

1.계획

- C언어에서 소화 할 수 있는 게임인 네모네모로직 퍼즐 구현
- 1.커스텀 모드 : 사용자의 입력을 받아 문제를 만들고 풀어 볼 수있는 모드
- 2.어드벤처 모드 : 난이도 별 문제를 사용자가 풀어 볼 수있는 모드.

2.요구분석

타이틀 화면

1. 노래 재생
2. 사용자가 원하는 모드와 퍼즐칸 선택

커스텀 모드

1. 사용자가 문제를 만들 수 있어야함

모드 공통점

1. 사용자가 키보드를 통해 퍼즐 풀이
2. 게임 도중 타이틀화면으로 이동 가능
3. 정답 제출을 통해 정답인지 확인 가능

외부 요구사항

1. 게임을 그만 둘 수 있음

내부 요구사항

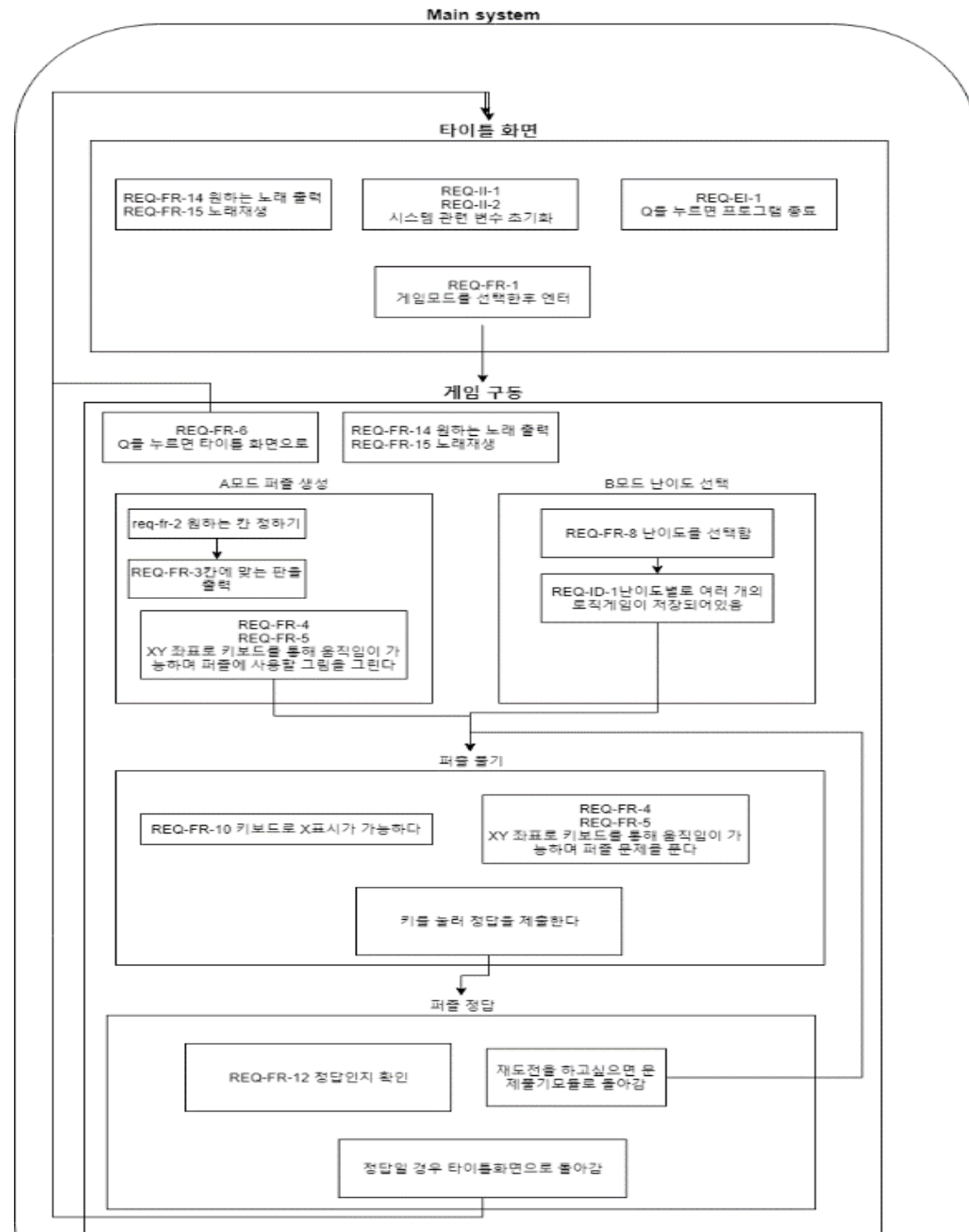
1. 난이도에 따른 로직문제 저장
2. 사용자가 만든 로직문제를 저장

3.설계

- 상위 설계(High Level Design)
- 하위 설계(Low Level Design)

3.1 상위 설계

함수는 크게
프로그램을 실행하면 난이도와 모드를
선택하는 타이틀함수와
사용자정의 모드를하는 A모드함수,
난이도에따른 로직을 푸는 B모드함수
로 이루어져있음.



