2024暑假C23toB大黄鸭争夺赛题解

广州市执信中学 信息学奥赛教练组

2024.7.20

A 爬呀爬（普及-）

- 题意

蜗牛爬井问题：一只蜗牛要爬上 v 米高的树桩，它每天白天向上爬 a 米，晚上向下掉 b 米。几天之后它可以爬到树桩顶？（输入为整数）

数据范围：对于 100% 的数据，保证 $1 \leq b < a \leq v \leq 1 \times 10^9$。

A 爬呀爬

- 题解

$$ans = \left\lceil \frac{v-a}{a-b} \right\rceil + 1$$

参考代码：C23liupeilin

```
#include<bits/stdc++.h>
using namespace std;
int main() {
    double a, b, c;
    cin >> a >> b >> c;
    cout << fixed << setprecision(0) << ceil((c - a) / (a - b)) + 1;
    return 0;
}
```

B 旋转字符串（入门）

- 题意

有两个单词 A, B，将 A 水平摆放，B 竖直摆放，两个单词重叠部分必须为同一字母，且这一字母须在 A, B 中第一次出现。按题目要求输出摆放结果。

数据范围：$0 < |A|, |B| \leq 1000$，且A, B只由大写字母构成。

B 旋转字符串

- 题解

枚举，找到第一个相同的字符，记录下标，按要求输出。

参考代码：C24wangzihan

```cpp
#include <bits/stdc++.h>
using namespace std;
string n, m;
int x, y, f;
int main() {
    cin >> n >> m;
    for (int i = 0; i < n.length(); i++) {
        for (int j = 0; j < m.length(); j++) {
            if (n[i] == m[j] && f != 1) {
                x = i;
                y = j;
                f = 1;
            }
        }
    }
    for (int i = 0; i < m.length(); i++) {
        for (int j = 0; j < n.length(); j++) {
            if (j == x) {
                cout << m[i];
            } else if (i == y) {
                cout << n[j];
            } else {
                cout << '.';
            }
        }
        cout << endl;
    }
    return 0;
}
```

C 小老鼠杰瑞（普及）

- 题意

奶酪高为h，奶酪中有n个球形空洞，空洞的半径为r，给出这些球心坐标(x, y, z)，问老鼠能否通过空洞，从奶酪下表面跑到上表面。

数据范围：

对于 20% 的数据，$n = 1$，$1 \le h$，$r \le 10^4$，坐标的绝对值不超过 10^4。

对于 40% 的数据，$1 \le n \le 8$，$1 \le h$，$r \le 10^4$，坐标的绝对值不超过 10^4。

对于 80% 的数据，$1 \le n \le 10^3$，$1 \le h, r \le 10^4$，坐标的绝对值不超过 10^4。

对于 100% 的数据，$1 \le n \le 1 \times 10^3$，$1 \le h, r \le 10^9$，$T \le 20$，坐标的绝对值不超过 10^9。

C 小老鼠杰瑞

- 题解

1. dfs搜索$O(n^2 \times T)$

参考代码：C23panweiming

```
signed main() {
    cin >> t;
    for(int ti = 0; ti < t; ti++) {
        cin >> n >> h >> r;
        f = 0;
        memset(xl, 0, sizeof xl);
        memset(yl, 0, sizeof yl);
        memset(zl, 0, sizeof zl);
        memset(vis, 0, sizeof vis);
        for(int i = 0; i < n; i++) {
            cin >> xl[i] >> yl[i] >> zl[i];
        }
        for(int i = 0; i < n; i++) {
            if(!vis[i] && zl[i] - r <= 0) {
                dfs(i);
                if(f == 1) {
                    break;
                }
            }
        }
        if(f == 0) {
            cout << "No\n";
        }
    }
    return 0;
}
```

```
#include<iostream>
#include<cmath>
#include<cstring>
#define int long long
using namespace std;
int t, f = 0;
int n, h, r;
int xl[1005];
int yl[1005];
int zl[1005];
bool vis[1005];
double dist(int ax, int ay, int az, int bx, int by, int bz) {
    return 1.0 * sqrt( (ax - bx) * (ax - bx) + (ay - by) * (ay - by) + (az - bz) * (az - bz) );
}
void dfs(int id) {
    if(zl[id] + r >= h && f == 0) {
        cout << "Yes\n";
        f = 1;
    }
    vis[id] = 1;
    for(int i = 0; i < n; i++) {
        if(!vis[i] && dist(xl[id], yl[id], zl[id], xl[i], yl[i], zl[i]) <= r * 2) {
            dfs(i);
            if(f == 1) {
                return;
            }
        }
    }
}
```

