洪水填充搜索

人员

柳力玮、韩昱辰、温郝冬、田心一、刘宸熙、赵书梵、李瑞涵、李知朔、纪博涵 到课, 苑钊、蒋叔璋、高健桓、初锦阳 线上

上周作业检查

上周作业链接: https://cppoj.kids123code.com/contest/1589

王向东老师周日三点半C++递归二叉树

刷新

#	用户名	姓名	编程分	时间	A	B	C	D	E	F
1	zhaoshufan	赵书梵	600	294	100	100	100	100	100	100
2	liuliwei	柳力玮	600	304	100	100	100	100	100	100
3	lizhishuo	李知朔	600	401	100	100	100	100	100	100
4	yuanzhao	苑钊	600	409	100	100	100	100	100	100
5	gaojianhuan	高健桓	600	411	100	100	100	100	100	100
6	tianxinyi	田心一	600	414	100	100	100	100	100	100
7	hanyuchen	韩昱辰	600	416	100	100	100	100	100	100
8	liruihan	李瑞涵	600	450	100	100	100	100	100	100
9	wenhaodong	温郝冬	500	362	100	100	100	100	100	
10	jiangshuzhang	蒋叔璋	400	346	100	100	100	100		
11	jibohan	纪博涵	400	351	100	100	100	100		
12	liuchenxi	刘宸熙	400	359	100	100	100	100	0	
13	chujinyang	初锦阳	200	258	100	100				

您正在共享屏幕 结束共享

本周作业

https://cppoj.kids123code.com/contest/1690 (课上讲了 A ~ E 题, 课后作业是 F G 题)

课堂表现

今天讲的洪水填充, 搜联通块的知识, 这个知识点不算难, 同学们课上整体吸收的都还不错

不过暴露出的问题是同学们代码小错误不断, 各种变量加减、数组定义等小错误层出不穷, 这一点需要同学们以后做题的时候更加认真来避免。

课堂内容

新二叉树 (上周作业)

把字符映射成对应的整数, 然后就跟二叉树模板题一样了

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

using namespace std;
```

```
const int maxn = 200 + 5;
int tr[maxn][2];

int get_int(char x) {
    if (x == '*') return 0;
    return x;
}

void dfs1(int u) {
    if (u == 0) return;
    cout << char(u);
    dfs1(tr[u][0]);
    dfs1(tr[u][1]);
}

int main()
{
    int n; cin >> n;
    int root;
    for (int i = 1; i <= n; i++) {
        string s; cin >> s;
        int a = get_int(s[0]), b = get_int(s[1]), c = get_int(s[2]);
        tr[a][0] = b, tr[a][1] = c;
        if (i == 1) root = a;
    }
    dfs1(root);
    return 0;
}
```

入门

从 '@' 这个符号开始, 往上下左右进行搜索, 把能搜到的点都标记下来, 最后看有几个点被标记了

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

using namespace std;

const int maxn = 20 + 5;
char s[maxn][maxn];
bool st[maxn][maxn];
struct node {
  int x, y;
};
int dx[] = {-1, 1, 0, 0};
int dy[] = {0, 0, -1, 1};
int n, m;

void dfs(int x, int y) {
  if (st[x][y]) return;
  st[x][y] = true;
```

```
  for (int i = 0; i < 4; ++i) {
    int nx = x + dx[i], ny = y + dy[i];
    if (nx>=1 && nx<=n && ny>=1 && ny<=m && s[nx][ny]=='.') dfs(nx, ny);
  }
}

int main()
{
  cin >> m >> n;
  for (int i = 1; i <= n; ++i) cin >> (s[i]+1);

  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= m; ++j) {
      if (s[i][j] == '@') dfs(i, j);
    }
  }

  int res = 0;
  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= m; ++j) {
      res += st[i][j];
    }
  }
  cout << res << endl;
  return 0;
}
```