3.2 하위 설계(순서도)

1. 게임시작

2. 게임모드 선택

2-1. 커스텀 모드

2-2 .어드벤처 모드

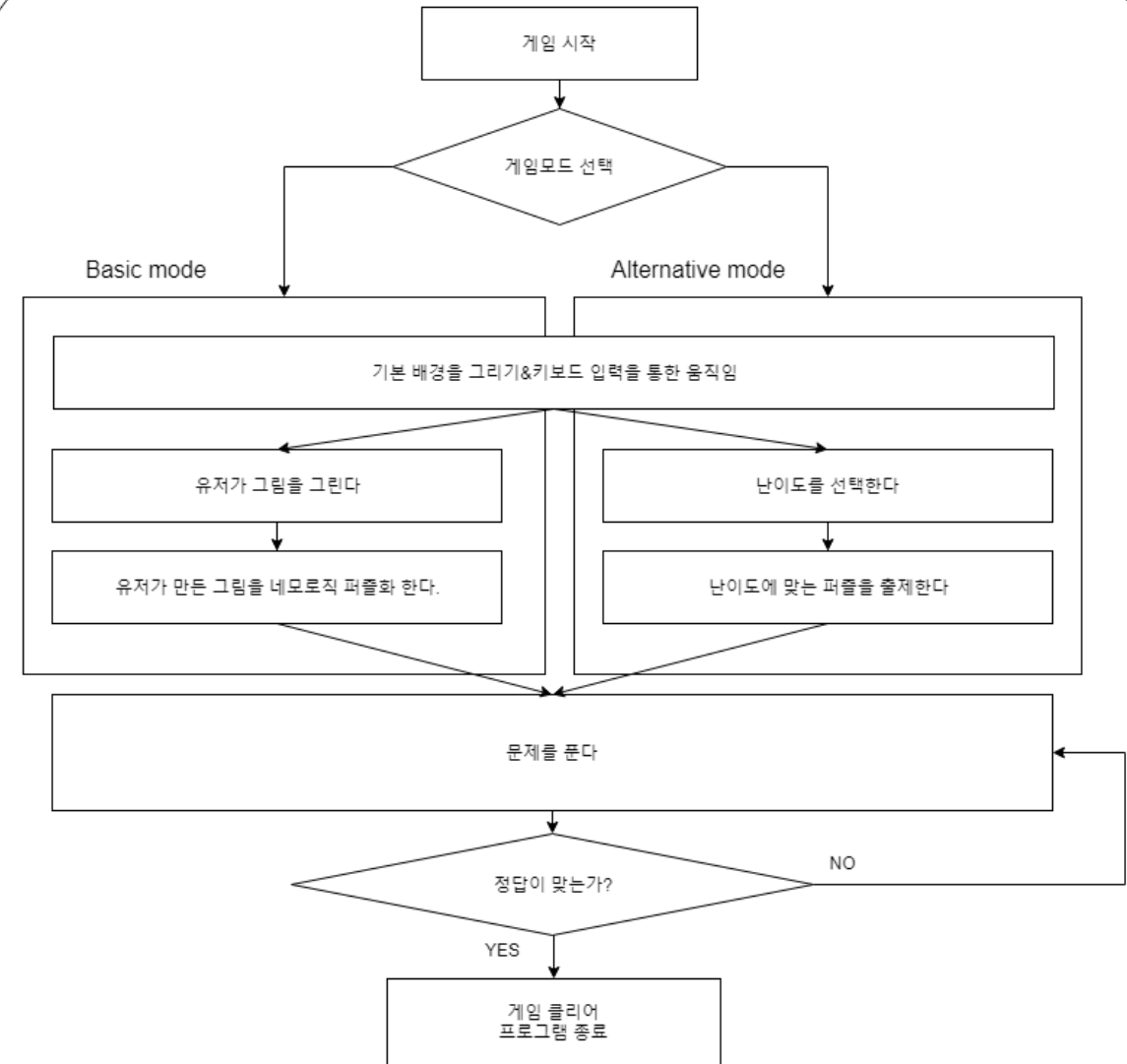
2-1-1. 문제화

2-2-1. 난이도 선택

2-1-2. 정답 확인

2-2-2. 정답 확인

3.종료



4.구현(헤더파일&전역변수&전역배열)

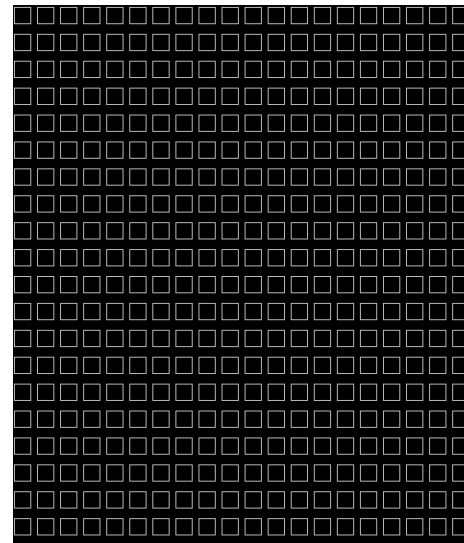
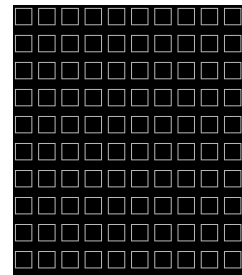
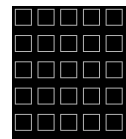
```
#include<stdio.h>
#include<windows.h>
#include<conio.h>
#include<mmsystem.h>
#pragma comment(lib, "winmm.lib");

static int mode, level; //전역변수 level
int x = 0, y = 0; //gotoxy에 필요한 전역변수
int a[20][20] = { 0 }, row[20][20] = { 0 }, col[20][20] = { 0 }, customrow[20][20] = {0}, customcol[20][20] = { 0 }, userQcol[20][20] = {0}, userQrow[20][20] =
//문제 배열들 0
```