C 小老鼠杰瑞

- 题解

  2. bfs搜索$O(n^2 \times T)$

  参考代码： C23shiwei

```cpp
#include<bits/stdc++.h>
#define int long long
#define db double
using namespace std;
int t, n, h, r, vis[1010];
struct node {
    int x, y, z;
} a[1010];
db dis(db x1, db y1, db z1, db x2, db y2, db z2) {
    int dx = x1 - x2, dy = y1 - y2, dz = z1 - z2;
    return sqrt(dx * dx + dy * dy + dz * dz);
}
```

```cpp
signed main() {
    cin >> t;
    while(t--) {
        int ans = 0;
        cin >> n >> h >> r;
        memset(vis, 0, sizeof(vis));
        queue<node>q;
        for(int i = 1; i <= n; i++) {
            cin >> a[i].x >> a[i].y >> a[i].z;
            if(a[i].z <= r) {
                vis[i] = 1;
                q.push(a[i]);
            }
        }
        while(!q.empty()) {
            auto t = q.front();
            q.pop();
            if(t.z + r >= h) {
                ans = 1;
                break;
            }
            for(int i = 1; i <= n; i++)if(vis[i] == 0) {
                    if(dis(t.x, t.y, t.z, a[i].x, a[i].y, a[i].z) <= 2 * r) {
                        vis[i] = 1;
                        q.push(a[i]);
                    }
                }
        }
        if(ans)cout << "Yes\n";
        else cout << "No\n";
    }
}
```

C 小老鼠杰瑞

- 题解

  2. 并查集

  参考代码： G23xuewentao

D 剑三江湖

- 模拟题，按题意模拟即可。

参考代码：C23liupeilin

```
#include<bits/stdc++.h>
using namespace std;
string a[105];
int n,m,hp,st,de,fx,fy,sti,dei,hpi,q;
int main(){
    cin>>n>>m;
    for(int i=0;i<n;i++)cin>>a[i];
    cin>>hp>>st>>de>>fx>>fy>>sti>>dei>>q;
    while(q--){
        int op;
        cin>>op;
        if(op==1){cout<<hpi<<" "<<sti<<" "<<dei<<endl;continue;}
        int z;
        cin>>z;
        switch(z){
            case 1:fy--;break;
            case 2:fy++;break;
            case 3:fx--;break;
            case 4:fx++;break;
        }
        switch(a[fx-1][fy-1]){
            case 'R':hpi=max(hpi-10,0);break;
            case 'Q':sti+=5;break;
            case 'Y':dei+=5;break;
            case 'M':int n=ceil((double)hp/max(1,sti-de))*max(1,st-dei);hpi+=max(1,n);break;
        }
    }
    return 0;
}
```

