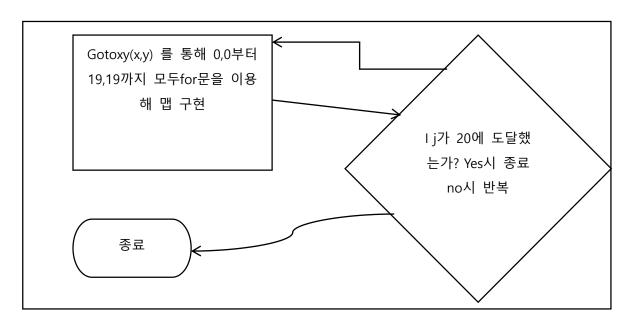
LLD 하위 설계 시스템

1. 맵 출력 함수

Void map(){} 인풋파라미터x 아웃풋 파라미터 로직의 출력값의 초기값설정과 비어있는 네모 설정



```
void map(){

int i;

int j;

gotoxy(x, y);

for (j = 0; j < 20; j++) {

for (i = 0; i < 20; i++) {

printf("□");

}

y++;

gotoxy(x, y);

}
```

2. 게임구동함수

Void playGame(){} 인풋 파라미터 x 아웃풋 파라미터 로직의 배열값과 출력되는 그림 변경

```
지역변수:
char curser

While

Switch Case: 키보드 입력(상하좌우)을 받아 로직에서 자신이 원하는
위치로 이동합니다.
스페이스바를 누르면 정답체크를 하며 그 위치를 까맣게 색칠합니다.
배열 a[x][y]가 □일때는 int값이 0이고 ■일때는 값이 1
지뢰찾기의 x처럼 ®는 3의 값
X를 치려면 자판 x를 누르면 ® 로 칠할 수 있습니다.
```

```
case 80:

if (y + 1 != 20)

y += 1;

break;

case 77:

if (x + 1 != 20)

x += 1;

break;

case 32:

if (a[x][y] == 0) {

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) {

printf("□");

a[x][y] = 0;

}

break;}

gotoxy(x * 2, y);

}
```

3. 정답공개 하는 함수

Void fillLogic() 인풋 x 아웃풋 파라미터 로직배열의 출력되는 그림 변경해줌.

Gotoxy()를 이용해 모든 배열을 For루프를 이용해 방문해 a[x][y] 가 1이면 그 자리를 ■로 칠한 다

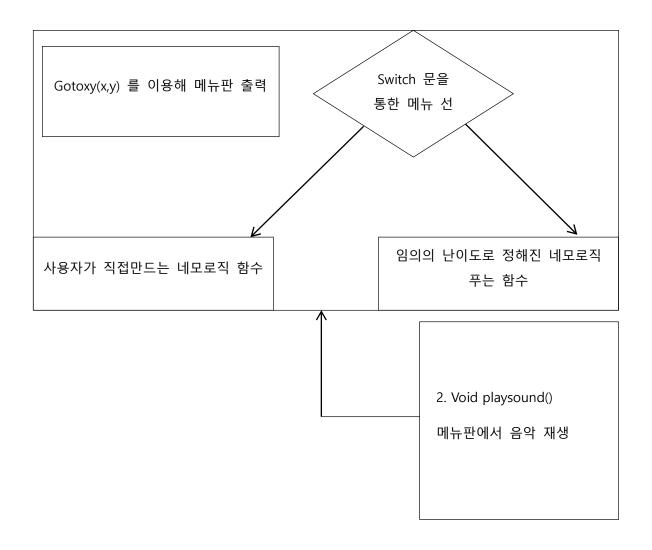
저장될 문제들

```
void fillLogic() {
    x = 0;
    y = 0;

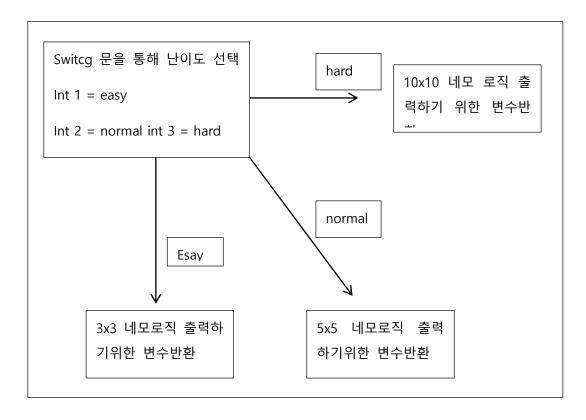
gotoxy(x, y);

for (int j = 0; j < 40; j++) {
    for (int i = 0; i < 20; i++) {
        gotoxy(j, i);
        if (a[j][i] == 1)
        printf("■");
    }
}</pre>
```

4. 모드선택 함수 void Selcet mod () = 인풋파라미터 x 아웃풋파라미터 void



5. int select difficult() 인풋 파라미터 x 아웃풋 파라미터 int (이 변수를 통해 난이도 구별)



6. 문제푸는 함수 void solve problem() 인풋 파라미터 x 아웃풋 파라미터 void

