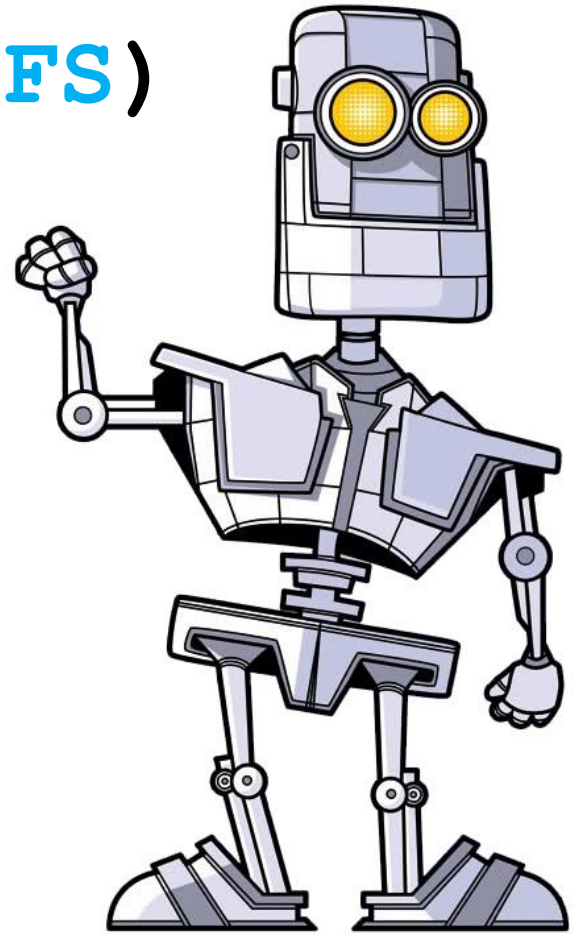
重庆南开信竞

广度优先搜索

Breadth First Search (BFS)

重庆南开



引例1：好人何老板nkoj1087

【题目描述】

王大爷在大街上摔倒了，恰好何老板下班路过，一项助人为乐的他赶紧抱起王大爷往医院跑。但好心的何老板面临着一个问题，城市里面有很多医院，到底哪家医院最近呢？

城市地图用一个由数字0，1，2，3构成的n*m方格矩阵表示(n,m<=1000)。格子中数字0表示可以行走的道路或空地。数字1表示王大爷摔倒的位置。数字2表示不可通过的建筑物或障碍物。数字3表示医院。

何老板只能延上下左右四个方向移动，每走一步可到达相邻方格。问到最近的医院需要走多少步？(地图中至少有一个可到达的医院)

【输入格式】

第一行，两个空格间隔的整数n和m

接下来是一个n*m的矩阵，用空格做间隔

【输出格式】

一个整数，表示最小的步数。

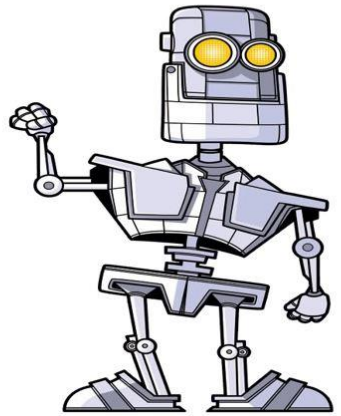
【样例输入】

```
5 8
3 0 0 0 0 2 0 3
2 0 0 2 3 0 2 0
0 2 0 2 0 3 0 2
0 1 0 2 0 0 0 0
0 0 0 0 0 0 0 3
```

