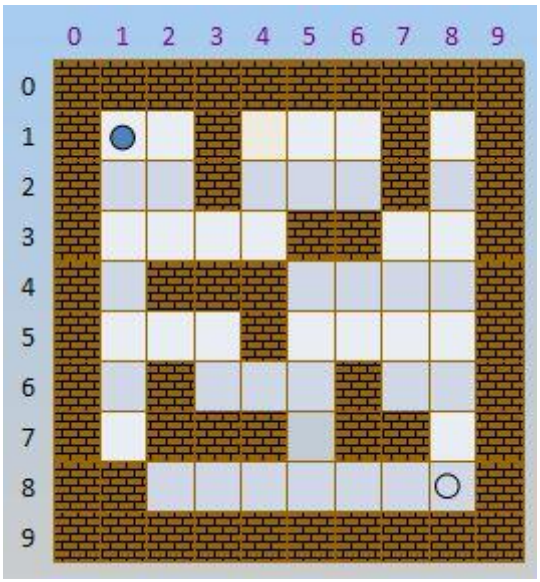
6--迷宫问题

本节目标

- **迷宫问题求解**
- **迷宫最短路径求解**

1. 迷宫问题求解



https://www.nowcoder.com/questionTerminal/cf24906056f4488c9ddb132f317e03bc

以上迷宫OJ题曾经是百度某一年的其中一个笔试题，迷宫问题本质就是一个图的遍历问题，从起点开始不断四个方向探索，直到走到出口，走的过程中我们借助栈记录走过路径的坐标，栈记录坐标有两方面的作用，一方面是记录走过的路径，一方面方便走到死路时进行回溯找其他的通路。

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<assert.h>

typedef struct Position
{
    int row;  // 行
    int col;  // 列
}PT;

typedef PT STDataType;
typedef struct Stack
{
    STDataType* a;
    int top;    // 栈顶下标
    int capacity;
}Stack;

// 初始化和销毁
void StackInit(Stack* pst);
```

```c
void StackDestory(Stack* pst);

// 入栈
void StackPush(Stack* pst, STDataType x);
// 出栈
void StackPop(Stack* pst);
// 获取数据个数
int StackSize(Stack* pst);
// 返回1是空，返回0是非空
int StackEmpty(Stack* pst);
// 获取栈顶的数据
STDataType StackTop(Stack* pst);

// 初始化和销毁
void StackInit(Stack* pst)
{
    assert(pst);
    pst->a = (STDataType*)malloc(sizeof(STDataType)*4);
    pst->top = 0;
    pst->capacity = 4;
}

void StackDestory(Stack* pst)
{
    assert(pst);
    free(pst->a);
    pst->a = NULL;
    pst->top = pst->capacity = 0;
}

// 入栈
void StackPush(Stack* pst, STDataType x)
{
    assert(pst);
    // 空间不够则增容
    if (pst->top == pst->capacity)
    {
        pst->capacity *= 2;
        STDataType* tmp = (STDataType*)realloc(pst->a, sizeof(STDataType)*pst-
>capacity);
        if (tmp == NULL)
        {
            printf("内存不足\n");
            exit(-1);
        }
        else
        {
            pst->a = tmp;
        }
    }

    pst->a[pst->top] = x;
    pst->top++;
}

// 出栈
void StackPop(Stack* pst)
{
```

```
    assert(pst);
    assert(pst->top > 0);

    --pst->top;
}

// 获取数据个数
int StackSize(Stack* pst)
{
    assert(pst);
    return pst->top;
}

// 返回1是空，返回0是非空
int StackEmpty(Stack* pst)
{
    assert(pst);

    return pst->top == 0 ? 1 : 0;
    //return !pst->_top;
}

// 获取栈顶的数据
STDataType StackTop(Stack* pst)
{
    assert(pst);
    assert(pst->top > 0);

    return pst->a[pst->top - 1];
}

//////////////////////////////////////////////////////////////////////////


// 打印迷宫
void PrintMaze(int** maze, int m, int n)
{
    for(int i = 0; i < m; ++i)
    {
        for(int j = 0; j < n; ++j)
        {
            printf("%d ", maze[i][j]);
        }
        printf("\n");
    }
}

bool IsPass(int** maze,int m, int n, PT p)
{
    if(p.row >= m || p.row < 0 || p.col >= n || p.col < 0)
    {
        return false;
    }

    if(maze[p.row][p.col] == 0)
        return true;
    else
        return false;
```

```
}

// 默认[0,0]是入口，[m-1][n-1]是出口
void GetMazePath(int** maze, int m, int n)
{
    // 核心逻辑
}

int main()
{
    int m = 0, n = 0;

    while(scanf("%d %d", &m, &n) != EOF)
    {
        // 创建迷宫，获取迷宫地图
        int** maze = (int**)malloc(sizeof(int*)*m);
        for(int i = 0; i < m; ++i)
        {
            maze[i] = (int*)malloc(sizeof(int)*n);
        }

        for(int i = 0; i < m; ++i)
        {
            for(int j = 0; j < n; ++j)
            {
                scanf("%d", &maze[i][j]);
            }
        }

        //PrintMaze(maze, m, n);

        GetMazePath(maze, m, n);

        // 销毁迷宫
        for(int i = 0; i < m; ++i)
        {
            free(maze[i]);
        }
        free(maze);
    }

    return 0;
}
```

2.迷宫最短路径求解

> https://www.nowcoder.com/questionTerminal/571cfbe764824f03b5c0bfd2eb0a8ddf
>
> 本题在曾经是美团某一年的笔试题，本题在上一个题的基础上计入了体力值的概念，但是本题还有一个隐藏条件，就是要找出迷宫的最短路径，如下图的两个测试用例，需要找出最短路径，才能通过全部测试用例。
>
> 4 4 10 1 0 0 1 1 1 0 1 0 1 1 1 0 0 1 1
>
> 10 10 50 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 1 0 0 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 1 0 0 1 1 1 1 1 1 1