**《算法设计与分析》实验报告**

# 实验五 分支限界

|  |  |  |  |
| --- | --- | --- | --- |
| **报告书** | | | |
| 姓名 | 吴宇敖 | 指导教师 |  |
| 学号 | 176001752 | 日 期 | 2020.5.28 |
| 班级 | 17计算机软件一班 |  |  |
| **实验内容** | | | |
| **[要求先在OJ上提交通过]**  请给出AC截图和代码；无需说明。  对查重结果问题请在文档开头予以说明并给出前后源码。  1）再填格子  2）最短路径 | | | |
| **实验目的** | | | |
| 堂练目的：考察学生对本阶段知识点及编程掌握的熟练程度 | | | |
| **实验过程和步骤** | | | |
| **【实验题目】**  **题目描述：**  **有一个由数字 0、1 组成的方阵中，存在一任意形状的封闭区域，封闭区域由数字1 包围构成，每个节点只能走上下左右 4 个方向。现要求只把【最大封闭区域】内的空间填写成2 。**  **例如： 6×6 的方阵：**  **6**  **0 1 0 0 0 0**  **1 0 1 1 1 1**  **0 1 1 0 0 1**  **1 1 0 0 0 1**  **1 0 0 0 0 1**  **1 1 1 1 1 1**  **填写后如下：**  **0 1 0 0 0 0**  **1 0 1 1 1 1**  **0 1 1 2 2 1**  **1 1 2 2 2 1**  **1 2 2 2 2 1**  **1 1 1 1 1 1**  **输入要求：**  **每组测试数据第一行一个整数 n(1≤n≤30)**  **接下来 n 行，由 0 和 1 组成的 n×n 的方阵。**  **封闭区域内至少有一个0，测试数据保证最大区域只有一个。**  **输出要求： 已经填好数字 2 的完整方阵。(每个数字后面有一个空格！）**  **【程序代码及注释】**  #include<iostream>  using namespace std;  int num[100][100];//来记录每个格子应该填写的状态  int res[100][100]={0};//防止无限循环递归  int n;  int cnt=0;  int maxn=0;  void DFS(int a,int b)  {  if(num[a][b]==1||num[a][b]==-1)//当当前格子已经确定状态后，返回  return ;  if(a==0||b==0||a==n-1||b==n-1)//如果为边界，直接定义为a-1（就是不需填写内容）  num[a][b]=-1;  if(a!=0&&res[a-1][b]!=1)//向上  {  res[a][b]=1;  DFS(a-1,b);  if(num[a-1][b]==-1)//说明没有被包围，填写为-1  num[a][b]=-1;  }  if(a!=n-1&&res[a+1][b]!=1)//向下  {  res[a][b]=1;  DFS(a+1,b);  if(num[a+1][b]==-1)  num[a][b]=-1;  }  if(b!=0&&res[a][b-1]!=1)//向左  {  res[a][b]=1;  DFS(a,b-1);  if(num[a][b-1]==-1)  num[a][b]=-1;  }  if(b!=n-1&&res[a][b+1]!=1)//向右  {  res[a][b]=1;  DFS(a,b+1);  if(num[a][b+1]==-1)  num[a][b]=-1;  }  }  void dfs(int a,int b)  {  if(num[a][b]==1)//遇到后，返回  return ;  cnt++;//记录范围大小  if(a!=0&&res[a-1][b]!=1)//向上  {  res[a][b]=1;  dfs(a-1,b);  }  if(a!=n-1&&res[a+1][b]!=1)//向下  {  res[a][b]=1;  dfs(a+1,b);  }  if(b!=0&&res[a][b-1]!=1)//向左  {  res[a][b]=1;  dfs(a,b-1);  }  if(b!=n-1&&res[a][b+1]!