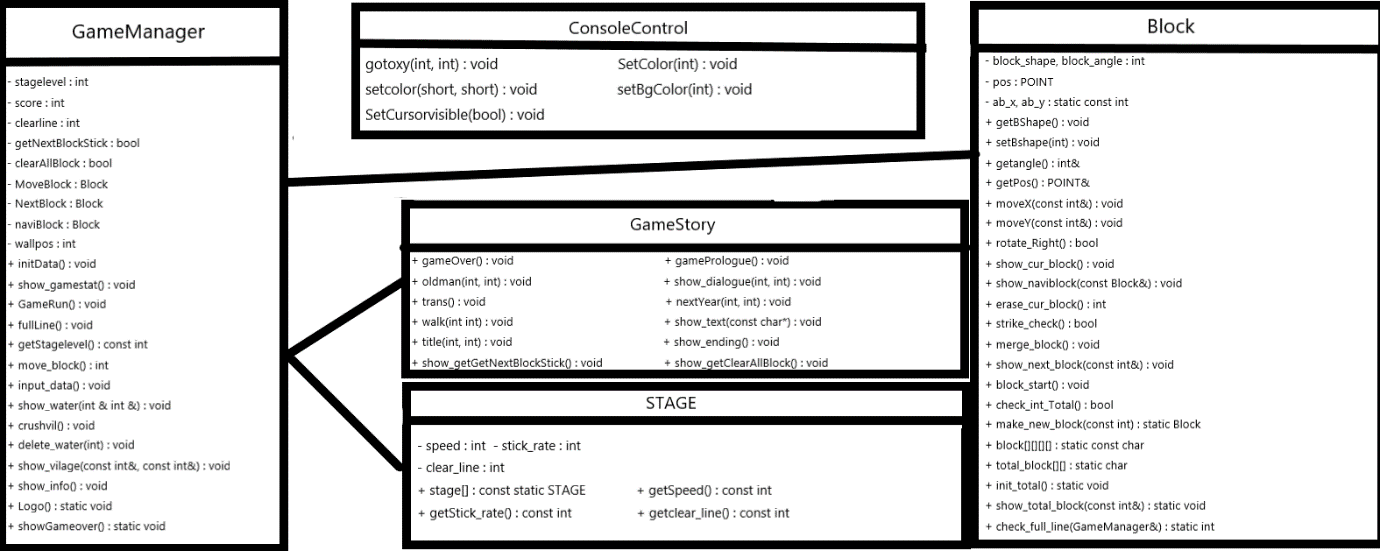
테트리스 게임에 우리들 만의 창의적 스토리를 만들어 게임을 변형

문제 분석 및 해결책 기술 :

먼저 우리가 아는 고전적인 테트리스 게임은 모두다 아는 “블럭을 쌓는” 게임입니다. 그런 게임에 스토리를 넣을 때 고려한 것이 기존의 테트리스 게임과 어느정도 연관이 있어야 하고, 스토리가 흥미로워야 하며, 사용자가 게임을 하는데 있어 스토리에 몰입하여 게임을 즐겁게 할 수 있어야 한다는 것입니다. 저희는 블록을 쌓는다는 것에서 벽을 쌓아 홍수로부터 마을을 지키는 그런 스토리를 테트리스에 넣어 게임을 만들었습니다.

먼저 저희 테트리스의 스토리는 어느 한 사람이 greenjoa 마을로 이사를 오면서 시작합니다. 그 사람은 주인공으로, 인사를 드릴 겸 마을의 촌장님에게 찾아가게 됩니다. 마을의 소개를 받은 후, 주인공은 촌장님에게 이 마을에는 매년 극심한 홍수가 찾아온다는 사실을 알게 됩니다. 주인공은 당황하지만, 촌장님에게 둑을 쌓는 방법을 배우고 마을을 홍수로부터 구하려고 노력합니다.

주인공은 촌장님에게 1년을 클리어한 직후, 블럭을 초기화하는 스킬을 전수받고, 8년을 클리어한 직후, 다음 블럭을 긴 블럭으로 바꾸는 스킬을 전수받게 됩니다. 이렇게 매해 홍수를 잘 막아내는 것 같았으나, 10년 째에는 전례 없는 극심한 홍수가 찾아와서 결국 greenjoa마을은 침수가 되어 버렸습니다. 그리고 결국 인간은 자연을 거스를 수 없다는 메시지가 나오면서 게임은 끝이 납니다.



소스코드 :

1**. ConsoleControl 클래스**

(클래스로 선언하지는 않았지만 콘솔창을 컨트롤하는 함수끼리 묶어 두었다.) -콘솔 창 컨트롤 함수 집합

* 1. Gotoxy(int x, int y) 콘솔창의 커서를 x,y위치로 이동해주는 함수:
  2. SetColor(int n): 콘솔창의 출력 색을 지정해주는 함수
  3. Setcolor(unsigned short,unsigned short) 콘솔창 출력, 배경 글 색 지정
  4. SetCursorvisible(bool visible) :(추가적으로 지정한 함수) :콘솔창의 커서를 보이게/안보이게 해주는 함수

#include "ConsoleControl.h"

void gotoxy(int x, int y)

{

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE); //윈도우 핸들 -콘솔창의 핸들을 변수로 설정

COORD pos; //long 변수x,y 두개를 가지고있는 윈도우api내 구조체 COORD

pos.Y = y;

pos.X = x;

SetConsoleCursorPosition(hConsole, pos); //콘솔 핸들의 커서를 pos위치로 설정

return;

}

void SetColor(int color)

{

HANDLE std\_output\_handle = GetStdHandle(STD\_OUTPUT\_HANDLE); //윈도우 핸들 -콘솔창의 핸들을 변수로 설정

SetConsoleTextAttribute(std\_output\_handle, color); //콘솔핸들의 텍스트 색 설정

}

void setcolor(unsigned short text, unsigned short back)

{

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), text | (back << 4));

}

void setBgColor(int bgcolor) {

CONSOLE\_SCREEN\_BUFFER\_INFO info;

GetConsoleScreenBufferInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &info);

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), ((bgcolor & 0xf) << 4) | (info.wAttributes & 0xf));

}

void SetCursorvisible(bool visible)

{

CONSOLE\_CURSOR\_INFO cursorInfo = { 0, };

cursorInfo.dwSize = 1;

cursorInfo.bVisible = visible;

SetConsoleCursorInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &cursorInfo);

}

1. **Stage 클래스**

A. 설계

|  |  |
| --- | --- |
| #pragma once  Class STAGE // 스테이지 1~10의 정보(속도, 클리어 할 라인 수, 스틱 나올확률)저장 클래스  {  private:      int speed; //스테이지의 블록떨어지는 속도 담을 변수      int stick\_rate; //스테이지의 블록이 스틱이 될 확률을 담을 변수      int clear\_line; //스테이지 완료 조건을 담을 변수      STAGE(int speed, int stick\_rate, int clear\_line) :speed(speed), stick\_rate(stick\_rate), clear\_line(clear\_line) {        }//생성자를 private화해서 외부에서 stage객체를 생성하지 못하게함      ~STAGE() {}  public:      //상수로 선언 시켜둔 stage의 정보 10개      const static STAGE stage[10];        int getSpeed()const { return speed; }      int getStick\_rate()const { return stick\_rate; }      int getclear\_line()const { return clear\_line; }    };  const STAGE STAGE::stage[10] = { STAGE(40,20,20),STAGE(38,18,20),STAGE(35,18,20),STAGE(30,17,20),STAGE(25,16,20),                  STAGE(20,15,20),STAGE(15,14,20),STAGE(10,13,20),STAGE(6,12,20) ,STAGE(4,11,99999) }; | [cs](http://colorscripter.com/info#e) |

