

# 2021-7-27 比赛报告（牛客小白月赛36）

目录：

- 2021-7-27 比赛报告（牛客小白月赛36）
  - A题（好哥哥）
    - 题目描述
    - 题目分析
    - 解题思路
    - 代码实现
  - B题（最短串）
    - 题目描述
    - 题目分析
    - 解题思路
    - 代码实现
  - C题（杨辉三角）
    - 题目描述
    - 题目分析
    - 解题思路
    - 代码实现：
  - E题（皇城PK）
    - 题目描述
    - 题目分析
    - 解题思路
    - 代码实现
  - F题（象棋）
    - 题目描述
    - 题目分析
    - 解题思路
    - 代码实现
  - H题（卷王之王）
    - 题目描述
    - 题目分析
    - 解题思路
    - 代码实现
  - I题（四面楚歌）
    - 题目描述



## 解题思路

- 维护一个栈,建立一个最大层数maxn。
- 记录合法括号的对数（左括号的个数）
- n块钱总共可以经过n+1个括号，那么如果最大层数大于n+1，则最多经过n+1对不重复括号。否则，我们可以知道访问一个最大层数链之外的“支链”，再返回到最大层数链的消耗需要\*2，那么我们可以得到理论上最多经过括号数量为 $(n - \text{maxn} + 1)/2$ ;但是！可能整个序列的括号数都没有这么多，需要和总括号的对数取个最小值。保证不超出括号的总数。

## 代码实现

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  char s[100000010];
4  int n,m,maxn,c,ans;
5  stack<char>st;
6  int main(){
7      ios::sync_with_stdio(0);
8      cin>>n>>m>>s+1;
9      for(int i=1;i<=2*n;i++){
10         if(s[i]=='(')st.push('('),c++;
11         else st.pop();
12         if(st.empty())break;
13         maxn=max(maxn,(int)st.size());
14     }
15     if(m+1<maxn)ans=m+1;
16     else ans=min(c,maxn+(m-maxn+1)/2);
17     cout<<ans<<endl;
18     return 0;
19 }
```

## B题（最短串）

[题目链接](#)

## 题目描述

给定2个由小写字母和问号组成的字符串a与b，问号代表你想要的任何字符。  
请你找出最短的字符串s，要求s包含a和b两个字符串，你只需要输出s的长度即可。

## 题目分析

给定两个字符串，'?'可代替成任何字符，询问一个最短公共字符串包含这两个字符串。

## 解题思路

- 暴力，两层for循环寻找匹配位置，然后交换字符串a,b，重复一次输出最小长度。
- 也可以使用KMP算法

## 代码实现

暴力：

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  const int maxn=1000010;
4  char s[maxn],t[maxn];
5  int n,m,ans,f;
6  int fun(){
7      for(int i=0;i<n;i++){
8          f=1;
9          for(int j=0;j<m&& i+j<n;j++){
10             if(s[i+j]!='?' || t[j]!='?' || s[i+j]!=t[j])continue;
11             f=0;break;
12         }
13         if(f) ans=min(ans,max(n,i+m));
14     }
15     return ans;
16 }
17 int main(){
18     cin>>s>>t;
19     n=strlen(s),m=strlen(t);
20     ans=m+n;
21     fun();
22     swap(s,t),swap(n,m);
23     fun();
24     cout<<ans<<endl;
25     return 0;
26 }
```

KMP算法：

```

1  #include<iostream>
2  #include<vector>
3  #include<algorithm>
4  #include<string.h>
5  using namespace std;
6  const int N=100010,M=1000010;
7  char p[N],s[M];
8  int n,m;
9  int ne[N];
10 vector<int>c;
11 int contain(char p[],char s[]){
12     if(!n) return n;
13     for (int i = 2, j = 0; i <=n; i ++ ){
14         while (j && p[i] != p[j + 1]&&p[i]!='?'&&p[j + 1]!='?') j = ne[j];
15         if (p[i] == p[j + 1]||p[i]=='?'||p[j + 1]=='?') j ++ ;
16         ne[i] = j;
17     }
18     for (int i = 1, j = 0; i <=m; i ++ ){
19         while (j && s[i] != p[j + 1]&&s[i]!='?'&&p[j + 1]!='?') j = ne[j];
20         if (s[i] == p[j + 1]||s[i]=='?'||p[j + 1]=='?') j ++ ;
21         if(j==n)
22             return n;
23     }
24     return 0;
25 }
26 int main(){
27     scanf("%s%s",p+1,s+1);
28     n=strlen(p+1);
29     m=strlen(s+1);
30     int len=0;
31     if(n>m){
32         swap(n,m);
33         len=contain(s,p);
34         swap(n,m);
35     }else len=contain(p,s);
36
37     if(len){
38         printf("%d",m+n-len);return 0;}
39     int ans=n+m;
40     int flag,i,j;
41     for(i=min(n,m);i>=1;i--){
42         flag=0;
43         for(j=1;j<=i;j++)if(p[j]!=s[m-i+j]&&p[j]!='?'&&s[m-i+j]!='?')break;
44         if(j>i)flag=1;
45         for(j=1;j<=i;j++)if(p[n-i+j]!=s[j]&&p[n-i+j]!='?'&&s[j]!='?')break;
46         if(j>i)flag=1;
47         if(flag)ans=min(ans,n+m-i);
48     }
49     printf("%d\n",ans);
50     return 0;
51 }