【样例输出】

```
6
```

重庆南开信竞入门课程



通过宽搜求解：

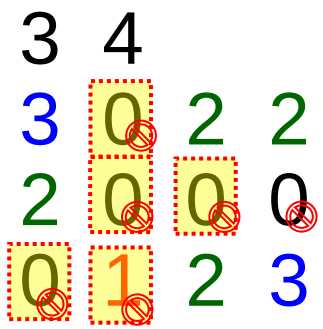
1.读入数据，找出起点坐标x_0，y_0

2.将起点加入队列。

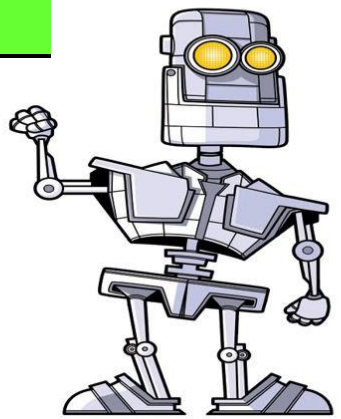
3.讨论队首元素的上下左右四个点，如果有值为3的点(目标点)，输出结果，程序结束。否则，如果值为0(可通过的点)，则将其入队，记录该点到起点的距离，**并将改点标记为不可通过**，队尾指针后移。

4.队首指针后移，执行第3步。

重庆南开信竞入门课程



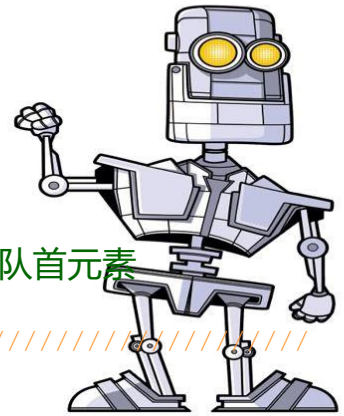
重庆南开信竞入门课程



```
struct node{int x,y,step;};                                  //x,y标记每个点的坐标，step记录该格子里起点的距离
int dx[4]={-1,1,0,0};   int dy[4]={0,0,-1,1};
queue<node>q;
int main()          //读入数据
{
        node Now,start,tmp;
        scanf("%d%d",&n,&m);
        for(i=1;i<=n;i++)
              for(j=1;j<=m;j++)
              {     scanf("%d",&a[i][j]);
                    if(a[i][j]==1){ start.x=i;  start.y=j;  start.step=0; }
              }
      ////////////////////////////////////////////////////宽搜 ////////////////////////////////////////////////////////////////////////////
      q.push(start);
      a[start.x][start.y]=2;                          //将起点标记为不可通过,避免死循环
      while(q.size())                                 //反复讨论，直到找出目标。若q.size()==0表示队列为空
       {
              Now=q.front();
              for(i=0;i<4;i++)                                          //讨论队首元素上下左右四个点。
              {
                      tx=Now.x+dx[i];          ty=Now.y+dy[i];          //dx[ ]和dy[ ]为增量数组
                      if((tx>0)&&(tx<=n)&&(ty>0)&&(ty<=m)&&(a[tx][ty]!=2))  //边界判断和可行判断
                      {
                               if(a[tx][ty]==3)                         //一旦找到目标，立即输出结果，程序结束。
                               {
                                        printf("%d\n",Now.step+1);
                                        return 0;
                               }
                              tmp.x=tx;     tmp.y=ty;                  //tmp记录新到达的点的信息
                              tmp.step=Now.step+1;                     //记录新到的点离起点的距离
                              q.push(tmp);                             //将新到达的点tmp加入队列
                              a[tx][ty]=2;                             //标记为不可通过，避免死循环
                      }
              }
              q.pop();                                  //讨论完当前队首元素后，删除当前队首，循环继续讨论新的队首元素
        }
      ////////////////////////////////////////////////////宽搜////////////////////////////////////////////////////////////////////////////
}
```