3. **Block 클래스**

A. 설계:

|  |  |
| --- | --- |
| #pragma once  class GameManager;  #include "mainheader.h"    #ifndef DEFBLOCK  #define DEFBLOCK 0  class Block  {  private:      //블록 모양,각도 저장 변수      int block\_shape, block\_angle;      POINT pos; //블록 위치 저장 변수        //블록 절대 위치 - 이곳을 기준으로 블록위치 지정      static const int ab\_x = 5, ab\_y = 1;    public:      Block(); //블록 생성자      Block(int shape, int angle = 0, POINT pos = { 0, 0 });      ~Block();      //멤버 변수 겟 함수      int getBShape();      void setBshape(int shape);      int& getangle();      POINT& getPos();      //x축방향으로 x 만큼  이동해주는 함수      void moveX(const int& x);      //y축 방향으로 y만큼 이동해주는 함수      void moveY(const int& y);      //회전할수 있을 때에만 angle+1 해주는 함수      bool rotate\_Right();          //블록 표시 모양과,각도 ,x,y를 받아서  블록 출력      void show\_cur\_block();      // 네비블록 표시      void show\_naviblock(const Block& origin);      //블록의 모양 각도 위치를 받아 출력 되었었던 블록을 빈 칸으로 그대로 덮어 씌워 지워줌      int erase\_cur\_block();      //충돌 체크 함수 -매개 x      bool strike\_check();      //자기자신블록과 토탈 블록을 토탈블록에 병합한다. 매개x      void merge\_block();      //다음 블록 출력함수-다음블록 칸에 출력하기      void show\_next\_block(const int& level);      //블록 이동 시작전 위치 초기화 함수      void block\_start();      //지금 블록이 게임 화면상에 있는지 확인해주는 함수      bool check\_in\_Total();        //새로운 블록 생성 새로운 블록의 모양 설정 ,- 레벨에따른 스틱 확률에 따른..      static Block make\_new\_block(const int stick\_rate);        //shape,각도별 블록모양-      static const char block[7][4][4][4];        //전체 블록      static char total\_block[BOTTOM+1][RIGHTSIDE+1];      //전체 블록 초기화      static void init\_total();      //전체 블록 출력      static void show\_total\_block(const int& level);      //토탈블록의 줄 전체가 가득 찬 부분이 있으면- 제거하고 토탈 블록 데이터 수정      static int check\_full\_line(GameManager &game);      };  #endif // !DEFBLOCK | [cs](http://colorscripter.com/info#e) |