```

## C题（杨辉三角）

[题目链接](#)

### 题目描述

小F对杨辉三角颇有研究，他把杨辉三角第 $n$ 行的数提出来，从左到右分别为 $a[0], a[1], \dots, a[n-1]$ 。

现在他想知道 $\sum_{i=0}^{n-1} i^2 \times a[i]$ 的值是多少，答案对99824353取模。

### 题目分析

- 杨辉三角第 $n$ 行的数分别为 $a[0], a[1], \dots, a[n-1]$ 。
- 求 $\sum_{i=0}^{n-1} i^2 * a[i]$ , 对998244353取模。

### 解题思路

推导过程：

多项式的展开公式： $(x+1)^n = \sum_{i=1}^n C_n^i x^i$ ;

两边同时求导得： $n(x+1)^{n-1} = \sum_{i=1}^n i C_n^i x^{i-1}$ ;

两边同时乘 $x$ 得： $nx(x+1)^{n-1} = \sum_{i=1}^n i C_n^i x^i$ ;

再次求导数得： $n(x+1)^{n-1} + n(n-1)x(x+1)^{n-2} = \sum_{i=1}^n i^2 C_n^i x^{i-1}$ ;

令 $x=1$ 得： $n2^{n-1} + n(n-1)2^{n-2} = \sum_{i=1}^n i^2 C_n^i$ ;

整理得： $n(n+1)2^{n-2} = \sum_{i=1}^n i^2 C_n^i$ ;

### 代码实现：

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  ll i,j,n,c,r=1;
5  const ll l=99824353;
6  ll fun(ll a,ll b){
7      while(b!=0){if(b%2) r=r*a%l;a=a*a%l;b/=2;}
8      return r%l;
9  }
10 int main(){
11     ios::sync_with_stdio(0);
12     cin>>n;
13     if(n==2){cout<<1<<endl;return 0;}n--;
14     cout<<n%l*((n+1)%l)%l*(fun(2,n-2)%l)%l<<endl;
15     return 0;
16 }

```

## E题（皇城PK）

[题目链接](#)

### 题目描述

有n名选手会进行m次比赛，每次比赛不会出现平局的情况，只会有一个胜者。在每次比赛完成之后，我们视胜者选手的实力比败者选手的实力强，如果出现选手AA打败选手BB，选手B打败选手C，选手C打败选手A，则视为他们的实力全部相同。

若该赛季最终冠军是属于实力最强者，请问依照现在已有的比赛结果，最多有多少个选手可能获得冠军（如果已知两个人的实力一样强，那么他们两个人都不能获得冠军）。

### 题目分析

n名选手进行m次比赛，一次没输过的人数，因为如果一个人失败了，那么他就不可能成为冠军(总有人比他实力更强)

### 解题思路

找出有多少人失败了，就能知道有多少名选手可能成为冠军。

### 代码实现

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  int a[1000010];
4  int n,m,m1,m2,c;
5  int main(){
6      cin>>n>>m;
7      for(int i=0;i<m;i++) cin>>m1>>m2,a[m2]=1;
8      for(int i=1;i<=n;i++) if(a[i]==1) c++;
9      cout<<n-c<<endl;
10     return 0;
11 }

```

## F题（象棋）

[题目链接](#)

### 题目描述

在中国象棋中正所谓新手玩车，熟手玩炮，老手玩马，由此可见象棋中炮的地位还是比较高的。给定一个 $n \times m$ 的棋盘，全部摆满炮，我们视所有炮都不属于同一阵营，他们之间可以相互攻击但不能不不进行攻击直接移动。请问经历若干次攻击，直到不能攻击后，最少能剩余多少个炮。

### 题目分析

给定一个 $n \times m$ 的棋盘，全部摆炮。按照象棋中的炮的规则，他们之间互相攻击但不能不不进行攻击直接移动。请问经历若干次攻击，直到不能攻击后，最少能剩余多少个炮。

### 解题思路

一行或一列数量 $>2$ 全是炮时，可以互相打击直到存活两个跑车，如果我们保证每行相同操作，最后会剩下两列炮，同理两列也能变成每列两个，最后存活 $2 \times 2$ 个炮。当一行或一列的数量 $<2$ 时需要特判一下。

