ACM 算法与微应用实验室 2021 年 11 月月赛题解

2021 年 12 月 1 日

题目概览

题目编号	题目名称	命题人	做法
A	克隆干员	AgOH	模拟
B	pro2	AgOH	模拟
C	三斜求积术	AgOH	模拟
D	子树大小	AgOH	dfs
E	pro5	AgOH	模拟
F	pro6	AgOH	模拟

A. 克隆干员

做法

模拟过程。比较简便的做法如下：

首先我们先将干员的四种不同朝向时的状况分别保存于数组中（也就是先把干员旋转好），然后开出一个初始值为 0 的数组用来表示战场，最后每输入一个点后就向这个战场数组中涂色（填 1）即可。

标程

```
#include <iostream>
#include <string>
using namespace std;
string s[10];
int fw[5][10][10],a[15][15]; // fw[i] i从1到4分别保存干员朝向上下左右时的攻击范围；a为战场
int main()
{
    for(int i=0;i<7;i++) cin>>s[i]; // 用string简单读入
    for(int i=1;i<=7;i++) for(int j=1;j<=7;j++) fw[1][i][j]=s[i-1][j-1]-'0'; // 朝上
    for(int i=1;i<=7;i++) for(int j=1;j<=7;j++) fw[2][i][j]=fw[1][8-i][j]; // 朝下
    for(int i=1;i<=7;i++) for(int j=1;j<=7;j++) fw[3][i][j]=fw[1][j][i]; // 朝左
    for(int i=1;i<=7;i++) for(int j=1;j<=7;j++) fw[4][i][j]=fw[1][8-j][i]; // 朝右
    int n;
    cin>>n;
    for(int t=0;t<n;t++)
    {
        int x,y,f;
        cin>>x>>y>>f;
        for(int i=x-3;i<=x+3;i++) // 因为刚才已经旋转好了所以直接涂色即可
            for(int j=y-3;j<=y+3;j++)
                if(!(i<1||i>10||j<1||j>10)) // 如果没涂到战场范围之外的话
                    a[i][j]|=fw[f][i+4-x][j+4-y]; // 就涂色即可（若之前是0我们要把它变成1，若之前是1则还是1，用或即可）
    }
    for(int i=1;i<=10;i++)
    {
        for(int j=1;j<=10;j++)
            cout<<a[i][j];
        cout<<endl;
    }
    return 0;
}
```

B. pro2

做法

标程

C. 三斜求积术

做法

签到题，按照题目说明中给出的海伦公式进行模拟即可。

标程

```cpp
#include <cstdio>
#include <cmath>
int main()
{
    int t;
    scanf("%d", &t);
    while(t--)
    {
        int a,b,c;
        scanf("%d%d%d",&a,&b,&c);
        double p = (a+b+c)/2.0;
        double s = sqrt(p*(p-a)*(p-b)*(p-c));
        printf("%.2f\n", s);
    }
    return 0;
}
```

D. 子树大小

做法

求各子树大小是树上的经典基操，一遍 dfs 即可解决。

标程

```
#include <iostream>
using namespace std;
const int maxn = 1e4+5;
struct E { int to,next; } Edge[maxn<<1]; // 链式前向星，因为是树所以开2倍maxn大小即可
int tot,Head[maxn];
inline void AddEdge(int u,int v) { Edge[tot]={v,Head[u]}; Head[u]=tot++; }
int siz[maxn];
void dfs(int u,int f)
{
    siz[u]=1;  // 每个结点的大小等于自己的所有子树的大小之和加1（因为还包括结点自己）
    for(int i=Head[u];~i;i=Edge[i].next)
    {
        int v = Edge[i].to;
        if(v==f) continue; // 防止走回父亲
        dfs(v,u);  // 计算v子树大小
        siz[u]+=siz[v];   // 在u子树大小中加上v子树大小
    }
}
#include <cstring>
int main()
{
    memset(Head,-1,sizeof(Head)); // 链式前向星-1写法，将Head数组初始化为-1
    int n; cin>>n;
    for(int i=1,u,v;i<n;i++)     // 读入数据，注意正反加两条边
        cin>>u>>v, AddEdge(u,v), AddEdge(v,u);
    for(int rt=1;rt<=n;rt++)
    {
        dfs(rt, 0);
        for(int i=1;i<=n;i++)
            cout<<siz[i]<<' ';
        cout<<endl;
    }
    return 0;
}
```

E. pro5

做法

标程

F. pro6

做法

标程