B. 소스 코드 (정의 부)

|  |  |
| --- | --- |
| #include "mainheader.h"    char Block::total\_block[21][14] = { 0, };  const char Block::block[7][4][4][4] = { //[도형의 모양 0~6][도형의 각도 0~3][y][x]     //막대모양     1,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,   1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,   1,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,   1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,       //네모모양     1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,   1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,   1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,   1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,       //'ㅓ' 모양     0,1,0,0,1,1,0,0,0,1,0,0,0,0,0,0,   1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,   1,0,0,0,1,1,0,0,1,0,0,0,0,0,0,0,   0,1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,       //'ㄱ'모양     1,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,   1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,   1,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,   0,0,1,0,1,1,1,0,0,0,0,0,0,0,0,0,       //'ㄴ' 모양     1,1,0,0,1,0,0,0,1,0,0,0,0,0,0,0,   1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,   0,1,0,0,0,1,0,0,1,1,0,0,0,0,0,0,   1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,       //'Z' 모양     1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,   0,1,0,0,1,1,0,0,1,0,0,0,0,0,0,0,   1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,   0,1,0,0,1,1,0,0,1,0,0,0,0,0,0,0,       //'S' 모양     0,1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,   1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,   0,1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,   1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0    };      void Block::init\_total()  {      for (int i = TOP; i < BOTTOM; i++)//y좌표      {          for (int j = LEFTSIDE; j <= RIGHTSIDE; j++) //x좌표          {              if ((j == LEFTSIDE) || (j == RIGHTSIDE)) //x좌표가 0또는 13일때 - 왼쪽 오른쪽 벽 블록 설정              {                  total\_block[i][j] = 1; //블록              }              else {                  total\_block[i][j] = 0;   //아니면 빈블록              }          }      }        for (int j = LEFTSIDE; j <= RIGHTSIDE; j++)         //화면의 제일 밑의 줄은 1로 채운다. 맨밑줄          total\_block[BOTTOM][j] = 1;    }    void Block::show\_total\_block(const int& level)  {      SetColor(DARK\_GRAY);// 전체 블록, 색 설정 다크그레이      for (int i = TOP; i <= BOTTOM; i++) //y축      {          for (int j = LEFTSIDE; j <= RIGHTSIDE; j++)//x축          {              if (j == LEFTSIDE || j == RIGHTSIDE || i == BOTTOM)      //레벨에 따라 외벽 색이 변함              {                  SetColor((level % 6) + 1); //레벨에 따른 외벽색                }              else {                  SetColor(DARK\_GRAY); //외벽이 아니면 그냥 다그색으로              }              gotoxy((j \* 2) + ab\_x, i + ab\_y); //해당 블록              if (total\_block[i][j] == 1)//빈칸이 아니면 네모 출력,              {                  printf("■");              }              else {                  printf("  ");              }            }      }      SetColor(BLACK);// 출력하는 글자 색 초기화-      gotoxy(77, 23);//커서 위치 재설정      return;  }    int Block::check\_full\_line(GameManager& game)  {      for (int i = TOP; i < BOTTOM; i++) //전체 모든 줄 체크      {          int j;          for (j = LEFTSIDE + 1; j < RIGHTSIDE; j++) //i 줄의 모든 칸을 체크          {              if (total\_block[i][j] == 0)//중간에 빈 곳이 있으면 break                  break;          }          if (j == 13)   //한줄이 다 채워졌음- j=13 break 없이 반복이 13번 수행된          {                  //출력-동작              show\_total\_block(game.getStagelevel()); //토탈블록 출력              SetColor(BLUE); // 블루로 출력              gotoxy(1 \* 2 + ab\_x, i + ab\_y); // 이 줄 위치로 가서              for (j = 1; j < 13; j++)              {                  printf("□"); //블루색으로 한번 출력 해주고                  Sleep(10); //0.01초마다 출력              }              gotoxy(1 \* 2 + ab\_x, i + ab\_y); //다시 줄의 시작으로 가서              for (j = 1; j < 13; j++)              {                  printf("  ");// 쭉 지워줌                  Sleep(10);              }                //데이터 수정부              for (int k = i; k > 0; k--)//지금 줄 i부터 그 윗줄 반복              {                  for (j = 1; j < RIGHTSIDE; j++) //x축                      total\_block[k][j] = total\_block[k - 1][j];//바로 위에 줄의 정보를 가져옴              }                for (j = 1; j < 13; j++) //맨 위의 줄 0으로 초기화 해줌                  total\_block[0][j] = 0;              game.fullLine();            }      }      show\_total\_block(game.getStagelevel());      return 0;  }          Block::Block()  {  }    Block::Block(int shape, int angle, POINT pos) :block\_shape(shape), block\_angle(angle), pos(pos)  {  }    Block::~Block()  {  }    int Block::getBShape()  {      return block\_shape;  }    void Block::setBshape(int shape)  {      this->block\_shape = shape;  }    int& Block::getangle()  {      return block\_angle;  }        POINT& Block::getPos()  {      return pos;  }    void Block::moveX(const int& x)  {      pos.x += x;  }    void Block::moveY(const int& y)  {      pos.y += y;  }    bool Block::rotate\_Right()  {      block\_angle = (block\_angle + 1) % 4; //angle을 하나 올리고      if (strike\_check()) { //충돌하면          block\_angle = (block\_angle - 1) % 4; //복귀          return false; //성공 실패리턴이나 결국 사용하지 않음      }      else          return true; //  }        Block  Block::make\_new\_block(const int stick\_rate)  {      // i= 확률      int i;      i = rand() % 100;      if (i <= stick\_rate)      //막대기 나올확률 계산          return Block(0);                //막대기 모양으로 리턴      else          return Block((rand() % 6) + 1);    }    void Block::show\_cur\_block()  {      //정의 해둔 SetColor함수를 이용하여 모양에 따른 색 출력 구현      switch (block\_shape)      {      case 0:          SetColor(RED);          break;      case 1:          SetColor(BLUE);          break;      case 2:          SetColor(SKY\_BLUE);          break;      case 3:          SetColor(WHITE);          break;      case 4:          SetColor(YELLOW);          break;      case 5:          SetColor(VOILET);          break;      case 6:          SetColor(GREEN);//greenjoa!          break;      default:          return;      }      for (int i = 0; i < 4; i++)      {          for (int j = 0; j < 4; j++)          {              if ((j + pos.y) < 0)// 출력되는 y좌표가 0이하 콘솔 창밖을 벗어날때는 출력하지 않음                  continue;                if (block[block\_shape][block\_angle][j][i] == 1) //해당하는 도형의 j,i위치가 빈블록이 아닐 때              {                  gotoxy((i + pos.x) \* 2 + ab\_x, j + pos.y + ab\_y); //(i+x)\*2 + ab\_x,j+y+ab\_y 위치에 출력 저 사각형의 출력이 2칸 차지하므로 x는 두배                  printf("■");                }          }      }        SetColor(BLACK);// 출력하는 글자 색 초기화-      gotoxy(77, 23);//커서 위치 재설정      return;  }    //네비블록 출력 함수  void Block::show\_naviblock(const Block& origin)  {      this->erase\_cur\_block(); //이전 네비 블록을 지우고      \*this = origin;      //origin으로 받아온 현재 이동블록으로 자기자신초기화        switch (block\_shape)      {//shape에 따라서 ..색 출력      case 0:          SetColor(DARK\_RED);          break;      case 1:          SetColor(DARK\_BLUE);          break;      case 2:          SetColor(DARK\_SKY\_BLUE);          break;      case 3:          SetColor(GRAY);          break;      case 4:          SetColor(DARK\_YELLOW);          break;      case 5:          SetColor(DARK\_VOILET);          break;      case 6:          SetColor(DARK\_GREEN);          break;      }          while (strike\_check() != true) //충돌 할때까지          moveY(1); //내려감      moveY(-1);        //출력부      for (int i = 0; i < 4; i++)//y축      {          for (int j = 0; j < 4; j++) //x축          {              if ((j + pos.y) < 0) //0보다 작을때에는 출력x                  continue;                if (block[block\_shape][block\_angle][j][i] == 1)              {                  gotoxy((i + pos.x) \* 2 + ab\_x, (j + pos.y) + ab\_y);                  printf("■");                }          }      }        //출력 후 초기화      SetColor(BLACK);      gotoxy(77, 23);      return;  }    //블록 지우기 함수  int Block::erase\_cur\_block()  {        if (block\_shape <= 6 && block\_shape >= 0)          for (int i = 0; i < 4; i++) //y축          {              for (int j = 0; j < 4; j++) //x축              {                  if (block[block\_shape][block\_angle][j][i] == 1)//해당하는 블록위치가 빈 블록이 아닐 때                  {                      if (total\_block[j + pos.y][i + pos.x] == 1) break; //이동블록이 끝까지 이동한후 다음블록이 끝까지 이동할때까지 출력되지 않는 문제가 있어 토탈블록의 부분일때는 안지워주게 추가                      gotoxy((i + pos.x) \* 2 + ab\_x, j + pos.y + ab\_y);//이 위치에 빈블록으로 지움                      printf("  ");                      //break;                    }              }          }        return 0;  }    //충돌 체크  bool Block::strike\_check()  {      int block\_dat;        if (block\_shape <= 6 && block\_shape >= 0) {          for (int i = 0; i < 4; i++)//y          {              for (int j = 0; j < 4; j++)//x              {                  if ((pos.x + j) == LEFTSIDE || (pos.x + j) == RIGHTSIDE)                      block\_dat = 1;   //이 블록이 지금 토탈 블록의 외곽부 이면 -1                  else if (pos.y + i < 0) //이경우 total\_block  에는 이부분이 없기에 추가  수정--                      block\_dat = 0;                  else                      block\_dat = total\_block[pos.y + i][pos.x + j];  //이블록이 토탈블록에 빈블록인지,찬 블록인지                      if ((block\_dat != 0) && (block[block\_shape][block\_angle][i][j] == 1))                  { //좌측벽의 좌표를 빼기위함? -- 이블록이 빈블록이 아니고 도형의 이 블록 부분도 1이면 충돌 !                      return true;                  }              }          }          return 0;      }      return 1;    }    //블록 병합  void Block::merge\_block()  {      for (int i = 0; i < 4; i++)//y축      {          for (int j = 0; j < 4; j++)//x축          { //토탈 블록에 현재블록 부분을 1로 바꿔줌              total\_block[pos.y + i][pos.x + j] |= block[block\_shape][block\_angle][i][j];          }      }      return;    }    void Block::show\_next\_block(const int& level)  {        SetColor((level + 1) % 6 + 1); //다음블록이 표기되는 테두리는 레벨따라 색이 다름      for (int i = 1; i < 7; i++) //다음에 무슨 블록이 나올지 알려주는 네모난 공간을 만듦      {          gotoxy(33, i);          for (int j = 0; j < 6; j++)          {              // 첫 행과 마지막 행만 색칠된 네모칸을 쭉 출력함. 나머지는 첫 번째 열과 마지막 줄 열에만 네모 출력.              if (i == 1 || i == 6 || j == 0 || j == 5)              {                  printf("■");              }              else {                  printf("  ");              }            }      }      Block temp(block\_shape, 0, { 15, 1 });      temp.show\_cur\_block(); // 15\*2+ab\_x하면 35,  다음블록 안에다가 도형 출력  }    void Block::block\_start()  {      pos = { 5,-3 };      block\_angle = 0;  }    bool Block::check\_in\_Total()  {      if (block\_shape <= 6 && block\_shape >= 0) {          for (int i = 0; i < 4; i++)//y for문 순서상 가장 위에잇는 칸부터          {              for (int j = 0; j < 4; j++)//x              {                  if (block[block\_shape][block\_angle][i][j] == 1) { //가장위에 있는 블록                      if (pos.y + i < 0) //블록의 가장위쪽 위치가 0보다 작으면 밖에 있음!                          return false;                      else                          return true;//아니면 안에                  }              }          }      }  } | [cs](http://colorscripter.com/info#e) |

1. **GameManager 클래스**
   1. 설계

|  |  |
| --- | --- |
| class GameManager  {  private:      //현재 스테이지 레벨      int stagelevel;      //현재 점수      int score;      //현재 클리어한 라인      int clearline;      //getNextBlockStick 스킬      bool getNextBlockStick = false;      //clearAllBlock 스킬      bool clearAllBlock = true;        //현재 이동하는 블록      Block MoveBlock;      //다음 블록      Block NextBlock;      //그림자 블록      Block naviBlock;      int wallpos = 70;      //빌리지 출력 위치      POINT vilagepos;  public:      GameManager();      ~GameManager();        void initData();      //게임 점수판 출력 함수 -해당 위치에 스테이지,점수 라인 정보 출력      void show\_gamestat();      //게임 주 실행함수      void GameRun();      //한 줄이 다찼으면 , 점수,clearline, 점수창 수정      void fullLine();      // 현재 스테이지 리턴 함수      int getStagelevel() const;        //현재 이동블록을 움직여주는 함수      // 리턴 값- 0은 정상 한칸 이동, 1이 리턴 게임 오버,      //2가 리턴 내려올 수 있는 곳까지 내려왔음(게임오버x)-이 경우 지금블록 병합처리 ,다음블록 처리      int move\_block();      //게임 시작시 - 도움말표기, 스테이지 선택 함수      void input\_data();        //물 출력 함수      void show\_water(int& i,int& gameover);      void crushvil(); //마을 침수      //물 지우기      void delete\_water(int i);      //벽 그리기      void addWall(int line);      //벽 지우기      void delete\_wall(int line);      //      void show\_vilage(const int& x,const int& y);      void show\_info();      //객체에 따라 다른 특성이 없는 함수는 정적으로      static void Logo(); //게임 로고출력함수      static void showGameover();  }; | [cs](http://colorscripter.com/info#e) |