### 代码实现



```

1  #include<bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  ll a,b,s;
5  int main(){
6      ios::sync_with_stdio(0);
7      cin.tie(0);
8      cin>>s;
9      for(int i=0;i<s;i++){
10         cin>>a>>b;
11         if(a==0||b==0) {cout<<0<<endl;continue;}
12         else if(a==1&&b==1) {cout<<1<<endl;continue;}
13         else if(a==1||b==1) {cout<<2<<endl;continue;}
14         else {cout<<4<<endl;continue;}
15     }
16     return 0;
17 }

```

## H题（卷王之王）

[题目链接](#)

### 题目描述

牛卷风是养蛊大学著名的小镇做题家，每天早上6点半起床，凌晨2点半睡觉，除了一日三餐，其他时间均用来学习，因此考试从未低于90分，人送外号“养蛊大学不眠传说”。

你从四处打听到，牛卷风如此之强的原因在于他有一套练习计算能力的秘诀，该秘诀如下：首先给出n个数字，第i个数字为a[i]。接下来进行m次操作，每次操作给出一个数字x，练习者在心中将所有值小于等于x的数字都加上x。当进行完这m次操作后，练习者再按顺序给出这n个数字。

话不多说，你立马着手练习。首先你让朋友给出一开始的n个数字和m次操作的x，请你给出进行完m次操作后的n个数字。

### 题目分析

给定n个数字，每次给定一个数字x将小于等于x的数加上x，最后输出m次操作后的n个数字。

### 解题思路

- 1.建立一个结构体类型的数据结构：{int int }：用于存放n个数和下标
- 2.建立一个优先队列按照n个数从小到大排列
- 3.每次取出小于x的数，放到容器t中

4.加上x后再次放入优先队列中

5.最后在优先队列中再次取出n个数及其下标，按下标输出

## 代码实现

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  typedef pair<int,int>p; //建立一个数据类型: 结构体{int, int}
4  const int N=100010;
5  int n,m,x,a[N];
6  priority_queue<p,vector<p>,greater<p>>q; //优先队列p:数据类型;vector<p>:数据格式;greater<p>:小
7  vector<p>t; //创建数组容器, 用于存放中间操作的数
8  int main(){
9      cin>>n>>m;
10     for(int i=1;i<=n;i++)cin>>x,q.push({x,i}); //将n个数和数的下标存入优先队列中
11     while(m--){
12         cin>>x;
13         if(!x)continue; //x为0, 跳过本次循环
14         t.clear(); //清空容器
15         while(q.size() && q.top().first<=x)t.push_back(q.top()),q.pop(); //q不为空且优先队列中的
16         for(auto&it:t)it.first+=x,q.push(it); //将取出的数都加上x, 再次存入优先队列中
17     }
18     while(q.size()) a[q.top().second]=q.top().first,q.pop(); //再将数从优先队列中取出, 存入到a
19     for(int i=1;i<=n;i++)cout<<a[i]<<" "; //输出数组
20     return 0;
21 }
```

## I题 (四面楚歌)

[题目链接](#)

### 题目描述

在游戏中，因为一次错误的决断，你的士兵被敌方实行围剿。为了挽回人员损失，你不得不开启金手指暂停敌方士兵的移动，从而尽量让自己的士兵能成功突围。

已知地图是一块 $n \times m$ 的区域，每块格子有以下几种类型：

.：表示此处为一块空地。

1：表示此处有敌方士兵，不许通过。因为开启了金手指，所以敌方士兵不会移动。

0：表示此处有我方士兵。

现规定我方士兵只能进行上/下/左/右四个方向的移动，只要某个士兵移动出了地图边界，那么就算该士兵突围成功。请问能有多少士兵成功突围。

### 题目分析

给定 $n \times m$ 的区域，1代表‘墙’，0代表人，‘.’代表空地，询问多少个0能移动到边界

## 解题思路

- 1.创建一个字符数组和上下左右移动的数组
- 2.从第边界开始dfs
- 3.如果遇到边界值0，计数器就加1；将边界内不是1的全部赋值为1，遇到1就绕道
- 4.输出计数器的值

## 代码实现

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  char ma[1010][1010];
4  int dx[4]={0,0,1,-1},dy[4]={1,-1,0,0};
5  int n,m,c,xx,yy;
6  void dfs(int x,int y){
7      for(int i=0;i<=3;i++){
8          xx=x+dx[i],yy=y+dy[i];
9          if(xx>=0&&xx<=n+1&&yy>=0&&yy<=m+1&&ma[xx][yy]!='1'){
10             if(ma[xx][yy]=='0')c++;
11             ma[xx][yy]='1';
12             dfs(xx,yy);
13         }
14     }
15 }
16 int main(){
17     cin>>n>>m;
18     for(int i=1;i<=n;i++)for(int j=1;j<=m;j++)cin>>ma[i][j];
19     dfs(0,0);
20     cout<<c<<endl;
21     return 0;
22 }
```