Windows.h= gotoxy와 sleep함수 필요
Conio.h=getch를 입력받기위해 필요
Mmsystem.h=배경음악재생을 위해 필요

4. 구현(gotoxy 함수 & Map 함수)

```
void gotoxy(int x, int y)
{
    COORD Cur;
    Cur.X = x;
    Cur.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Cur);
} // gotoxy 함수 좌표값이동
```



```
void map(int level) {
    system("cls");
    if (level == 1) {
        level = 5;
    }
    else if (level == 2) {
        level = 10;
    }
    else if (level == 3) {
        level = 20;
    }
    int i, j;
    x = 0, y = 0;
    gotoxy(x, y);
    for (j = 0; j < level; j++) {
        for (i = 0; i < level; i++) {
            printf("□");
        }
        y++;
        gotoxy(x, y);
    }
}
```

```
gotoxy(60, 20);
printf("네모네모로직 ver 1.0");
gotoxy(60, 22);
printf("[a를 누르면 정답을 제출합니다.]");
gotoxy(60, 24);
printf("[r을 누르면 게임을 포기합니다.]");
gotoxy(60, 26);
printf("[x를 누르면 깃발을 세울 수 있습니다.]");
gotoxy(60, 28);
if (level == 5) {
    printf("[Easy Mode]");
}
else if (level == 10) {
    printf("[Normal Mode]");
}
else if (level == 20) {
    printf("[Hard Mode]");
}
```

// 네모네모로직 맵 구현 함수

네모네모로직 ver 1.0

[a를 누르면 정답을 제출합니다.]

[r을 누르면 게임을 포기합니다.]

[x를 누르면 깃발을 세울 수 있습니다.]

[Easy Mode]

[Normal Mode]

[Hard Mode]

4. 구현(Main & run 함수)

```
int main() {  
    PlaySound(TEXT("C:\\Users\\user\\Downloads\\bgm.wav"), NULL, SND_ASYNC | SND_LOOP);  
    title();  
    run();  
    system("pause");  
    return 0;  
} //메인함수
```

```
void run() {  
    system("cls");  
    selectMode();  
    selectLevel();  
    if ((mode == 1) && (level >= 1) && (level <= 3)) {  
        customGame(level);  
    }  
    else if ((mode == 2) && (level >= 1) && (level <= 3)) {  
        adventure(level);  
    }  
    else {  
        exit(1);  
    }  
} //전체적인 실행함수 (메인함수에서 실행됨)
```

4.구현(title & selcectLevel/Mode)

```
void title() {  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("    □    □ □  □□□□ □□□□ □□□□ □\n");  
    printf("    □    □ □  □  □  □□    □    □  □\n");  
    printf("    □    □ □  □□□□    □□    □  □  □\n");  
    printf("    □  □□□□ □    □    □□□□    □□□□□\n");  
    printf("    □□□□ □ □  □□□□ □□□□    □\n");  
    printf("\n");  
    printf("    □    □ □  □□□□ □□□□ □□□□ □\n");  
    printf("    □    □ □  □  □  □    □    □  □\n");  
    printf("    □    □ □  □□□□    □□    □  □  □\n");  
    printf("    □  □□□□ □    □    □□□□    □□□□□\n");  
    printf("    □□□□ □ □  □□□□ □□□□    □\n");  
    printf("\n");  
    printf("    계속하려면 아무키나 누르십시오.  \n");  
    _getch();  
}
```

//타이틀 출력 함수

```
□    □ □  □□□□ □□□□ □□□□ □  
□    □ □  □  □  □□    □    □  □  
□    □ □  □□□□    □□    □  □  □  
□  □□□□ □    □    □□□□    □□□□□  
□□□□ □ □  □□□□ □□□□    □
```

계속하려면 아무키나 누르십시오.

```
void selectMode() {  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("    모드를 선택하세요\n");  
    printf("1:Custom    2:Adventure >> ");  
    scanf_s("%d", &mode);  
}
```

//모드 선택 함수

모드를 선택하세요

1:Custom 2:Adventure >> _

```
void selectLevel() {  
    system("cls");  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("\n");  
    printf("    난이도를 선택해주세요.\n");  
    printf("1: 5 X 5    2: 10 X 10    3: 20 X 20 >>");  
    scanf_s("%d", &level);  
}
```

//레벨선택함수

난이도를 선택해주세요.