* 1. 소스코드 (정의 부)

|  |  |
| --- | --- |
| #include "mainheader.h"  #include "GameStory.h"    GameManager::GameManager()  {      initData();      vilagepos = { 35,10 };  }    GameManager::~GameManager()  {    }    //데이터 초기화 함수  void GameManager::initData()  {      //블록의 토탈 블록 초기화함수 실행      Block::init\_total();      //현재 스테이지레벨, 라인수,점수 초기화      stagelevel = 0;      clearline = 0;      score = 0;      getNextBlockStick = false; // 스킬 존재 여부      clearAllBlock = false;  }    void GameManager::show\_gamestat()  {      //처음에 한번만 출력한뒤 스테이지 스코어 라인 글씨는 다시 출력 안하려 한 부분인거 같은데    //printed\_text를 0이 아닌 수로 변하는 부분이 없어 딱히 의미가 없는듯? -일단 그냥 유지      static int printed\_text = 0; // 밑에있는 조건문의 조건에 들어가는 변수 조건문을 실행해줌        SetColor(GRAY);      if (printed\_text == 0)      { //점수 출력          gotoxy(35, 7);          printf("STAGE");            gotoxy(35, 9);          printf("SCORE");            gotoxy(35, 12);          printf("LINES");      }      gotoxy(41, 7);      printf("%d", stagelevel + 1);      gotoxy(35, 10);      printf("%10d", score);      gotoxy(35, 13);      if (STAGE::stage[stagelevel].getclear\_line() > 20) {          printf("%7s", "???");      }      else {          printf("%7d", STAGE::stage[stagelevel].getclear\_line() - clearline);      }  }    void GameManager::GameRun()  {      //게임 오버 여부 담을 변수      SetCursorvisible(false); //시작시 커서끔      int is\_gameover = 0;      initData(); //초기화 함수 수행      Logo();      GameStory story;        // 프롤로그        story.gamePrologue();        system("CLS");        while (true) //      {          show\_info();          SetCursorvisible(true);//스테이지 선택시 커서 보여줌          input\_data(); //스테이지 선택 함수          SetCursorvisible(false); //스테이지 선택후 커서 끔          Block::show\_total\_block(stagelevel); //토탈 블록 출력 - 맨처음 프레임 출력            show\_vilage(vilagepos.x, vilagepos.y); //마을 출력            //최초 블록 세팅          MoveBlock = Block::make\_new\_block(STAGE::stage[stagelevel].getStick\_rate()); //블록 생성          MoveBlock.block\_start(); //시작 블록 초기화            NextBlock = Block::make\_new\_block(STAGE::stage[stagelevel].getStick\_rate()); //다음 블록 생성            naviBlock.show\_naviblock(MoveBlock); //네비 블록보이기          MoveBlock.show\_cur\_block(); // 이동블록 출력-네비보다 먼저 하면 네비에서출력이지워짐          NextBlock.show\_next\_block(stagelevel); //다음 블록 칸 표시          show\_gamestat();            for (int i = 0; true; i++) { //무한 루프              if (\_kbhit()) { //키입력이 있으면                  char keytemp = \_getch();                  if (keytemp == 0) {                      keytemp = \_getch();                      if (keytemp == 65) {                          fullLine();                      }                  }                  if (keytemp == KEY\_ESC) {      //esc입력이 들어오면-종료                      initData(); //데이터 초기화                      system("cls"); //전체 초기화                      MoveBlock.block\_start(); //                      break;                  }                  else if (keytemp == 'd' || keytemp == 'D') { // 치트키 스킬 오픈                      clearAllBlock = true;                      getNextBlockStick = true;                  }                  else if (clearAllBlock && (keytemp == 'A' || keytemp == 'a')) { // clearAllBlock 스킬                      Block::init\_total();                      Block::show\_total\_block(stagelevel);                  }                  else if (getNextBlockStick && (keytemp == 'S' || keytemp == 's')) { //getNextBlockStick 스킬                      NextBlock = Block(0);                      NextBlock.show\_next\_block(stagelevel);                  }                  else if (keytemp == EXT\_KEY) {//확장키- 방향키입력시                      keytemp = \_getch();//확장키 구분                      switch (keytemp)                      {                        case KEY\_UP: { //업키- 회전 키                          MoveBlock.erase\_cur\_block(); //이동블록 일단 지우고                          MoveBlock.rotate\_Right(); //회전 시킬수 있을 때에만 회전해주는 함수                          naviBlock.show\_naviblock(MoveBlock); //네비 블록보이기                          MoveBlock.show\_cur\_block(); // 이동블록 출력                          break;                      }                      case KEY\_LEFT: {//좌측키 -왼쪽이동키                          if (MoveBlock.getPos().x > LEFTSIDE) { //이동블록이 가장 좌측에 있지 않으면                              MoveBlock.erase\_cur\_block(); //이동 블록 일단지우고                              MoveBlock.moveX(-1); //한칸 이동후                              if (MoveBlock.strike\_check()) { //충돌하면                                  MoveBlock.moveX(1); //다시 원 위치로                              }                              naviBlock.show\_naviblock(MoveBlock); //네비 블록보이기                              MoveBlock.show\_cur\_block(); // 이동블록 출력                          }                          break;                      }                      case KEY\_RIGHT: {//우측키 오른쪽 이동키                          if (MoveBlock.getPos().x < RIGHTSIDE) {//이동 블록이 가장 우측에 있지 않으면 -이거는 별로의미가 없는듯                              MoveBlock.erase\_cur\_block(); //이동 블록 일단 지우고                              MoveBlock.moveX(1); //한 칸 이동후                              if (MoveBlock.strike\_check()) {//충돌하면                                  MoveBlock.moveX(-1); //다시 원위치로                              }                              naviBlock.show\_naviblock(MoveBlock); //네비 블록보이기                              MoveBlock.show\_cur\_block(); // 이동블록 출력                          }                          break;                      }                      case KEY\_DOWN: { //아래키 - 내려가게하는 키                          is\_gameover = move\_block(); //블록을 한칸 아래로 이동함수                          if (is\_gameover == 0)  //0값이 출력 되었으면                              naviBlock.show\_naviblock(MoveBlock); //네비 블록보이기                            //is\_gameover ==2인 아래에 도달했을때는 어쩌피 네비와 위치가 같아 출력x                          MoveBlock.show\_cur\_block(); // 이동블록 출력                            break;                      }                      default:                          break;                      }                      continue;                  }                  if (keytemp == SPACEBAR) { //스페이스바를 누르면                      while (is\_gameover == 0) {                          is\_gameover = move\_block();// 블록이 아래에 도착할 때까지 수행                      }//이동블록 출력  이부분에서는 네비블록과 위치가 동일 하기에..                      MoveBlock.show\_cur\_block();                  }                }              show\_water(i, is\_gameover);                if (is\_gameover == 1) { //게임 오버되면                  showGameover(); //게임오버 출력                  SetColor(GRAY); //                  initData(); //데이터 초기화                  MoveBlock.block\_start();                  is\_gameover = 0;                  break;              }              if (STAGE::stage[stagelevel].getclear\_line() <= clearline) //깬 라인수가 클리어 수보다 많으면              {                  i = 0;                  delete\_water(i);                  delete\_wall(clearline);                  if (stagelevel == 7) { // 스킬 s getNextBlockStick 추가                      getNextBlockStick = true;                      system("cls");                      story.show\_getGetNextBlockStick();                      /\*show\_vilage(vilagepos.x, vilagepos.y);                      NextBlock.show\_next\_block(stagelevel);\*/                  }                  if (stagelevel == 0) {// 스킬 a clearAllBlock 추가                      clearAllBlock = true;                      system("cls");                      story.show\_getClearAllBlock();                      /\*show\_vilage(vilagepos.x, vilagepos.y);                      NextBlock.show\_next\_block(stagelevel);\*/                  }                  system("cls");                  story.nextYear(10, 10);                    show\_info();                  