=1)//向右  {  res[a][b]=1;  dfs(a,b+1);  }  }  int main()  {  cin>>n;  for(int i=0;i<n;i++)//初始化  for(int j=0;j<n;j++)  cin>>num[i][j];  for(int i=0;i<n;i++)  for(int j=0;j<n;j++)  if(num[i][j]==0)//只有为0时需要进行填写判断  {  for(int l=0;l<n;l++)//初始化res数组，来防止进行死循环  for(int k=0;k<n;k++)  res[l][k]=0;  DFS(i,j);  }  for(int i=0;i<n;i++)  for(int j=0;j<n;j++)  if(num[i][j]==0)  {  for(int l=0;l<n;l++)//初始化res数组，来防止进行死循环  for(int k=0;k<n;k++)  res[l][k]=0;  cnt=0;  dfs(i,j);  num[i][j]=2+cnt;  if(cnt>maxn)  maxn=cnt;  }  for(int i=0;i<n;i++)//修改填写的内容，使之符合题目要求  {  for(int j=0;j<n;j++)  {  if(num[i][j]==1)  num[i][j]=1;  else if(num[i][j]==maxn+2)  num[i][j]=2;  else num[i][j]=0;  }  }  for(int i=0;i<n;i++)//进行输出  {  for(int j=0;j<n;j++)  cout<<num[i][j]<<' ';  cout<<endl;  }  //system("pause");  return 0;  }  **【实验结果及遇到的问题】**      **【实验题目】**  **题目描述：**    **输入要求：**  第1行2个整数，表示区域的行数m和列数n。1<=m,n<=20  第2行4个整数，表示起点坐标和终点坐标，坐标计数从0开始。  第3行开始，m行n列的区域数据，0表示可通行，-1表示不可通行（图中绿色部分）。  **输出要求：**  如图a的二维信息数据，数值表示步数。起点终点分别用字符a、b表示。  最后与b同层的点，除了b之外，其他点无需标记。比如sample out只有b，没有9。  每个数值靠右占3位输出（含符号位），每行最后一个数值无空格换行。  详见sample output。（如无路径，按规则输出即可。）  **【程序代码及注释】**  #include<iostream>  #include <iomanip>  using namespace std;  int num[20][20];//记录全部路  int n,m;//n行，m列  int xa,xb,ya,yb;//起点和终点坐标  int res[20][20]={0};  int resn;  void BFS(int count)  {  int k=0;//记录这一层是否还有路  for(int i=0;i<n;i++)//来记录经过count层时，可以到达的格子  for(int j=0;j<m;j++)  if(num[i][j]==0)  {  if(num[i-1][j]==count-1&&i!=0)  {  num[i][j]=count;  k++;  }  if(num[i+1][j]==count-1&&i!=n-1)  {  num[i][j]=count;  k++;  }  if(num[i][j-1]==count-1&&j!=0)  {  num[i][j]=count;  k++;  }  if(num[i][j+1]==count-1&&j!=m-1)  {  num[i][j]=count;  k++;  }  }  if(num[xb][yb]!=0||k==0)//找到终点，或这一层没有路了  return ;  BFS(count+1);//递归下一层  }  int main()  {  cin>>n>>m;  cin>>xa>>ya>>xb>>yb;//初始化  for(int i=0;i<n;i++)  for(int j=0;j<m;j++)  cin>>num[i][j];  num[xa][ya]=1;//定义起点为1  BFS(2);//从第二层开始  resn=num[xb][yb];  for(int i=0;i<n;i++)//由于起点为1，所以应该全部减一  for(int j=0;j<m;j++)  if(num[i][j]>0)  num[i][j]--;  for(int i=0;i<n;i++)//将和终点在同一层上面的格子定义为0（终点的这一层只输出终点）  for(int j=0;j<m;j++)  if(num[i][j]>=resn-1&&resn!=0)  num[i][j]=0;  for(int i=0;i<n;i++)//进行输出  {  for(int j=0;j<m;j++)  {  if(i==xa&&j==ya)  cout<<" a";  else if(i==xb&&j==yb)  cout<<" b";  else cout<<setw(3)<<num[i][j];  }  cout<<endl;  }  //system("pause");  return 0;  }  **【实验结果及遇到的问题】** | | | |