1: 5 X 5 2: 10 X 10 3: 20 X 20 >>

4. 구현(playGame 함수)

```
while (1) {
    curser = _getch();
    switch (curser) {
        case 72:
            if ((y - 1) != (-1))
                y -= 1;
            break;
        case 75:
            if ((x - 1) != (-1))
                x -= 1;
            break;
        case 80:
            if ((y + 1) != level)
                y += 1;
            break;
        case 77:
            if ((x + 1) != level)
                x += 1;
            break;
        case 32:
            if ((a[x][y] == 0) || (a[x][y] == 3)) {
                printf("■");
                a[x][y] = 1;
            }
            else if (a[x][y] == 1) {
                printf("□");
                a[x][y] = 0;
            }
            break;
    }
```

```
        case 'x':
            if ((a[x][y] == 0) || (a[x][y] == 1)) {
                printf("■");
                a[x][y] = 3;
            }
            else if (a[x][y] == 3) {
                printf("□");
                a[x][y] = 0;
            }
            break;
        case 'a':
            answer();
            break;
        case 'r':
            initRowcol();
            run();
            break;
        case 'z':
            saveCol();
            saveRow();
            map(level);
            initA();
            customProblem(level);
            playGamecustom();
            _getch();
            break;
    }
    gotoxy(x + 2, y);
}
```

```
void playGame(int level) {
    char curser;
    if (level == 1) {
        level = 5;
    }
    else if (level == 2) {
        level = 10;
    }
    else if (level == 3) {
        level = 20;
    }
}
```

72: Up
75: Left
80: Down
77: Right
32: Space

4.구현(커스텀 게임 함수)

```
void customGame(int level) {  
    customMap(level);  
    playGame(level);  
} // 커스텀 게임 실행 함수
```

```
gotoxy(60, 20);  
printf("네모네모로직 ver 1.0");  
gotoxy(60, 22);  
printf("[a를 누르면 정답을 제출합니다.]");  
gotoxy(60, 24);  
printf("[r을 누르면 게임을 포기합니다.]");  
gotoxy(60, 26);  
printf("[x를 누르면 깃발을 세울 수 있습니다.]");  
gotoxy(60, 28);  
printf("[z를 누르면 문제화 시킬 수 있습니다.]");
```

```
case 'a':  
    answer();  
    answerCustom();  
    customProblem(level);  
    break;  
  
case 'r':  
    initCustom();  
    run();  
    break;
```

```
void playGame(int level) {  
    char curser;  
    if (level == 1) {  
        level = 5;  
    }  
    else if (level == 2) {  
        level = 10;  
    }  
    else if (level == 3) {  
        level = 20;  
    }  
}
```

Playgame 함수와
playgameCustom은 거의 흡사하
지만 아주 살짝 다름

4.구현(answer)

```
void answer() {  
    //test 오른쪽 아래 칼럼, 로우 표시만 지우기  
    for (int i = 0; i < 20; i++) {  
        for (int j = 0; j < 20; j++) {  
            gotoxy(40 + 2 * j, i);  
            printf("                ");  
            gotoxy(2 * j, 20 + i);  
            printf(" ");  
        }  
    }  
    //row,col 배열 초기화  
    for (int i = 0; i < 20; i++) {  
        for (int j = 0; j < 20; j++) {  
            row[i][j] = 0;  
            col[i][j] = 0;  
        }  
    }  
    setRow();  
    setCol();  
} //초기화한후 퍼즐값을 변환하는 함수
```

유저가 사용할 때마다 값을 다시
보여줘야 함으로 초기화 한후

Row와 col배열을 설정

4. 구현(setRow)

```
void setRow() {
    int count = 0, temp = 0; //연속으로 10이면 count++, col변수 중복으로 덮어씌우지 않게

    for (int i = 0; i < 20; i++) {
        count = 0;
        for (int j = 0; j < 20; j++) {
            //마지막 칸일때
            if (j == 19) {
                if (a[j][i] == 1) {
                    count++;
                    row[temp][i] = count;
                }
                else
                    row[temp][i] = count;
            }
            else if (a[j][i] == 1) {
                count++;
            }
            else {
                row[temp][i] = count;
                temp++;
                count = 0;
            }
        }
        temp = 0;
    }
}
```

```
//row 행렬 출력
if (level == 20 && mode == 2) {
    for (int i = 0; i < 20; i++) {
        for (int j = 0; j < 20; j++) {
            if (row[j][i] != 0) {
                gotoxy(62 + (3 * temp), i);
                printf("%d ", row[j][i]);
                temp++;
            }
        }
        temp = 0;
        printf("\n");
    }
}
else if (level == 20 && mode == 1) {
    for (int i = 0; i < 20; i++) {
        for (int j = 0; j < 20; j++) {
            if (row[j][i] != 0) {
                gotoxy(82 + (3 * temp), i);
                printf("%d ", row[j][i]);
                temp++;
            }
        }
        temp = 0;
        printf("\n");
    }
}
else {
    for (int i = 0; i < 20; i++) {
        for (int j = 0; j < 20; j++) {
            if (row[j][i] != 0) {
                gotoxy(42 + (3 * temp), i);
                printf("%d ", row[j][i]);
                temp++;
            }
        }
        temp = 0;
        printf("\n");
    }
}
```