Block::init\_total();                  Block::show\_total\_block(++stagelevel);  //줄 초기화 -수정                  show\_vilage(vilagepos.x, vilagepos.y);                  NextBlock.show\_next\_block(stagelevel);                  show\_gamestat();              }              if ((i % STAGE::stage[stagelevel].getSpeed()) == 0) {                  if (is\_gameover != 1) {                      is\_gameover = move\_block();                      naviBlock.show\_naviblock(MoveBlock); //네비 블록보이기                      MoveBlock.show\_cur\_block(); // 이동블록 출력                  }                }              //출력부 초기화              gotoxy(77, 23);              Sleep(15);              gotoxy(77, 23);          }          //함수 초기화          initData();          show\_info();      }    }    void GameManager::fullLine()  {      clearline += 1;      score += 100 + (stagelevel \* 10) + (rand() % 10); //점수 레벨에따른 상승      show\_gamestat(); //게임 정수판 수정      show\_vilage(vilagepos.x, vilagepos.y);      addWall(clearline);  }    int GameManager::getStagelevel() const  {      return stagelevel;  }  //블록움직여 주는 함수 블록의 모양,앵글,좌표,다음모양 을 포인터변수로 매개- 0은 정상 한칸 이동, 1이 리턴 게임 오버, 2가 리턴 내려올 수 있는 곳까지 내려왔음(게임오버x)-이 경우 지금블록 병합처리 ,다음블록 처리  int GameManager::move\_block()  {      MoveBlock.erase\_cur\_block(); //이전 위치의 블록 지우기        MoveBlock.moveY(1); //아래로 한 칸 이동      if (MoveBlock.strike\_check()) {// 만약 이블록이 토탈 블록과 충돌 했으면          MoveBlock.moveY(-1);          if (MoveBlock.check\_in\_Total()) {//블록 다 떨어짐                MoveBlock.merge\_block();              Block::check\_full\_line(\*this);              Block::show\_total\_block(stagelevel);              show\_gamestat(); //게임 정수판 수정              MoveBlock = NextBlock; //블록 새롭게 바꾸기              MoveBlock.block\_start(); //              NextBlock = Block::make\_new\_block(STAGE::stage[stagelevel].getStick\_rate());              NextBlock.show\_next\_block(stagelevel);              return 2;          }          else {              return 1;          }        }      return 0;  }    void GameManager::Logo()  {      system("mode con cols=120 lines=30");      int i, j;// 반복문에서 사용되는 변수      GameStory story;      story.title(30, 0);      gotoxy(45, 28);      printf("Please Press Any Key~!");      // 키를 입력 받을때 까지 계속 반복해서 블록들 출력함      for (i = 0; i >= 0; i++)      {          if (i % 40 == 0)          {              for (j = 0; j < 5; j++)              {                  gotoxy(14, 20 + j);                  printf("                                                                                     ");                  }              for (int i = 23; i < 28; i++) {                  gotoxy(18, i); printf("%25s", ""); //수정-잔상지우기위해              }              Block(rand() % 7, rand() % 4, { 14,20 }).show\_cur\_block();              Block(rand() % 7, rand() % 4, { 22,20 }).show\_cur\_block();              Block(rand() % 7, rand() % 4, { 31,20 }).show\_cur\_block();              Block(rand() % 7, rand() % 4, { 38,20 }).show\_cur\_block();            }          if (kbhit())              break;          Sleep(30);      }        getche();      system("cls");        return;  }    void GameManager::input\_data()  {      int i = 0; // 입력받는 값을 변수로 선언      SetColor(GRAY);      gotoxy(10, 7);      printf("┏━━━━━━━ < GAME KEY > ━━━━━━┓"); //수정      Sleep(10);      gotoxy(10, 8);      printf("┃ UP   : Rotate Block       ┃");      Sleep(10);      gotoxy(10, 9);      printf("┃ DOWN : Move One-Step Down ┃");      Sleep(10);      gotoxy(10, 10);      printf("┃ SPACE: Move Bottom Down   ┃");      Sleep(10);      gotoxy(10, 11);      printf("┃ LEFT : Move Left          ┃");      Sleep(10);      gotoxy(10, 12);      printf("┃ RIGHT: Move Right         ┃");      Sleep(10);      gotoxy(10, 13);      printf("┗━━━━━━━━━━━━━━━━━━━━━━━━━━━┛"); //수정          while (i < 1 || i > 8)// 1미만 8초과인수일때 다시 입력받음      {          gotoxy(10, 3);          printf("Select Start level[1-8]:              \b\b\b\b\b\b\b\b\b\b\b\b\b\b");          scanf("%d", &i);          if (i == 1234) {              i = 10;              break;          }          while (getchar() != '\n'); //버퍼를 비움:scanf의 경우 엔터가 버퍼에 남아 값이 있다고 인식되어 무한반복되므로          }          stagelevel = i - 1;      system("cls");      return;  }      void GameManager::show\_water(int& i, int& gameover)  {      int easy = 234; // 속도값 작을수록 빠름      if (stagelevel == 9) {          easy = 55;      }      int abx = 45;      int set = i / easy;      int width = 40;      int nowY = STAGE::stage[stagelevel].getclear\_line();      if (nowY > 20) {          nowY = 21;      }        if (i % easy == 0) {          for (int i = 0; i < nowY; i++) {              gotoxy(100 - set, 21 - i);              setBgColor(BLUE);              cout << "   ";          }          set++;          setBgColor(BLACK);      }      if (set > 28) {          gameover = 1;          crushvil();      }  }    void GameManager::crushvil()  {      int nowY = STAGE::stage[stagelevel].getclear\_line();      if (nowY > 20) {          nowY = 21;      }      for (int i = clearline; i < nowY; i++) {          gotoxy(wallpos, 21 - i);          setBgColor(BLUE);          cout << "  ";      }      for (int j = 2; j < 24; j++, j++) {//x축          for (int i = 0; i < nowY; i++) {//y축              gotoxy(wallpos - j, 21 - i);              cout << "  ";          }          Sleep(200);      }      setBgColor(BLACK);  }    void GameManager::delete\_water(int i)  {      int nowY = clearline; //일단 스테이지 완료시 스테이지에서 깬 라인수= 물 높이 이니까      for (int i = 0; i < nowY; i++) {          gotoxy(72, 21 - i);          setBgColor(BLACK);          cout << "                               ";      }  }    void GameManager::addWall(int line)  {      gotoxy(wallpos, 22 - line);      SetColor(DARK\_RED);      cout << "■";      SetColor(BLACK);  }    //클리어 라인 초기화 및 벽 제거  void GameManager::delete\_wall(int line)  {      setBgColor(BLACK);      for (int i = 1 + clearline - STAGE::stage[stagelevel].getclear\_line(); i <= clearline; i++) {          gotoxy(wallpos, 22 - i);          cout << "    ";      }      clearline -= STAGE::stage[stagelevel].getclear\_line(); //라인수 초기화    }    void GameManager::show\_vilage(const int& x, const int& y)  {      show\_info();      int width = 20;      SetColor(GREEN);      for (int i = 0; i < 6; i++) {          gotoxy(x + width - i \* 2, y + i);          for (int j = 0; j < 2 \* (i + 1); j++) {              cout << "▲";          }      }      SetColor(RED);      for (int i = 6; i < 12; i++) {            gotoxy(x + width - 6, y + i);          cout << "■■■■■■■■";      }      SetColor(GRAY);      gotoxy(x + width - 2, y + 8);      cout << "greenjoa";      gotoxy(72, 40);  }        void GameManager::show\_info()  {      gotoxy(5, 22);      SetColor(WHITE);      cout << "┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┓"; //98개      for (int i = 0; i < 5; i++) {          gotoxy(5, 23 + i);          cout << "┃                                                                                                ┃";      }      gotoxy(5, 28);      cout << "┗━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┛"; //98개      gotoxy(6, 24);      cout << "← : 왼쪽으로 이동\t→ : 오른쪽으로 이동\t\t↑ : 블럭회전";      gotoxy(6, 25);      cout << "ESC : 게임 종료\t\tSpaceBar : 블럭  하강";      if (clearAllBlock)      {          gotoxy(6, 26);          cout << "다시블록! : ON";      }      if (getNextBlockStick) {          gotoxy(35, 26);          cout << "스틱! : ON";      }  }              void GameManager::showGameover()  {      //게임 오버 출력      GameStory story;      story.show\_ending();      story.gameOver();      fflush(stdin); //지금까지 쌓여있던 입력 버퍼를 다 지워줌      Sleep(1000); //1초 정지후      getche();      system("cls");        return;  } | [cs](http://colorscripter.com/info#e) |

