# A

**枚举删除的台阶就行了。**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**zzblack \*\***

**2016-04-14 \*\***

**Orz \*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**#include <set>**

**#include <map>**

**#include <queue>**

**#include <stack>**

**#include <cmath>**

**#include <string>**

**#include <cctype>**

**#include <cstdio>**

**#include <vector>**

**#include <cstdlib>**

**#include <cstring>**

**#include <iomanip>**

**#include <sstream>**

**#include <iostream>**

**#include <algorithm>**

**using namespace std;**

**#define ls id<<1,l,mid**

**#define rs id<<1|1,mid+1,r**

**#define OFF(x) memset(x,-1,sizeof x)**

**#define CLR(x) memset(x,0,sizeof x)**

**#define MEM(x) memset(x,0x3f,sizeof x)**

**typedef long long ll ;**

**typedef pair<int,int> pii ;**

**const int maxn = 1e6 + 50 ;**

**const int maxm = 1e6 + 50;**

**const double eps = 1e-10;**

**const int max\_index = 62;**

**const int inf = 0x3f3f3f3f ;**

**const int MOD = 1e9+7 ;**

**inline int read(){**

**char c = getchar();**

**while (!isdigit(c)) c = getchar();**

**int x = 0;**

**while (isdigit(c)) {**

**x = x \* 10 + c - '0';**

**c = getchar();**

**}**

**return x;**

**}**

**int a[maxn];**

**int main () {**

**#ifdef LOCAL**

**// freopen("6.in","r",stdin);**

**// freopen("6.out","w",stdout);**

**#endif**

**int T = read(), cas = 1;**

**while (T--) {**

**int n = read();**

**int ans = inf, dif = 0;**

**for (int i = 1; i <= n; i++) a[i] = read();**

**for (int i = 2; i <= n; i++) dif = max(dif, a[i] - a[i - 1]);**

**for (int i = 2; i < n; i++) ans = min(ans, max(dif, a[i + 1] - a[i - 1]));**

**printf("Case %d: %d\n", cas++, ans);**

**// cout << ans << "\n";**

**}**

**return 0;**

**}**

**时间复杂度 O(n)**

# B

**输出n在二进制下表示的1的个数就行了，注意负数**

**此程序提供两种写法**

**时间复杂度 O(logn)**

**//**

**// Created by Running Photon on 2016-04-13**

**// Copyright (c) 2015 Running Photon. All rights reserved.**

**//**

**#include <algorithm>**

**#include <cctype>**

**#include <cmath>**

**#include <cstdio>**

**#include <cstdlib>**

**#include <cstring>**

**#include <iomanip>**

**#include <iostream>**

**#include <map>**

**#include <queue>**

**#include <string>**

**#include <sstream>**

**#include <set>**

**#include <vector>**

**#include <stack>**

**#define ALL(x) x.begin(), x.end()**

**#define INS(x) inserter(x, x,begin())**

**#define ll long long**

**#define CLR(x) memset(x, 0, sizeof x)**

**using namespace std;**

**const int inf = 0x3f3f3f3f;**

**const int MOD = 1e9 + 7;**

**const int maxn = 1e6 + 10;**

**const int maxv = 1e3 + 10;**

**const double eps = 1e-9;**

**int main() {**

**#ifdef LOCAL**

**// freopen("1.in", "r", stdin);**

**// freopen("1.out","w",stdout);**

**#endif**

**// ios\_base::sync\_with\_stdio(0);**

**int n, T;**

**cin >> T;**

**// while(T--) {cin >> n; cout << \_\_builtin\_popcount(n) << endl;} 写法1**

**while(T--) {**

**cin >> n;**

**int res = 0;**

**for(int i = 0; i < 32; i++) {**

**if((1 << i) & n) res++;**

**}**

**cout << res << endl;**

**}**

**return 0;**

**}**

# C

**拓扑排序，如果不存在拓扑序，说明一定是记错了**

**时间复杂度 O(n)**

**//**

**// Created by CQU\_CST\_WuErli**

**// Copyright (c) 2016 CQU\_CST\_WuErli. All rights reserved.