// 오른쪽에 저장되어있는 퍼즐값

4. 구현(setCol)

```
void setCol() {
    int count = 0, temp = 0; //연속으로 10이면 count , row변수 중복으로 덮어씌우지 않게
    for (int i = 0; i < 20; i++) {
        count = 0;
        for (int j = 0; j < 20; j++) {
            //마지막 칸일때
            if (j == 19) {
                if (a[i][j] == 1) {
                    count++;
                    col[i][temp] = count;
                }
                else
                    col[i][temp] = count;
            }
            else if (a[i][j] == 1) {
                count++;
            }
            else {
                col[i][temp] = count;
                temp++;
                count = 0;
            }
        }
        temp = 0;
    }
}
```

```
void saveCol() {
    int count, temp = 0; //연속으로 10이면 count ,row변수 중복으로 덮어씌우지 않게
    for (int i = 0; i < 20; i++) {
        count = 0;
        for (int j = 0; j < 20; j++) {
            //마지막 칸일때
            if (j == 19) {
                if (a[i][j] == 1) {
                    count++;
                    customcol[i][temp] = count;
                }
                else
                    customcol[i][temp] = count;
            }
            else if (a[i][j] == 1) {
                count++;
            }
            else {
                customcol[i][temp] = count;
                temp++;
                count = 0;
            }
        }
        temp = 0;
    }
}
```

4. 구현 (Problem 함수)

```
void problem(int level) {
    int temp = 0;
    if (level == 5) {
        //오른쪽 easy 문제 출력 row
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                if (easyQrow[j][i] != 0) {
                    gotoxy(12 + (3 * temp), i);
                    printf("%d ", easyQrow[j][i]);
                    temp++;
                }
            }
            temp = 0;
            printf("\n");
        }
        //아래 easy 문제 출력 col
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                if (easyQcol[i][j] != 0) {
                    gotoxy(i + 2, 6 + temp);
                    printf("%2d", easyQcol[i][j]);
                    temp++;
                }
            }
            temp = 0;
        }
    }
    else if (level == 10) {
        //오른쪽 easy 문제 출력 row
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                if (normalQrow[j][i] != 0) {
                    gotoxy(22 + (3 * temp), i);
                    printf("%d ", normalQrow[j][i]);
                    temp++;
                }
            }
            temp = 0;
            printf("\n");
        }
        //아래 easy 문제 출력 col
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                if (normalQcol[i][j] != 0) {
                    gotoxy(i + 2, 11 + temp);
                    printf("%2d", normalQcol[i][j]);
                    temp++;
                }
            }
            temp = 0;
        }
    }
    else if (level == 20) {
        //오른쪽 easy 문제 출력 row
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                if (hardQrow[j][i] != 0) {
                    gotoxy(42 + (3 * temp), i);
                    printf("%d ", hardQrow[j][i]);
                    temp++;
                }
            }
            temp = 0;
            printf("\n");
        }
        //아래 easy 문제 출력 col
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                if (hardQcol[i][j] != 0) {
                    gotoxy(i + 2, 21 + temp);
                    printf("%2d", hardQcol[i][j]);
                    temp++;
                }
            }
            temp = 0;
        }
    }
}
```


4.구현(answerCustom)

```
void answerCustom() {
    int judgeans = 0; //정답인지 확인하는 함수.
    for (int i = 0; i < 20; i++) {
        for (int j = 0; j < 20; j++) {
            if ((row[i][j] != customrow[i][j]) || (col[i][j] != customcol[i][j])) {
                gotoxy(48, 24);
                printf("[X]");
                gotoxy(43, 25);
                printf("정답이 아닙니다.");
                judgeans++;
            }
        }
    }

    if (judgeans == 0) {
        gotoxy(48, 24);
        printf("[O]");
        gotoxy(43, 25);
        printf("정답 입니다. ");
    }
}
```

4.구현(어드벤처 게임 함수)

```
void run() {  
    system("cls");  
    selectMode();  
    selectLevel();  
    if ((mode == 1) && (level >= 1) && (level <= 3) ) {  
        customGame(level);  
    }  
    else if ((mode == 2) && (level >= 1) && (level <= 3)) {  
        adventure(level);  
    }  
    else {  
        exit(1);  
    }  
} //전체적인 실행함수(메인함수에서 실행됨)
```

```
void adventure(int level) {  
    system("cls");  
    initRowcol();  
    map(level);  
    playGameforadventure();  
} //어드벤처게임실행함수
```

```
case 'a':  
    answer();  
    answerAdventure();  
    problem(level);  
    break;  
  
case 'r':  
    initA();  
    if (level == 5) {  
        drawEasy();  
    }  
    else if (level == 10) {  
        drawNormal();  
    }  
    else if (level == 20) {  
        drawHard();  
    }  
  
    fillLogic(level);  
    Sleep(3000);  
    initA();  
    run();  
    break;
```

playGame과
playGameforAdventure도 미세하
게 다른 점이 존재

4. 구현(색칠 함수)

```
void drawEasy() {  
    for (int i = 0; i < 5; i++) {  
        a[2][i] = 1;  
        a[i][4] = 1;  
    }  
    a[1][2] = 1;  
    a[3][2] = 1;  
    a[0][3] = 1;  
    a[4][3] = 1;  
}
```