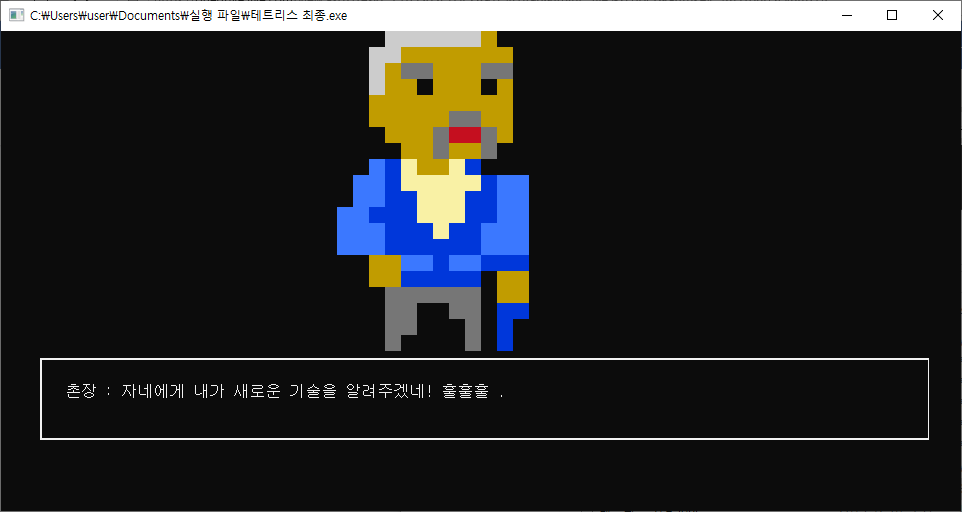
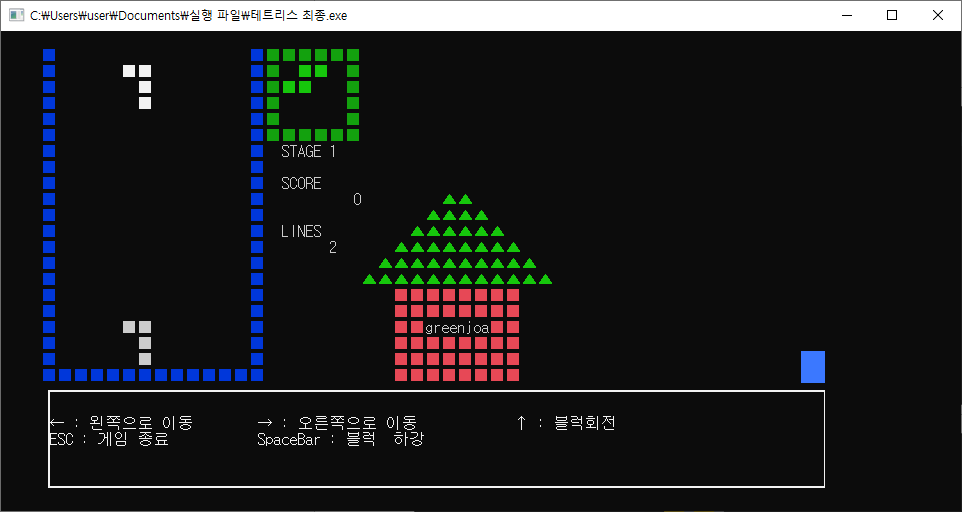
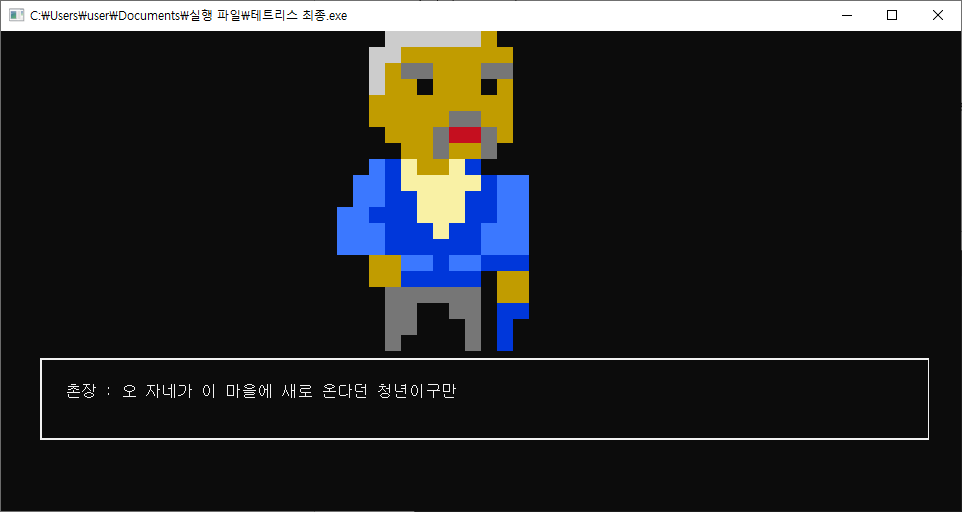
1. **GameStory 클래스**
   1. 설계

|  |  |
| --- | --- |
| class GameStory  {  public:      GameStory();      ~GameStory();      //게임 오버 출력함수      void gameOver();      // 게임 프롤로그 함수      void gamePrologue();      // 촌장님 출력 함수      void oldman(int, int);      //대화 출력      void show\_dialogue(int, int);      // 물올라왔다 내려갔다 하는 함수      void trans();      //일년 후!      void nextYear(int x, int y);      // 걷는 것 출력함수      void walk(int x, int y);      // 텍스트 띄우는 함수 출력 후, Enter누를 때까지 대기      void show\_text(const char\* str);      //타이틀 출력 함수      void title(int, int);      //엔딩 씬 출력 함수      void show\_ending();      // 기술 추가 해주는 스토리 출력      void show\_getGetNextBlockStick();      // 기술 추가 해주는 스토리 출력      void show\_getClearAllBlock();    }; | [cs](http://colorscripter.com/info#e) |