E 数列排序（普及）

- 题意

  n个数的数量，移动数字位置，使得数列从小到大排序，问最小的移动数字之和。

  数据范围：

  对于 60% 的数据：$0 < n \leq 60$，$-10^7 \leq k_i \leq 10^7$

  对于 80% 的数据：$0 < n \leq 80$，$-10^7 \leq k_i \leq 10^7$

  对于 100% 的数据：$0 < n \leq 10^2$，$-10^7 \leq k_i \leq 10^7$

E 数列排序

- 题解

移动的数字之和最小，即留下的不下降序列之和最大。因此转化为求数列的最大不下降子序列ans，最后用数列之和sum减去最大不下降子序列ans即为答案。

```
#include<bits/stdc++.h>
using namespace std;
const int maxn = 1e3 + 5;
int n, a[maxn], dp[maxn];
int main() {
    int t;
    scanf("%d", &t);
    while(t--) {
        scanf("%d", &n);
        int sum = 0, ans = 0;
        for (int i = 1; i <= n; i++)
            scanf("%d", &a[i]), sum += a[i];
        for (int i = 1; i <= n; i++) {
            dp[i] = a[i];
            for (int j = i - 1; j >= 1; j--)
                if (a[i] >= a[j])
                    dp[i] = max(dp[i], dp[j] + a[i]);
            ans = max(ans, dp[i]);
        }
        printf("%d\n", sum - ans);
    }
    return 0;
}
```

F 数对（普及）

- 题意

给定n个正整数a_i，请你求出有多少个数对 (i,j) 满足 $1 \le i \le n$，$1 \le j \le n$，$i \ne j$ 且 a_i 是 a_j 的倍数。

数据范围：

对于 40% 的数据，$n \le 1000$。
对于 70% 的数据，$1 \le a_i \le 5 \times 10^3$。
对于 100% 的数据，$2 \le n \le 2 \times 10^5$，$1 \le a_i \le 5 \times 10^5$。

答案爆int，记得开long long。

F 数对

- 题解1

a_i 是 a_j 的倍数，则说明 a_j 是 a_i 的因数。

因此我们可以枚举求出每个数的因数，用桶存起来，即cnt数组，此时时间复杂度为 $O(\sqrt{a_i} \times n)$ 。接着枚举每个数，看该数所在的桶里面元素的个数，即是该数的倍数个数+1（因为该数也在桶内）。所以$ans \mathrel{+}= cnt[a[i]] - 1$。

```
#include<bits/stdc++.h>
typedef long long ll;
using namespace std;
const int maxn = 5e5 + 10;
int a[maxn], cnt[maxn];
int n, num1, num2;
ll ans = 0;
int main() {
    scanf("%d", &n);
    for(int i = 1; i <= n; i++) {
        scanf("%d", &a[i]);
        for(int j = 1; j * j <= a[i]; j++) {
            if(a[i] % j == 0) {
                num1 = j, num2 = a[i] / j;
                cnt[num1]++;
                if(num1 != num2)
                    cnt[num2]++;
            }
        }
    }
    for(int i = 1; i <= n; i++)
        ans = ans + cnt[a[i]] - 1;
    printf("%lld\n", ans);
    return 0;
}
```

F 数对

- 题解2

计数出现次数：首先遍历输入数组，记录每个数出现的次数。用一个数组 cnt 记录每个数出现的频次。

对于每一个数 i，我们需要考虑两个情况：

1. 相同的数：如果 $a_i = a_j$，那么数对数目是 $cnt[i] \times (cnt[i] - 1)$，因为选择两个不同的位置。

2. 不同的数：如果 $ai \neq aj$，那么 a_j是 i 的倍数，即 $a_j = k \times i$（其中 k 是一个正整数），我们只需统计倍数 $k \times i$的出现次数 $cnt[k \times i]$，并且计算数对数目为 $cnt[i] \times cnt[k \times i]$。

```
#include<bits/stdc++.h>
typedef long long ll;
using namespace std;
const int maxn = 5e5 + 10;
int cnt[maxn];
int n, x;
ll ans = 0;
int main() {
    scanf("%d", &n);
    for(int i = 1; i <= n; i++) {
        scanf("%d", &x);
        cnt[x]++;
    }
    for(int i = 1; i < maxn; i++) {
        ans += 1ll * cnt[i] * (cnt[i] - 1);
        for(int j = i + i; j < maxn; j += i)
            ans += 1ll * cnt[i] * cnt[j];
    }
    printf("%lld\n", ans);
    return 0;
}
```