////정답공개(쉬움)

```
void drawNormal() {  
    a[0][1] = 1;  
    a[9][1] = 1;  
    a[8][6] = 1;  
    a[6][7] = 1;  
    a[8][7] = 1;  
    a[9][7] = 1;  
    for (int i = 0; i < 3; i++) {  
        a[i][0] = 1;  
        a[i + 7][0] = 1;  
        a[i + 2][1] = 1;  
        a[i + 5][1] = 1;  
        a[i + 2][2] = 1;  
        a[i + 5][2] = 1;  
        a[i + 2][3] = 1;  
        a[i + 5][3] = 1;  
        a[i + 2][4] = 1;  
        a[i + 5][4] = 1;  
        a[i + 2][5] = 1;  
        a[i + 5][5] = 1;  
        a[i + 2][6] = 1;  
        a[i + 5][6] = 1;  
        a[i + 3][7] = 1;  
        a[i + 4][8] = 1;  
        a[i + 7][8] = 1;  
        a[i + 4][9] = 1;  
        a[i + 7][9] = 1;  
    }  
}
```

```
void drawHard() {  
    for (int j = 0; j < 20; j++) {  
        for (int i = 0; i < 20; i++) {  
            a[i][j] = 1;  
        }  
    }  
} //색칠
```

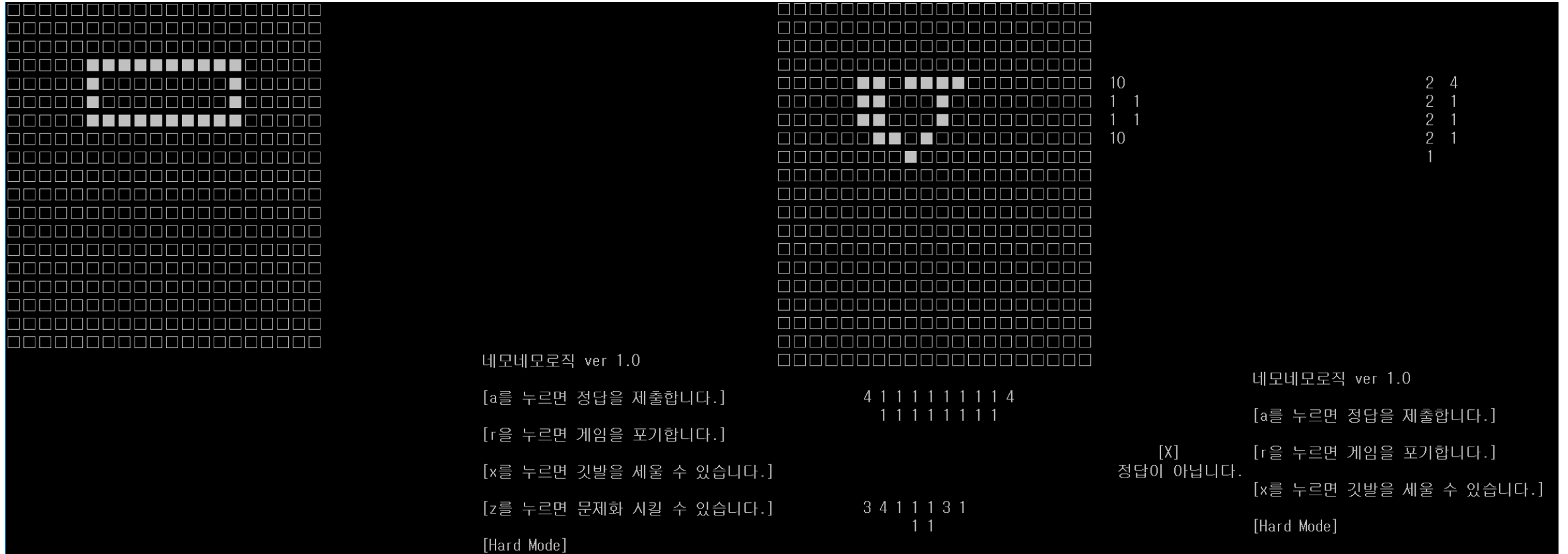
```
for (int i = 0; i < 2; i++) {  
    a[i + 9][1] = 0;  
    a[i + 2][6] = 0;  
    a[i + 16][6] = 0;  
    a[i + 8][9] = 0;  
    a[i + 8][10] = 0;  
    a[i + 2][11] = 0;  
    a[i + 2][12] = 0;  
    a[i + 8][17] = 0;  
    a[i + 11][16] = 0;  
    a[i + 11][17] = 0;  
}
```