**

**//**

**#include <iostream>**

**#include <cstring>**

**#include <cstdio>**

**#include <cstdlib>**

**#include <cctype>**

**#include <cmath>**

**#include <string>**

**#include <vector>**

**#include <map>**

**#include <queue>**

**#include <stack>**

**#include <set>**

**#include <algorithm>**

**#include <sstream>**

**#define CLR(x) memset(x,0,sizeof(x))**

**#define OFF(x) memset(x,-1,sizeof(x))**

**#define MEM(x,a) memset((x),(a),sizeof(x))**

**#define BUG cout << "I am here" << endl**

**#define lookln(x) cout << #x << "=" << x << endl**

**#define SI(a) scanf("%d",&a)**

**#define SII(a,b) scanf("%d%d",&a,&b)**

**#define SIII(a,b,c) scanf("%d%d%d",&a,&b,&c)**

**#define rep(flag,start,end) for(int flag=start;flag<=end;flag++)**

**#define Rep(flag,start,end) for(int flag=start;flag>=end;flag--)**

**const int INF\_INT=0x3f3f3f3f;**

**const long long INF\_LL=0x7f7f7f7f;**

**const int MOD=1e9+7;**

**const double eps=1e-10;**

**const double pi=acos(-1);**

**typedef long long ll;**

**using namespace std;**

**const int N=100000+100;**

**vector<int> G[N];**

**int deg[N];**

**int n,m;**

**int vis[N];**

**bool solve() {**

**queue<int> q;**

**CLR(vis);**

**int cnt=n;**

**for (int i=1;i<=n;i++) {**

**if (!deg[i]) {**

**q.push(i);**

**vis[i]=1;**

**cnt--;**

**}**

**}**

**while(q.size()) {**

**int x=q.front();q.pop();**

**for (int i=0;i<G[x].size();i++) {**

**int v=G[x][i];**

**if (vis[v]) continue;**

**deg[v]--;**

**if (deg[v]==0) {**

**q.push(v);**

**vis[v]=1;**

**cnt--;**

**}**

**}**

**}**

**return cnt!=0;**

**}**

**int main() {**

**#ifdef LOCAL**

**freopen("4.in","r",stdin);**

**freopen("4.out","w",stdout);**

**#endif**

**int t;SI(t);**

**while(t--) {**

**SII(n, m);**

**for (int i=1;i<=n;i++) G[i].clear();**

**CLR(deg);**

**for (int i=1;i<=m;i++) {**

**int u,v;**

**SII(u,v);**

**G[v].push\_back(u);**

**deg[u]++;**

**}**

**if (solve()) puts("Yes");**

**else puts("No");**

**}**

**return 0;**

**}**

# D

**类似于括号的匹配问题，只不过变成了acm的匹配问题，可以想想开两个stack模拟一下就行了**

**时间复杂度 O(n)**

**//**

**// Created by Running Photon on 2016-04-14**

**// Copyright (c) 2015 Running Photon. All rights reserved.