求细胞数量

遍历每一个位置上, 如果这个位置上是个数字, 并且没有被标记, 那么这就是一个联通块, 并从这个点开始做搜索找全这个联通块即可

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

using namespace std;

const int maxn = 100 + 5;
char s[maxn][maxn];
bool st[maxn][maxn];
int n, m;
int dx[] = {-1, 1, 0, 0}, dy[] = {0, 0, -1, 1};

void dfs(int x, int y) {
  st[x][y] = true;
  for (int i = 0; i < 4; ++i) {
    int nx = x+dx[i], ny = y+dy[i];
    if (nx>=1 && nx<=n && ny>=1 && ny<=m && s[nx][ny]!='0' && !st[nx][ny]) dfs(nx,
ny);
  }
}
```

```
int main()
{
  cin >> n >> m;
  for (int i = 1; i <= n; ++i) cin >> (s[i]+1);

  int res = 0;
  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= m; ++j) {
      if (s[i][j]!='0' && !st[i][j]) {
        ++res; dfs(i,j);
      }
    }
  }
  cout << res << endl;
  return 0;
}
```

圣诞夜的极光

整体思路跟上一道题一致, 就是判断联通的逻辑改一下即可

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

using namespace std;

const int maxn = 100 + 5;
char s[maxn][maxn];
bool st[maxn][maxn];

void dfs(int x, int y, int n, int m) {
  st[x][y] = true;
  for (int i = x-2; i <= x+2; ++i) {
    for (int j = y-2; j <= y+2; ++j) {
      int d = abs(x-i) + abs(y-j);
      if (d > 2) continue;
      if (i>=1 && i<=n && j>=1 && j<=m && s[i][j]=='#' && !st[i][j]) dfs(i, j, n,
m);
    }
  }
}

int main()
{
  int n, m; cin >> n >> m;
  for (int i = 1; i <= n; ++i) cin >> (s[i]+1);

  int res = 0;
  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= m; ++j) {
      if (s[i][j]=='#' && !st[i][j]) {
        dfs(i, j, n, m); ++res;
```

```
      }
    }
  }
  cout << res << endl;
  return 0;
}
```

填涂颜色

所有跟外围的 0 连在一起的 0 都是外围的, 不跟外围的 0 连在一起的就是内部的

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

using namespace std;

const int maxn = 30 + 5;
int w[maxn][maxn];
bool st[maxn][maxn];
int n;
int dx[] = {-1, 1, 0, 0}, dy[] = {0, 0, -1, 1};

void dfs(int x, int y) {
  st[x][y] = true;
  for (int i = 0; i < 4; ++i) {
    int nx = x+dx[i], ny = y+dy[i];
    if (nx>=1 && nx<=n && ny>=1 && ny<=n && w[nx][ny]==0 && !st[nx][ny]) dfs(nx,
ny);
  }
}

int main()
{
  cin >> n;
  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= n; ++j) cin >> w[i][j];
  }

  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= n; ++j) {
      if ((i==1||i==n||j==1||j==n) && !w[i][j]) dfs(i,j);
    }
  }

  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= n; ++j) {
      if (w[i][j] == 1) cout << w[i][j] << " ";
      else {
        if (st[i][j]) cout << 0 << " ";
        else cout << 2 << " ";
      }
    }
```

```
    cout << endl;
  }
  return 0;
}
```

[USACO10OCT] Lake Counting S

求联通块数量

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

using namespace std;

const int maxn = 100 + 5;
char s[maxn][maxn];
bool st[maxn][maxn];

int dx[] = {-1, -1, -1, 0, 0, 1, 1, 1};
int dy[] = {-1, 0, 1, -1, 1, -1, 0, 1};

void dfs(int x, int y, int n, int m) {
  st[x][y] = true;
  for (int i = 0; i < 8; ++i) {
    int nx = x+dx[i], ny = y+dy[i];
    if (nx>=1 && nx<=n && ny>=1 && ny<=m && s[nx][ny]=='W' && !st[nx][ny]) dfs(nx,
ny, n, m);
  }
}

int main()
{
  int n, m; cin >> n >> m;
  for (int i = 1; i <= n; ++i) cin >> (s[i]+1);

  int cnt = 0;
  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= m; ++j) {
      if (s[i][j]=='.' || st[i][j]) continue;
      dfs(i, j, n, m);
      ++cnt;
    }
  }
  cout << cnt << endl;
  return 0;
}
```