* 1. 소스 코드 (정의부)

|  |  |
| --- | --- |
| #include "GameStory.h"  #include "mainheader.h"  #include <fstream>  #include <mmsystem.h>  #include <Windows.h>    #pragma comment(lib,"winmm.lib")  GameStory::GameStory()  {  }      GameStory::~GameStory()  {  }    void GameStory::gameOver()  {      SetColor(WHITE);      FILE\* f = NULL;      fopen\_s(&f, "gameover.txt", "r");      int i, j;      int n;      int arr[169][2];      int cnt = 0;      int now = 0;      int now\_i = 0;      fscanf\_s(f, "%d", &n);      SetCursorvisible(false);      system("mode con: cols=120 lines=31");      for (i = 0; i < n; i++)      {          fscanf\_s(f, "%d %d", &arr[i][0], &arr[i][1]);      }      Sleep(50);        for (i = 1; i < 120; i++)      {          gotoxy(i, 7);          printf("─");          gotoxy(120 - i, 24);          printf("─");          Sleep(10);      }      for (i = 0; i < n; i++)      {          if (now != arr[i][0]) {              now = arr[i][0];              setcolor(0, 7);              for (j = now\_i; j <= i; j++)              {                  gotoxy(arr[j][0] \* 2 + 30, arr[j][1] + 2);                  printf("  ");              }              now\_i = i;              Sleep(50);          }          setcolor(0, 7);          gotoxy(arr[i][0] \* 2 + 30, arr[i][1] + 2);          printf("  ");          Sleep(5);      }      //Fade Out      Sleep(3000);      for (i = 0; i < n; i++)      {          if (now != arr[i][0]) {              now = arr[i][0];              Sleep(40);              setcolor(0, 0);              for (j = now\_i; j < i; j++)              {                  gotoxy(arr[j][0] \* 2 + 30, arr[j][1] + 2);                  printf("■");                  gotoxy(1, 24);              }              now\_i = i;          }          setcolor(0, 0);          gotoxy(arr[i][0] \* 2 + 30, arr[i][1] + 2);          //printf("■");      }      gotoxy(27 \* 2 + 1, 23);      printf("                                        ");      for (i = 1; i < 120; i++)      {          setcolor(0, 0);          gotoxy(i, 7);          printf("■■■■");          gotoxy(120 - i, 24);          printf("■■■■");          Sleep(10);      }      system("CLS");      setcolor(7, 0);  }  void GameStory::title(int X, int Y)  {      ifstream file;      file.open("Title.txt");      int x, y;      file >> y >> x;      char pixel[30][80];      for (int i = 0; i < x; i++) {          for (int j = 0; j < y; j++) {              file >> pixel[i][j];          }      }      setBgColor(BLACK);      int cnt = 0;      for (int i = 0; i < y; i++) {          for (int j = 0; j < 6; j++) {              gotoxy(X + i, Y + j);                if (pixel[j][i] == '1') {                  setBgColor(GREEN);                  printf(" ");              }              else {                  setBgColor(BLACK);                  printf(" ");              }              cnt++;          }          if(cnt%6 == 0)              Sleep(10);      }      Sleep(500);        for (int i = 0; i < y; i++) {          for (int j = 12; j < 18; j++) {              gotoxy(X + i, Y + j);              if (pixel[j][i] == '1') {                  setBgColor(RED);                  printf(" ");              }              else {                  setBgColor(BLACK);                  printf(" ");              }              cnt++;          }          Sleep(10);          if (cnt % 6 == 0)              Sleep(10);      }        Sleep(500);        for (int i = 0; i < y; i++) {          for (int j = 6; j < 12; j++) {              gotoxy(X + i, Y + j);              if (pixel[j][i] == '1') {                  setBgColor(YELLOW);                  printf(" ");              }              else {                  setBgColor(BLACK);                  printf(" ");              }          }      }    }  void GameStory::show\_text(const char\* str) {      printf(str);      getchar();  }  void GameStory::gamePrologue()  {      SetColor(WHITE);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("나 : 후우, 드디어 도착한건가. 피곤하다....");      gotoxy(8, 23);        show\_text("일단 짐부터 풀고 마을 구경이나 해볼까?");      system("CLS");      show\_dialogue(4, 20);      gotoxy(8, 22);        show\_text("나 : 그 전에 새로 이 지역의 식구가 된거니까 이 마을 촌장님께 인사를 드리는게 맞겠지?");      system("CLS");      show\_dialogue(4, 20);        PlaySound(TEXT("./Walk.wav"), NULL, SND\_ASYNC); //경로 수정 요망      walk(40, 5);      Sleep(3000);      system("CLS");      show\_dialogue(4, 20);        PlaySound(TEXT("./Door.wav"), NULL, SND\_ASYNC); //경로 수정 요망      Sleep(2000);      gotoxy(8, 22);      show\_text("나 : 촌장님 지금 안에 계시나요? ");      system("CLS");      oldman(40, 0);      show\_dialogue(4, 20);        gotoxy(8, 22);      show\_text("촌장 : 오 자네가 이 마을에 새로 온다던 청년이구만");      gotoxy(8, 23);      show\_text("이 늙은이가 우리 마을을 좀 소개해줘도 괜찮겠나 ? ");        system("CLS");      oldman(40, 0);      show\_dialogue(4, 20);        gotoxy(8, 22);      show\_text("나 : 아 감사합니다!");        system("CLS");      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("잠시 후");      oldman(40, 0);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("촌장 : 에휴...");      gotoxy(8, 22);      show\_text("나 : 무슨 걱정이라도 있으신가요?");      gotoxy(8, 22);      show\_text("촌장 : 사실 우리 greenjoa 마을에는 매년 엄청난 홍수가 찾아오는데, 둑을 쌓을 인력이 영 부족해서 말이야...");      system("CLS");      oldman(40, 0);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("나 : 저라도 괜찮다면 도움이 될 수 있을까요?");      gotoxy(8, 22);      show\_text("촌장 : 에구구... 이제 막 이사온 사람한테 미안해서 어쩌나?");      system("CLS");      oldman(40, 0);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("나 : 괜찮습니다. 둑은 어떻게 쌓으면 되나요?");      gotoxy(8, 23);      show\_text("촌장 : 바로 이렇게 쌓는 것이지 훌훌훌");  }    void GameStory::show\_dialogue(int x, int y)  {      gotoxy(x, y);      cout << "┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┓";        for (int i = 1; i <= 5; i++) {          gotoxy(x, y + i);          cout << "┃                                                                                                              ┃";      }        gotoxy(x, y + 5);      cout << "┗━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┛";    }  void GameStory::trans()  {      setBgColor(BLUE);      for (int i = 0; i < 30; i++) {          cout << "                                                                                                                        " << endl;          Sleep(50);      }      setBgColor(BLACK);      gotoxy(0, 0);      Sleep(1000);      for (int i = 0; i < 30; i++) {          cout << "                                                                                                                        " << endl;          Sleep(50);      }      setBgColor(BLACK);    }    void GameStory::show\_ending()  {      SetColor(RED);      gotoxy(52, 5);      printf("■");        for (int i = 1; i < 10; i++) {          gotoxy(52 - i \* 2, 5 + i);          for (int j = 0; j < i \* 2 + 1; j++) {              printf("■");          }      }      SetColor(GREEN);      for (int i = 0; i < 8; i++) {          gotoxy(40, 14 + i);          for (int j = 0; j <= 12; j++) {              printf("■");          }      }        Sleep(1000);      for (int i = 0; i <= 30; i++) {          gotoxy(0, i);          for (int j = 0; j < 120; j++) {              setBgColor(BLUE);              printf(" ");          }          Sleep(20);      }      trans();  //    getchar();      setBgColor(BLUE);      SetColor(WHITE);  }  void GameStory::show\_getGetNextBlockStick()  {      SetColor(WHITE);      system("CLS");      oldman(40, 0);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("촌장 : 자네에게 내가 새로운 기술을 알려주겠네! 훌훌훌 .");      gotoxy(8, 22);      show\_text("촌장 : 이기술은 스틱! 이라는 기술이라네, s 나 S키 를 누르면 막대기 블록을 얻을 수 있을거야..");      system("CLS");      oldman(40, 0);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("촌장 : 잘해보게나!");      system("cls");  }  void GameStory::show\_getClearAllBlock()  {      SetColor(WHITE);      oldman(40, 0);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("촌장 : 자네에게 내가 새로운 기술을 알려주겠네! 훌훌훌 .");      gotoxy(8, 22);      show\_text("촌장 : 이기술은 다시블록! 이라는 기술이라네,a 나 A키를 누르면 블록을 처음부터 쌓을수 있지...");      system("CLS");      oldman(40, 0);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("촌장 : 하지만 계속 다시 하다간 시간이 부족할 것이야...");      gotoxy(8, 22);      system("CLS");      oldman(40, 0);      show\_dialogue(4, 20);      gotoxy(8, 22);      show\_text("촌장 : 잘해보게나!");      system("cls");  }  void GameStory::nextYear(int X, int Y)  {      ifstream next;      next.open("OneYear.txt");      int x, y;      int cnt;      next >> y >> x;        char pixel[30][80];      for (int i = 0; i < x; i++) {          for (int j = 0; j < y; j++) {              next >> pixel[i][j];          }      }      for (int i = 0; i < y; i++) {          for (int j = 0; j < x; j++) {              gotoxy(X + i, Y + j);              if (pixel[j][i] == '1') {                  setBgColor(WHITE);                  printf(" ");              }              else {                  setBgColor(BLACK);                  printf(" ");              }          }      }      Sleep(1000);      cnt = 0;      for (int i = 0; i < y; i++) {          for (int j = 0; j < x; j++) {              gotoxy(X + i, Y + j);              if (pixel[j][i] == '1') {                  setBgColor(GRAY);                  printf(" ");              }              else {                  setBgColor(BLACK);                  printf(" ");              }          }          Sleep(25);      }      for (int i = 0; i < y; i++) {          for (int j = 0; j < x; j++) {              gotoxy(X + i, Y + j);              if (pixel[j][i] == '1') {                  setBgColor(BLACK);                  printf(" ");              }              else {                  setBgColor(BLACK);                  printf(" ");              }          }          Sleep(50);      }  }  void GameStory::walk(int X, int Y)  {      ifstream file;      file.open("Walk.txt");      int x, y;      file >> y >> x;        char pixel[30][80];      for (int i = 0; i < x; i++) {          for (int j = 0; j < y; j++) {              file >> pixel[i][j];          }      }      for (int i = 0; i < x; i++) {          gotoxy(X, Y + i);          for (int j = 0; j < y; j++) {              switch (pixel[i][j]) {              case '0':                  setBgColor(BLACK);                  break;              case '1':                  setBgColor(DARK\_RED);                  break;              case '2':                  setBgColor(BLUE);                  break;              case '3':                  setBgColor(DARK\_YELLOW);                  break;              case '4':                  setBgColor(BLACK);                  break;              case '5':                  setBgColor(DARK\_RED);                  break;              case '6':                  setBgColor(BLUE);                  break;              case '7':                  setBgColor(YELLOW);                  break;              case '8':                  setBgColor(DARK\_BLUE);                  break;              }              cout << "  ";          }          cout << endl;          setBgColor(BLACK);      }  }    void GameStory::oldman(int X, int Y)  {      ifstream oldman;      oldman.open("Oldman.txt");      int x, y;      oldman >> y >> x;        char pixel[30][80];      for (int i = 0; i < x; i++) {          for (int j = 0; j < y; j++) {              oldman >> pixel[i][j];          }      }      for (int i = 0; i < x; i++) {          gotoxy(X, Y + i);          for (int j = 0; j < y; j++) {              switch (pixel[i][j]) {              case '0':                  setBgColor(BLACK);                  break;              case '1':                  setBgColor(GRAY);                  break;              case '2':                  setBgColor(DARK\_YELLOW);                  break;              case '3':                  setBgColor(DARK\_GRAY);                  break;              case '4':                  setBgColor(BLACK);                  break;              case '5':                  setBgColor(DARK\_RED);                  break;              case '6':                  setBgColor(BLUE);                  break;              case '7':                  setBgColor(YELLOW);                  break;              case '8':                  setBgColor(DARK\_BLUE);                  break;              }              cout << "  ";          }          cout << endl;          setBgColor(BLACK);      }  } | [cs](http://colorscripter.com/info#e) |

