一、项目环境

2022版Visual Studio, 64位 Windows 10操作系统

二、程序运行过程

1.读入地图文件

传入参数：地图文件地址和名称
需要模块：读地图文件模块，参数为地图文件地址和名称，返回值为存有初始地图的map结构体

2.展示地图+输入指令

等待输入指令
传入参数：输入的指令
需要模块：读指令模块，无参数，返回值为指令序列数组；显示模块，展示初始地图

3.计算每步状态

根据初始地图和输入指令，计算每执行一个指令后地图和机器人状态
传入参数：存有初始地图的map结构体，指令序列数组
需要的模块：计算模块，参数为存有初始地图的map结构体，指令序列结构体数组，返回值为状态结构体数组，数组每个元素为执行一个指令后的游戏状态结构体，游戏状态结构体应包括机器人状态和地图状态的信息

4.根据输出的计算状态绘图

根据过程数组绘制每步的地图
传入参数：游戏状态结构体数组
需要的模块：绘图模块，根据游戏状态结构体绘制每步的情形并存盘
返回值：保存文件的路径数组
传出显示每步状态的bmp文件

5.显示状态

在命令行里逐一显示状态图片
需要的模块：显示模块
传入参数：保存路径数组
功能：在命令行里逐张显示状态图片

三、公共结构规定

位置

```
struct Position {
        int x, y; // x 表示列号，y 表示行号
};
```

机器人

```
struct Robot {
        Position pos; // 机器人位置
        int direction; // 机器人朝向，1前，2左，3下，4右
};
```

单元格

```
struct Cell {
        int height; // 高度
        int light_id; // 灯标识，0表示该单元格上没有灯
        bool robot; // true/false分别表示机器人在/不在该单元格上
    int light;//灯状态，0代表没有灯，1代表有灯没点亮，2代表有灯且点亮
};
```

一个地图状态

```
struct Map {
        // 单元格组成二维数组，MAX_ROW、MAX_COL为合理常数
        Cell cells[MAX_ROW][MAX_COL];
        int row, col;
        // 有效行数、有效列数
        int num_lights;
        // 地图上同时只有一个机器人
        Robot robot;
        // 每个过程的指令数限制
};
```

指令序列

```
enum OpType { TL, TR, MOV, JMP, LIT, CALL, P1, P2};
// TL为左转，TR为右转，MOV为向前行走，JMP为跳跃，LIT为点亮灯，P1和P2代表函数指令序列；
// 使用CALL表示调用MAIN，P1/2表示调用P1/2。
```