//공백변환2

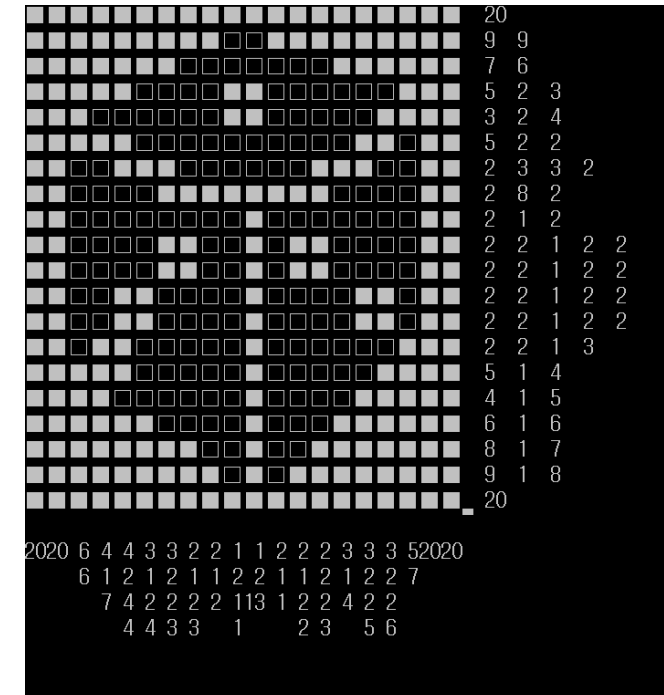
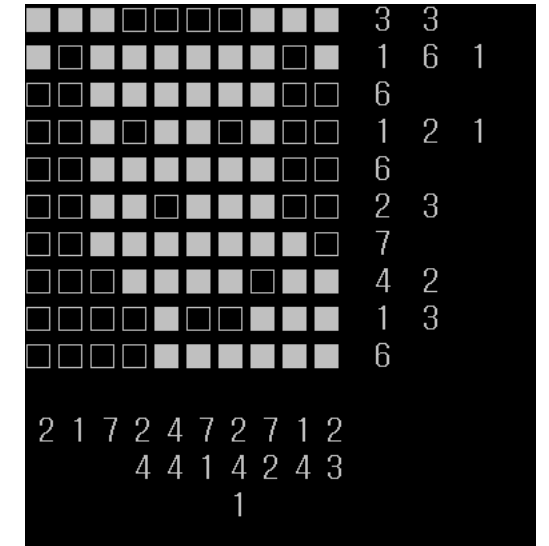
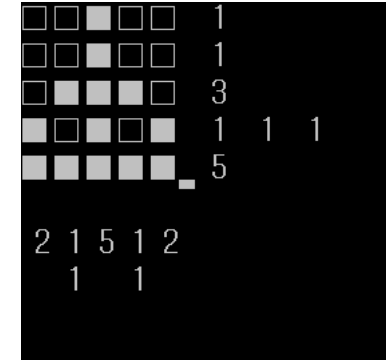
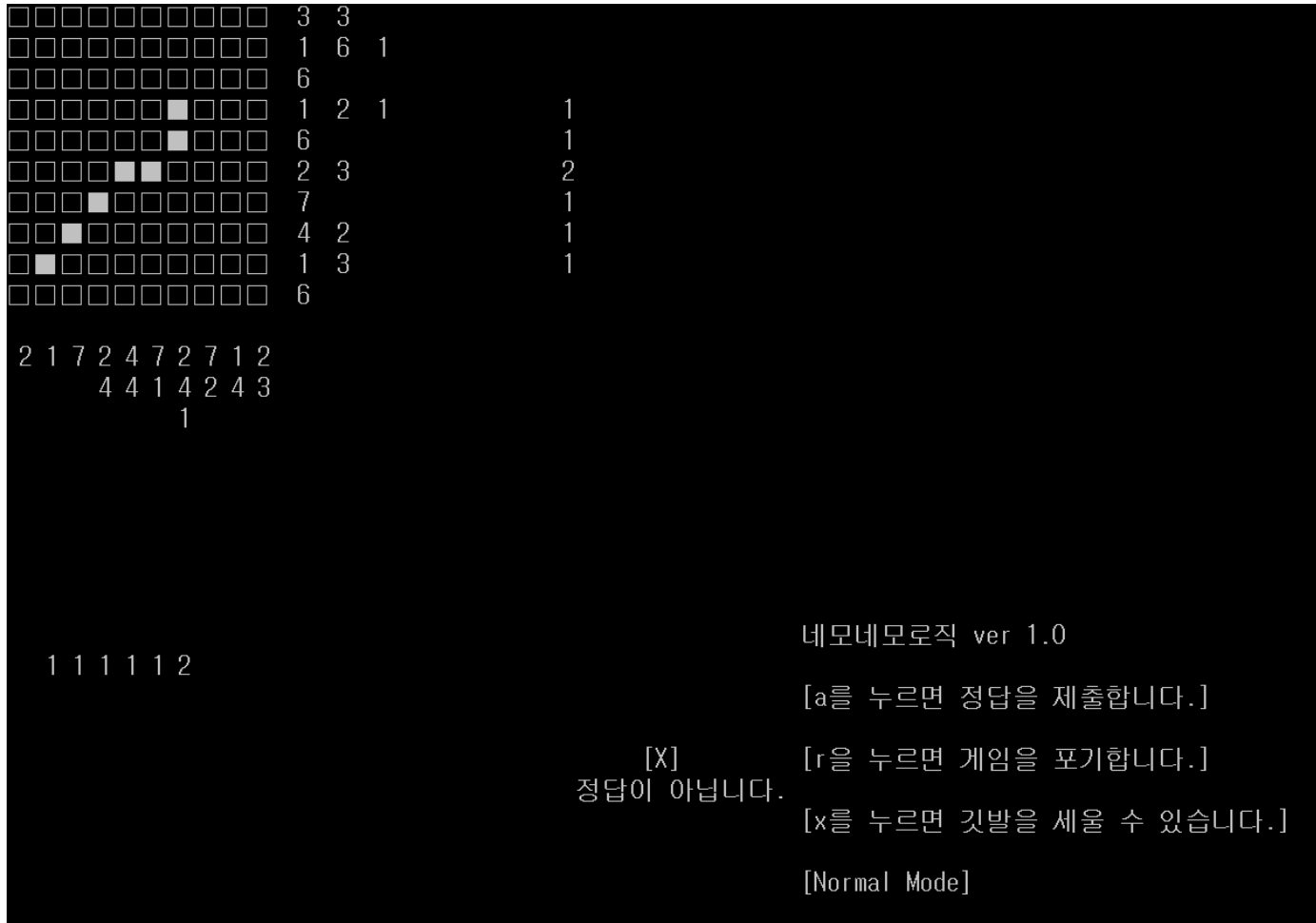
```
void fillLogic(int level) {  
    if (level == 1) {  
        level = 5;  
    }  
    else if (level == 2) {  
        level = 10;  
    }  
    else if (level == 3) {  
        level = 20;  
    }  
    x = 0, y = 0;  
    gotoxy(x, y);  
    for (int j = 0; j < level; j++) {  
        for (int i = 0; i < level; i++) {  
            gotoxy(j + 2, i);  
            if (a[j][i] == 1)  
                printf("■");  
            else if ((a[j][i] == 0) || (a[j][i] == 3)) {  
                printf("□");  
            }  
        }  
    }  
}
```