重庆南开信竞入门课程

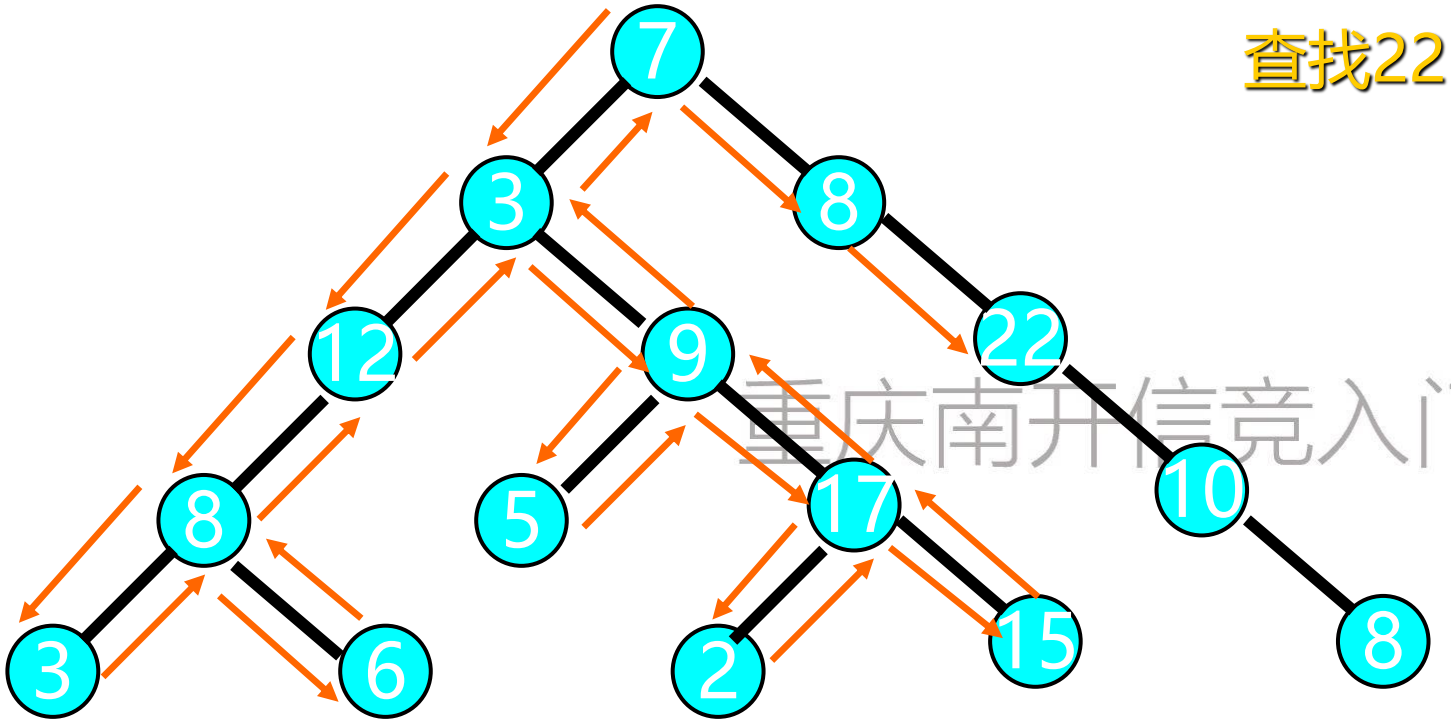
重庆南开信竞入门课程



深度优先搜索

从问题的某一种可能出发，搜索从这种情况出发所能达到的所有可能，当这一条路走到“尽头”而没找到解时，再倒回上一个步，从另一个可能出发，继续搜索.

