蛟龙四班

经典宽度优先搜索问题

Mas

#2122、二进制迷宫

题目描述

给定一个 $n \times m$ 的矩阵，其中矩阵中的每个数字表示这个格子能否到达四周的格子。

每个数字的范围是 $0 \sim 15$，即 4 位二进制数。

其中，二进制的最低位表示能否到达上面的格子，倒数第二位表示能否到达右面的格子，第二位表示能否到达下面的格子，第一位表示能否到达左面的格子。如果为 1 则为不能到达，如果为 0 则可以到达。

输入格式

第一个行输入俩个整数 $n\ m$ 代表矩阵的长宽。

接下来的 n 行，每行输入 m 个整数，代表矩阵的连通性，保证矩阵是合法的。接下来输入四个整数 x_1, y_1, x_2, y_2 。分别代表 A, B 俩点的坐标。（$1 \le x_1, x_2 \le n$， $1 \le y_1, y_2 \le m$）

输出格式

输出一个整数，代表从 A 点走到 B 点的最短距离，如果俩个点之间无法到达输出 `-1` 。

数据约定：

对于 30% 的数据：$1 \le n, m \le 5$ 。

对于 100% 的数据：$1 \le n, m \le 500$ 。

样例输入

```
2 4
13 5 5 3
13 5 5 6
1 1 2 1
```

样例输出

```
7
```

```cpp
int bfs()
{
  queue<node> q;
  vis[sx][sy] = true, q.push({sx, sy, 0});
  while (!q.empty())
  {
    cur = q.front(), q.pop();
    if (cur.x == ex && cur.y == ey)
      return cur.step;
    for (int i = 0; i < 4; i++)
    {
      int tx = cur.x + dir[i][0], ty = cur.y + dir[i][1];
      int ban = a[cur.x][cur.y] >> i & 1; //拆分二进制位
      if (tx >= 1 && tx <= n && ty >= 1 && ty <= m && !ban && !vis[tx][ty])
      {
        vis[tx][ty] = true;
        q.push({tx, ty, cur.step + 1});
      }
    }
  }
  return -1;
}
```

#261、奋斗的小强

题目描述

在一条直线上，小强初始在坐标 0 位置。每次他可以选择如下三种操作中的一种：

- 向左一个位置，即从位置 x 移动到位置 $x-1$
- 向右一个位置，即从位置 x 移动到位置 $x+1$
- 位置倍增，即从位置 x 移动到位置 $x \times 2$

现在有 Q 次询问 q_i ，对于每次询问，小强想知道他每次从位置 0 出发，到达位置 q_i 最少需要多少次操作?

输入格式

第一行一个 Q ，表示询问次数。

第二行 Q 个整数 q_i ，表示每次询问的目标位置。

输出格式

对于每个询问，输出一行一个整数，表示到达的目标位置的最少操作次数。如果无法到达，输出 -1 。

样例输入1

```
4
0
-1
2
4
```

样例输出1

```
0
1
2
3
```

范围说明

对于 20% 的数据有： $1 \leq Q \leq 10, 0 \leq q_i \leq 10$ 。

对于 50% 的数据有： $1 \leq Q \leq 1000, 0 \leq q_i \leq 1000$ 。

对于 100% 的数据有： $1 \leq Q \leq 10^5, -10^6 \leq q_i \leq 10^6$ 。

#1169、奇怪的电梯

每一次询问进行BFS,时间复杂度为$O(10^6)$,空间复杂度$O(2\times10^6)$

总时间复杂度为$O(Q\times10^6)$

不难发现 $0\to-100$ 的最少步数与$0\to100$的最少步数一致

且由于起点固定,仅需要做一次BFS,将对于每一个到达的点x,记录在数组dis中

对于每一次询问直接输出 $dis[\,|q_i|]$

时间复杂度$O(Q+10^6)$

#569、八数码问题

题目描述

在 3×3 的棋盘上，摆有八个棋子，每个棋子上标有 1 至 8 的某一数字。棋盘中留有一个空格，空格用 0 来表示。
空格周围的棋子可以移到空格中。
要求解的问题是：给出一种初始布局（初始状态）和目标布局（为了使题目简单,设目标状态为 123804765 ），找到一种最少步骤的移动方法，实现从初始布局到目标布局的转变。

> *保证在最少步数不超过* 20

输入

输入初试状态，一行九个数字，空格用 0 表示

输出

只有一行，该行只有一个数字，表示从初始状态到目标状态需要的最少移动次数(测试数据中无特殊无法到达目标状态数据)

样例输入

```
283104765
```

样例输出

```
4
```

#569、八数码问题

最小步数模型需要标记每个状态是否出现过

对于八数码问题,若使用十进制整数表示当前状态

最大数为876543210,需要约 $830MB$ 空间

可以使用 STL 中的 map 进行标记

map<string,bool> vis可暂时理解为下标为字符串的$bool$数组

观察每一个状态仅是0 ~ 8的一个排列

数组也许不需要开876543210 !?