실행결과 :



느낀 점 및 토의사항 :

1. 스토리를 구성하는 것이 쉽지 않았지만, 팀원과 머리를 맞대고 토의를 하다 보니 상당히 재미있으면서도 교훈을 주는 스토리를 구성할 수 있었음.
2. 발표까지 서로 협업하여 영상과 ppt, 프로그램을 짜는 과정이 큰 의미가 있었다고 생각함. 다만 과정에서 쓴 노력에 비해 발표가 부실했던 점은 큰 아쉬움으로 남음. 그러나 이를 계기로 오히려 상호간의 협응성과 협조가 매우 중요하며, 특히 역할분담을 확실히 해야함을 깨닫게 됨.
3. 저희 조 모두 프로젝트를 시작하고 매 수업시간 종료 후 항상 팀원과 최소 10분에서 길게는 1시간까지 토의, 대화를 나누었던 것 같습니다. 팀원과 이야기한 부분은 정말 많았지만, 구현으로 까지 이어진 부분은 많지 않았습니다. 지금 돌이켜 생각해보면 이야기를 시작하면 항상 처음부터 이야기 하게 되었습니다. 그 이유로는 구현의 전체적인 틀을 제대로 이야기 하지 못한 것, 이야기 한 내용들 중 정리한 것을 적어 두어 계속 상기 할 수 있게 하지 않은 것 등으로 생각되어집니다. 그래서 항상 이전에 이야기 했던 부분 인 것 같으면서도 계속 해결 방안을 찾게 되었고, 더 나아가 팀원들과 의견을 나누는 부분 까지도 문제가 되었던 것 같습니다.   
     
   그런 미흡한 부분이 있었지만 팀원들과 자주 이야기하며 구현에 대해 대화를 나누었던 것이 도움이 되었습니다. 저 혼자라면 막연하게 이렇게 해야 지 하며 제대로 정리가 안된 상태로 진행하던 것들이 팀원과 대화를 하면서 좀 더 명확하게 정리가 된 상태 혹은 더 좋은 방식으로 구현을 하게 되어 더 편하게 구현할 수 있었고. 또한 의사 소통을 하며 이런 저런 서로 이야기를 나눈 부분이 생각하지 못했던 것을 포함해 많은 좋은 아이디어를 나눌 수 있어 정말 좋았습니다.  
     
   결과적으로 협업능력에 대해 많은 것을 생각하게 되는 프로젝트였던 것 같습니다. 팀원들과 대화하는 과정, 결정한 내용에 따라 구현하는 과정 모두 정말 재밌게 잘했습니다.

개인 활동 내역

1. 양석준 : Block 클래스 구현, Stage클래스 구현, GameStory 클래스 구현, 게임 스토리 구성, 도트와 사운드 및 각종 게임의 그래픽 작업, 게임을 소개하는 영상 제작.

2. 박진영 : Block 클래스 구현, Stage클래스 구현, GameStory 클래스 구현, 게임 스토리 구성, 게임의 시연을 위한 STAGE 난이도 조절 및 테스터 역할, 게임 내에 있는 버그 찾기, 게임의 전반적인 구조도 작성

3. 조윤직 : GameManager 클래스 구현 ,코드 전반적인 검토와 수정

4. 송재원 : GameManager 클래스 구현 ,코드 전반적인 검토와 수정, PPT작성

공동으로 나누어 작업했던 부분 :

전체함수에 대한 분석, 버그 수정 ,보고서 작성,

조원 모두 불참여나 덜했다고 생각하지 않고 조원 모두 적절히 자신이 맡은 부분 잘 수행하였다고 생각.