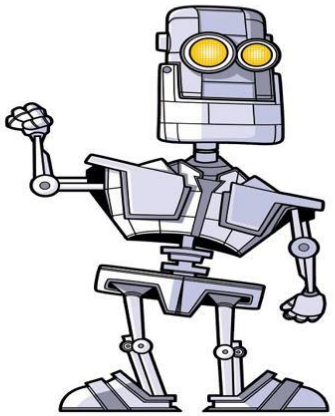
重庆南开信竞入门课程



查找22

重庆南开信竞入门课程

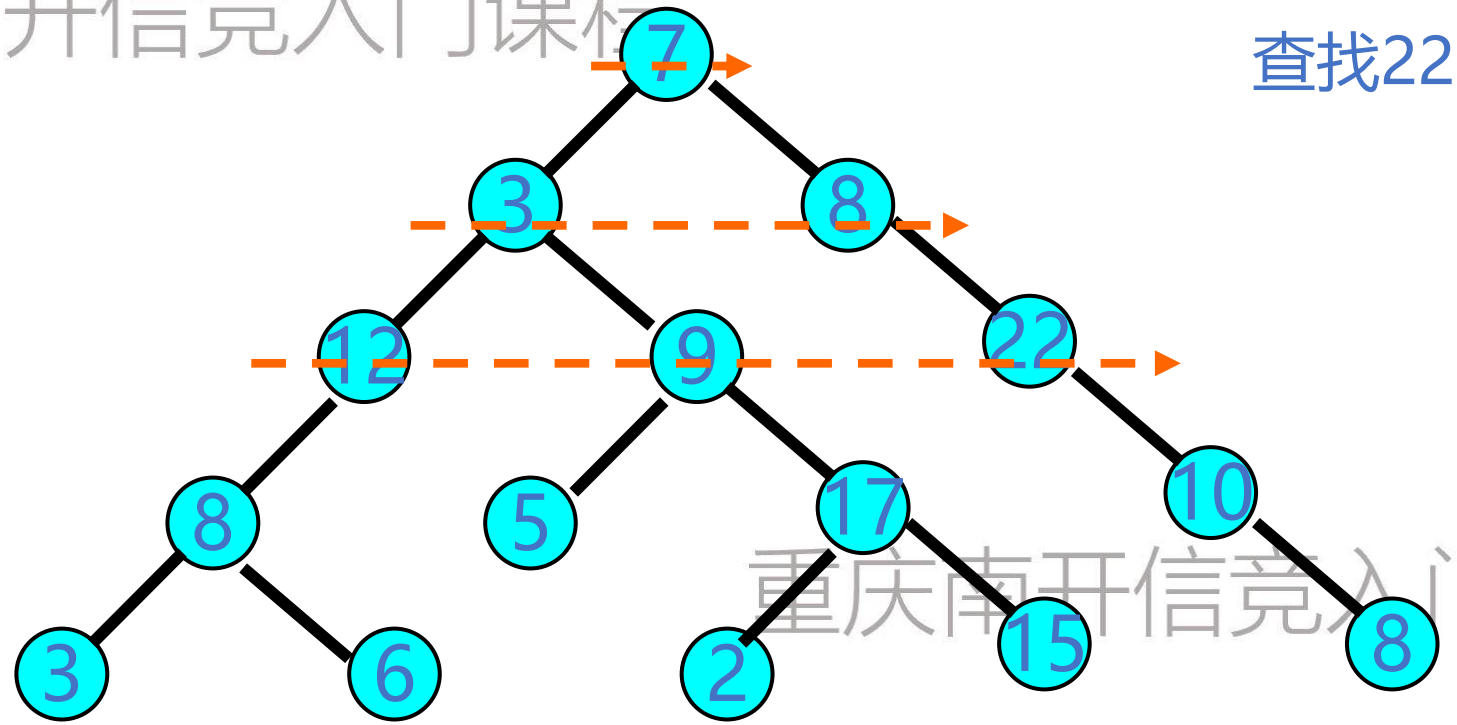
一直走到底，走不通就掉头试下一条路。



广度（宽度）优先搜索

从问题的起点出发, 逐层搜索所有的点.