//정답공개함수

5.테스트(커스텀 게임화면)



5.테스트(어드벤처 게임화면)



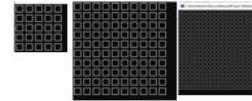
5.테스트

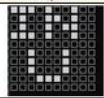
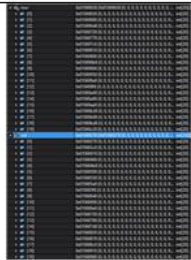

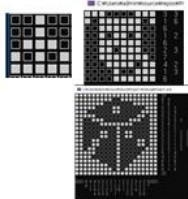
요구사항	테스트 방법
모드 선택 및 난이도 선택 (REQ-FR-1,2,3,8,9)	테스트시 모든 경우의 모드와 난이도를 실행
키보드 입력 게임 플레이 및 종료(REQ-FR-4,5,6,10,13)	각 모드에서 모든 키가 정상인지 확인
커스텀 모드 문제 만들기 및 풀기 (REQ-FR-7,11)	커스텀 모드에서 문제 퍼즐화와 풀기가 정상적인지 확인한다.
게임 배경음악 재생(REQ-FR-14,15)	게임 실행 중 배경음악은 잘 재생되는지 확인
시스템 초기화 함수(REQ-II-1,2)	시스템 초기화 함수가 정상적으로 작동하는지 확인
화면에 x,y좌표에 원하는 값을 출력하는 함수(REQ-II-5)	정답을 알려주는 filllogic 함수가 정상적인지 확인
어드벤처 모드에서 정상적으로 문제가 출제되는지 확인한다.	어드벤처모드에서 정상적인 퍼즐인지 확인

5.테스트

게임 1차	2차	3차	4차	5차	6차	7차	8차
정상	정상	정상	정상	정상	정상	버그 발견	정상

모드선택 및 난이도 선택	easy	normal	Hard
어드벤처 모드	정상	정상	정상
커스텀 모드	정상	정상	정상

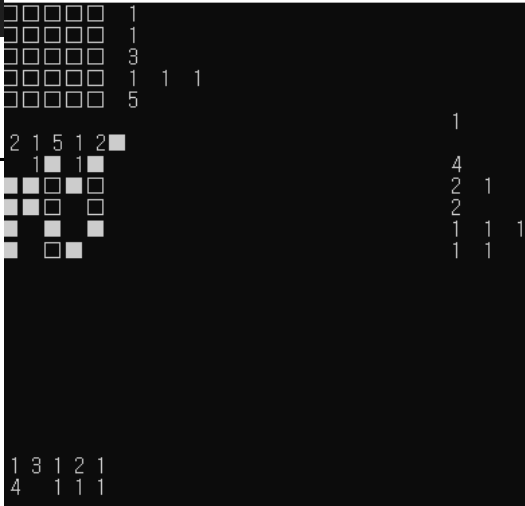


	자료	결과
커스텀모드		정상
게임 배경음악		정상
시스템 초기화 함수		정상
문자 가 호기 출력		filllogic 함수 정상 작동
어드벤처 모드		정상적으로 출력이 가능하다

5.테스트 오류추적 및 해결방안

```
void gotoxy(int x, int y)
{
    COORD Cur;
    Cur.X = x;
    Cur.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Cur);
} //gotoxy함수 좌표값이동
```

위 함수 사용시 커서의 위치 초기화



커서가 맵 밖에 있는데 값을 변경한 경우

6.유지보수

- 함수가 재귀적으로 짜여 있어 너무 오래 플레이하면 메모리 과부하 발생 가능
- 같은 방식의 함수 재사용 -> 간결화 했으면 좋았을 듯