思考:对于一个排列能否求出其排名/编号?

```
map<string, bool> vis;
struct node
{
  string state;
  int step;
} cur;
int bfs()
{
  queue<node> q;
  q.push({"123804765", 0});
  vis["123804765"] = true;
  while (!q.empty())
  {
    cur = q.front(), q.pop();
    if (cur.state == target)  return cur.step;
    int pos = cur.state.find('0'), row = pos / 3, col = pos % 3;
    for (int i = 0; i < 4; i++)
    {
      int tr = row + dir[i][0], tc = col + dir[i][1], tpos = tr * 3 + tc;
      if (tr >= 0 && tr < 3 && tc >= 0 && tc < 3)
      {
        tmp = cur.state;
        swap(tmp[pos], tmp[tpos]);
        if (!vis[tmp])
          q.push({tmp, cur.step + 1}), vis[tmp] = true;
      }
    }
  }
}
```

#1021、玉米迷宫

题目描述

今年秋天，约翰带着奶牛们去玩玉米迷宫。迷宫可分成 $N \times M$ 个格子，有些格子种了玉 米，种有玉米的格子无法通行。
迷宫的四条边界上都是种了玉米的格子，其中只有一个格子 没种，那就是出口。
在这个迷宫里，有一些神奇的传送点 6 每个传送点由一对点组成，一旦 走入传送点的某个结点， 机器就会强制把你送到传送点的另一头去。
所有的传送点都是双向的，如果你定到了另一头，机器也会把你送回来。
奶牛在一个单位的时间内只能向相邻的四个方向移动一格，不过传送机传送是瞬间完成的。现在 $Bessie$ 在迷宫里迷路了，她只知道目前的位置在哪里，请你帮助她用最短的时间走出迷宫吧。

输入格式

第一行：两个用空格分开的整数： N 和 M
第 $2 \sim N+1$ 行：第 $i+1$ 行有 M 个连续的字符，描述了迷宫第 i 行的信息。其中 `#` 代 表不能通行的玉米地， `.` 代表可以通行的草地， `@` 代表贝西的起始位置， `=` 代表迷宫出口，大写字母 `A` 到 `Z` 总是成对出现的，代表一对传送点。

输出格式

一个整数，表示 $Bessie$ 走出迷宫的最短时间，保证逃离迷宫的路线一定存在。

输入样例

```
5 6
###=##
#.W.##
#.####
#.@W##
######
```

输出样例

```
3
```

#1021、玉米迷宫

对于是普通格子,到达后重复到达无意义所以不允许重复经过

对于传送门,若是从普通格子到达后,不可再次从非传送门到达

但可从其他传送门传送到达,不应被标记

```
##########=#
#.........F#
#.##########
#..........#
##########@#
#..........#
#.##########
#.........F#
############
```

如果传送门有多个且步强制传送如何求出最小步数

```
vector<node> d[128];
int bfs()
{
  queue<node> q;
  q.push({sx, sy, 0});
  vis[sx][sy] = true;
  while (!q.empty())
  {
    cur = q.front(), q.pop();
    if (maze[cur.x][cur.y] == '=')
      return cur.step;
    for (int i = 0; i < 4; i++)
    {
      int tx = cur.x + dir[i][0], ty = cur.y + dir[i][1];
      if (tx < 0 || tx >= n || ty < 0 || ty >= m || vis[tx][ty] || maze[tx][ty] == '#')
        continue;
      vis[tx][ty] = true;
      if (!isalpha(maze[tx][ty]))
        q.push({tx, ty, cur.step + 1}); //不是传送门
      else
        for (auto &&i : d[maze[tx][ty]]) //强制传送
          if (tx != i.x || ty != i.y)
            q.push({i.x, i.y, cur.step + 1});
    }
  }
  return -1;
}
```

#898、迷宫路径

题目描述

给定一个 $n \times m$ 个字符迷宫,起点使用 `A` 表示,终点使用 `B` 表示。
请你计算最少需要多少步才能走出迷宫,并且输出距离路径

输入格式

第一行输入两个正整数 n, m
接下来 m 行 m 列表示迷宫,

- `.` 表示空地
- `#` 表示墙壁(不能通信)
- `A` 表示起点
- `B` 表示终点

输出格式

输入无法到达输出 `NO`
否则第一行输出 `YES` ,第二行输出最少需要的步数,第三行输出一行具体行走方向

- `U` 表示向上移动
- `D` 表示向下移动
- `L` 表示向左移动
- `R` 表示向右移动

当存在多组解时,输出字典序最小的一组行走方向

> *若每次优先选择 $ASCII$ 最小的方向移动,那么最终方案则为字典序最小*

输入样例1

```
5 8
########
#.A#...#
#.##.#B#
#......#
########
```

输出样例1

```
YES
9
LDDRRRRRU
```

数据规模

对于 30% 的数据, $1 \leq n \leq m \leq 20$
对于 100% 的数据, $1 \leq n \leq m \leq 1000$

#898、迷宫路径

在BFS最小步数模型中,有时需要输出具体路径或方案

在扩展新节点时

使用二维数组pre记录方向数组中的编号,假设从$dir[3]$为往右的

若(x,y)向右走一步到达(tx,ty),那么$pre[tx][ty]=3$

当BFS结束搜索后,可从终点顺藤摸瓜递归输出路径

```
int bfs(int x, int y)
{
  queue<node> q;
  vis[x][y] = true, q.push({x, y, 0});
  while (q.size())
  {
    cur = q.front(), q.pop();
    if (cur.x == ex && cur.y == ey)
      return cur.step;
    for (int i = 0; i < 4; i++)
    {
      int tx = cur.x + dir[i][0], ty = cur.y + dir[i][1];
      if (tx >= 1 && tx <= n && ty >= 1 && ty <= m && maze[tx][ty] != '#' && !vis[tx][ty])
      {
        pre[tx][ty] = i; //记录方向
        vis[tx][ty] = true, q.push({tx, ty, cur.step + 1});
      }
    }
  }
  return -1;
}
void output(int x, int y)
{
  if (x == sx && y == sy)
    return;
  int d = pre[x][y];
  output(x - dir[d][0], y - dir[d][1]);
  putchar(td[d]);
}
```

谢谢观看