**

**//**

**#include <algorithm>**

**#include <cctype>**

**#include <cmath>**

**#include <cstdio>**

**#include <cstdlib>**

**#include <cstring>**

**#include <iomanip>**

**#include <iostream>**

**#include <map>**

**#include <queue>**

**#include <string>**

**#include <sstream>**

**#include <set>**

**#include <vector>**

**#include <stack>**

**#define ALL(x) x.begin(), x.end()**

**#define INS(x) inserter(x, x,begin())**

**#define ll long long**

**#define CLR(x) memset(x, 0, sizeof x)**

**using namespace std;**

**const int inf = 0x3f3f3f3f;**

**const int MOD = 1e9 + 7;**

**const int maxn = 1e6 + 10;**

**const int maxv = 1e3 + 10;**

**const double eps = 1e-9;**

**int main() {**

**#ifdef LOCAL**

**// freopen("in.txt", "r", stdin);**

**// freopen("out.txt","w",stdout);**

**#endif**

**// ios\_base::sync\_with\_stdio(0);**

**int T;**

**cin >> T;**

**while(T--) {**

**string s;**

**cin >> s;**

**int t1 = 0;**

**int t2 = 0;**

**int ans = 0;**

**for(int i = 0; i < s.size(); i++) {**

**if(s[i] == 'a' || s[i] == 'A') t1++;**

**if((s[i] == 'c' || s[i] == 'C') && t1) {t1--; t2++;}**

**if((s[i] == 'm' || s[i] == 'M') && t2) {t2--; ans++;}**

**}**

**cout << ans << endl;**

**}**

**return 0;**

**}**

# E

**要求寻找到一个最长的序列，使得连着的两个数不互素，暴力的方式会tle，可以把每个数唯一分解，然后用动态规划的方法，建立状态方程dp[i][j]表示处理了前i个数，且结尾的数一定包含素因子j的最大长度，**

**设以第i个数结尾的最大长度为T，那么有**

**T = max{dp[i-1][t]}, 其中 t | T且，t是素数.**

**然后dp[i][t] = T, t | T.**

**发现i每次用i-1更新，阔以去掉第一维，减少空间消耗**

**标程中mp[i]代表i的最大素因子**

**时间复杂度 O(nlogn)**

**#include <cstdio>**

**#include <cstdlib>**

**#include <iostream>**

**#include <algorithm>**

**#include <cstring>**

**#include <climits>**

**#include <string>**

**#include <vector>**

**#include <cmath>**

**#include <stack>**

**#include <queue>**

**#include <set>**

**#include <map>**

**#include <sstream>**

**#include <cctype>**

**using namespace std;**

**typedef long long ll;**

**typedef unsigned long long ull;**

**typedef pair<int ,int> pii;**

**#define MEM(a,b) memset(a,b,sizeof a)**

**#define CLR(a) memset(a,0,sizeof a);**

**#define pi acos(-1.0)**

**#define maxn 40000**

**#define maxv 100005**

**const int inf = 0x3f3f3f3f;**

**const int MOD = 1e9 + 7;**

**int mp[maxv];**

**int dp[maxv];**

**int a[maxv];**

**int main()**

**{**

**// freopen("1.in", "r", stdin);**

**// freopen("1.out", "w", stdout);**

**for(int i=2;i<=1e5;i++){**

**if(!mp[i]){**

**for(int j=i;j<=1e5;j+=i){**

**mp[j] = i;**

**}**

**}**

**}**

**int n;**

**int T;**

**cin >> T;**

**while(T--){**

**cin >> n;**

**MEM(dp,0);**

**for(int i=1;i<=n;i++)cin >> a[i];**

**int nmax = 0;**

**for(int i=1;i<=n;i++){**

**int x = 0;**

**for(int j=a[i];j>1;j/=mp[j]){**

**x = max(x,dp[mp[j]]);**

**}**

**for(int j=a[i];j>1;j/=mp[j]){**

**dp[mp[j]] = max(dp[mp[j]],x+1);**

**}**

**nmax = max(x+1,nmax);**

**}**

**printf("%d\n",nmax);**

**}**

**return 0;**

**}**

# F

**一道比较机智题目，模拟后会发现，每个人的钱的数量，加上总人数减去当前位置是恒定不变的，并且总的钱的数量是不变的**

**那么对每个人我们用一个函数f(i)表示他的独立性，且 f(i) = money + n – i**

**如果出现相同的f,说明是无穷交换队列,时间复杂度 O(n)**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**zzblack \*\***

