P1443 马的遍历

题目描述

有一个 n×m 的棋盘，在某个点 (x,y) 上有一个马，要求你计算出马到达棋盘上任意一个点最少要走几步。

输入格式

输入只有一行四个整数，分别为 n,m,x,y。

输出格式

一个 n×m 的矩阵，代表马到达某个点最少要走几步（左对齐，宽5格，不能到达则输出-1）。

输入样例

3 3 1 1

输出样例

0 3 2
3 -1 1
2 1 4

解析

关键字“最少步骤”，当仁不让的选择广搜算法。

编码

```
#include<bits/stdc++.h>

using namespace std;
//最大的边界值
int const maxRow = 405;
int const maxCol = 405;

struct Node {
    int Row; //行数
    int Col; //列数

    //有参数的构造函数
    Node(int row, int col) {
        this->Row = row;
```

```
        this->Col = col;
    }

    //无参数的构造函数
    Node() {};
};

//当前地图的宽高
int gRow, gCol;
//答案数组
int ans[maxRow][maxCol];
//搜索队列，队列版
queue<Node> SearchQueue;

int Forward[8][2] = {
        //x描述的纵向变化，y描述的是横向变化
       {1,  -2}, //左下1日
       {2,  -1}, //左下2日
       {2,  1},  //右下2日
       {1,  2},  //右下1日

       {-1, 2},  //右上2日
       {-2, 1},  //右上1日
       {-1, -2},  //右上2日
       {-2, -1},  //右上1日
};

//检测节点是否有效，可以被存储
bool CheckNode(int row, int col) {
    //1、是否越界
    //2、是否被访问过
    if (row >= 1 && col >= 1 && row <= gRow && col <= gCol) {
        if (-1 == ans[row][col]) {
            return true;
        }
    }
    return false;
}

//广度优先搜索（通过队列）
void BfsByQueue(Node start) {
    memset(ans, -1, sizeof(ans));
    //将起点的步数设置为0
    ans[start.Row][start.Col] = 0;
    //对列中有值,清空队列
    while (!SearchQueue.empty()) {
        SearchQueue.pop();
    }
    //将起点放到队列
```

```
    SearchQueue.push(start);
     //启动搜索循环
    while (!SearchQueue.empty()) {
         //取出第一个点
        Node curNode = SearchQueue.front();
         //从队列中删除
        SearchQueue.pop();
         //当前步数
        int step = ans[curNode.Row][curNode.Col];
         //查找八个可以走的方向
        for (int i = 0; i < 8; ++i) {
             int newRow = curNode.Row + Forward[i][0];
             int newCol = curNode.Col + Forward[i][1];
             if (CheckNode(newRow, newCol)) {
                 //将当前节点存入队列
                 SearchQueue.push(Node(newRow, newCol));
                 ans[newRow][newCol] = step + 1;
             }
         }
     }
}

int main() {
     //输入基本数据
    cin >> gRow >> gCol;
     int startX, startY;
     cin >> startX >> startY;
     //定义起点，开始执行无限制搜索
    Node start = Node(startX, startY);
     BfsByQueue(start);
     //打印到达每个终点的最小步数
    for (int i = 1; i <= gRow; ++i) {
         for (int j = 1; j <= gCol; ++j) {
             printf("%-5d", ans[i][j]);
         }
         cout << endl;
     }
     return 0;
}
```