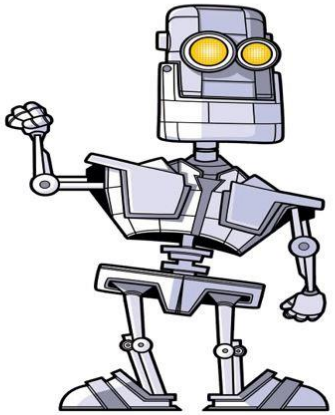
重庆南开信竞入门课程



查找22

重庆南开信竞入门课程

由于是逐层搜索，所以它总能保证找出的是离起点最近的点。所以比较适合用于求解最优值的问题。



例1：追牛nkoj1088

农民约翰在清点牛棚里的牛时，发现少了一头牛。他想尽快把牛找回来。约翰在离牛棚N米的地方通过望远镜发现了那头牛，那头牛正在距离牛棚K米的地方吃草(你可以理解为：牛棚、约翰和牛在一条直线上)。

重庆南开信竞入门课程

农民约翰可以通过两种方法去追牛：步行和瞬间移动。如果约翰所在的点离牛棚的距离为X，那么：

*步行：用一分钟，约翰可以走到点X-1或点X+1
*瞬间移动：用一分钟，约翰可以移动到点2*X

牛一直在原地吃草，不会移动。约翰追到牛最少需要多少分钟？

(K<N的情况是可能的)

输入格式:

一行，空格间隔的两个整数N和K(0<=N,K<=100000)

输出格式:

一行，一个整数，即最短时间。

重庆南开信竞入门课程

样例输入:

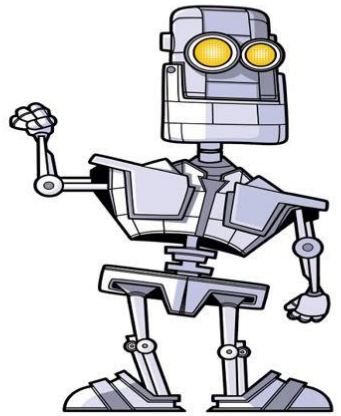
```
5 17
```

样例输出:

```
4
```

样例说明:

追上牛最快的方式是5->10->9->18->17



来源:USACO 2007 Open Silver

参考代码，核心部分：

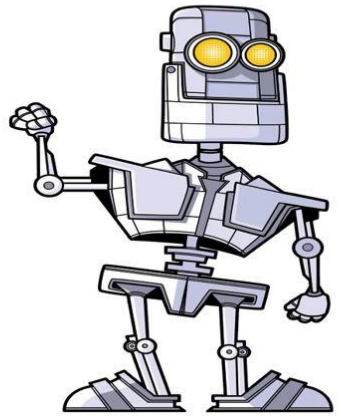
重庆南开信竞入门课程

```
bool Mark[300000];          //Mark[i]为true表示i号地点已经到过了。
struct node
{
        int Pos,Time;       //Pos记录当前约翰所在地点，Time记录到达当前地点所用时间
};
queue<node>Q;
node Next,Now;

int ans=0;

Now.Pos=n;
Now.Time=0;
Q.push(Now);                //宽搜开始，起点进队
```

重庆南开信竞入门课程



参考代码，核心部分：

```
while(Q.size())
{
        Now=Q.front();   Q.pop();
        if(Now.Pos+1==k || Now.Pos-1==k || Now.Pos*2==k){ ans = Now.Time+1; break; }
        if(Now.Pos+1<k && Mark[Now.Pos+1]==0)
        {
                Next.Pos = Now.Pos+1;
                Next.Time = Now.Time+1;
                Q.push(Next);
                Mark[Next.Pos] = 1;
        }
        if(Now.Pos-1>=0 && Mark[Now.Pos-1]==0)
        {
                Next.Pos = Now.Pos-1;
                Next.Time = Now.Time+1;
                Q.push(Next);
                Mark[Next.Pos] = 1;
        }
        if(Now.Pos<k && Mark[Now.Pos*2]==0)
        {
                Next.Pos = Now.Pos*2;
                Next.Time = Now.Time+1;
                Q.push(Next);
                Mark[Next.Pos] = 1;
        }
}
```

重庆南开信竞入门课程

重庆南开信竞入门课程