**2016-04-14 \*\***

**Orz \*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**#include <set>**

**#include <map>**

**#include <queue>**

**#include <stack>**

**#include <cmath>**

**#include <string>**

**#include <cctype>**

**#include <cstdio>**

**#include <vector>**

**#include <cstdlib>**

**#include <cstring>**

**#include <iomanip>**

**#include <sstream>**

**#include <iostream>**

**#include <algorithm>**

**using namespace std;**

**#define ls id<<1,l,mid**

**#define rs id<<1|1,mid+1,r**

**#define OFF(x) memset(x,-1,sizeof x)**

**#define CLR(x) memset(x,0,sizeof x)**

**#define MEM(x) memset(x,0x3f,sizeof x)**

**typedef long long ll ;**

**typedef pair<int,int> pii ;**

**const int maxn = 1e5 + 50 ;**

**const int maxm = 1e6 + 50;**

**const double eps = 1e-10;**

**const int max\_index = 62;**

**const int inf = 0x3f3f3f3f ;**

**const int MOD = 1e9+7 ;**

**inline int read(){**

**char c = getchar();**

**while (!isdigit(c)) c = getchar();**

**int x = 0;**

**while (isdigit(c)) {**

**x = x \* 10 + c - '0';**

**c = getchar();**

**}**

**return x;**

**}**

**int a[maxn], Hash[2000005];**

**char ans[2][20] = {"No", "Yes"};**

**int main () {**

**#ifdef LOCAL**

**// freopen("C:\\Users\\zzblack\\OneDrive\\文档\\原创题\\土豪村排队\\11.in","r",stdin);**

**// freopen("C:\\Users\\zzblack\\OneDrive\\文档\\原创题\\土豪村排队\\11.out","w",stdout);**

**#endif**

**int T = read(), cas = 1;**

**while (T--) {**

**int n = read();**

**memset(Hash, 0, sizeof Hash);**

**int flag = 1;**

**for (int i = 1; i <= n; i++) a[i] = read();**

**for (int i = 1; i <= n; i++) {**

**int x = a[i] + n - i;**

**if (Hash[x]) flag = 0;**

**Hash[x] = 1;**

**}**

**printf("Case %d: %s\n", cas++, ans[flag]);**

**}**

**return 0;**

**}**

# G

**并查集，出题欠考虑，错误估计暴力的复杂度$\_$，给放过了。**

**最笨的方法就是可持久化并查集**

**一个比较暴力的方法就是，对每个部落开一颗set(平衡树)每次合并部落的时候把两个set合并，并把原始的set clear掉，删除的时候，把该部落的所有set clear，并且再插入他自己的编号.**

**还有一个方法，因为节点是从1开始的，我们可以把0号节点赋予特定的含义，每次删除的时候，就把该部落的根节点和0合并表示该部落被解散，然后在合并的时候，比如需要合并u,v如果查询到u是和0相连的，此时，新开一个节点k（哈希一次，把u节点映射到k ，k一定不能和已有的节点编号冲突，简单的说就是ID[u] = k）表示u，把k与v合并，便解决了这个问题。时间复杂度O(n+m),空间复杂度O(m)**

**//-------------------------//**

**// /\\_/\ //**

**// \_\_\_\_/ o o \ Nya~~ //**

**// /~\_\_\_\_ = ¦Ø =/ //**

**// (\_\_\_\_\_\_)\_\_m\_m) //**

**//-------------------------//**

**#include <map>**

**#include <set>**

**#include <cmath>**

**#include <queue>**

**#include <stack>**

**#include <cstdio>**

**#include <cstdlib>**

**#include <cstring>**

**#include <iomanip>**

**#include <iostream>**

**#include <algorithm>**

**#define lson id << 1**

**#define rson id << 1 | 1**

**#define sacnf scanf**

**#define falg flag**

**using namespace std;**

**typedef long long ll;**

**typedef pair<int, int> P;**

**const int inf = 0x3f3f3f3f;**

**const int maxn = 1e5 + 15;**

**const int maxm = 2e6 + 15;**

**int par[maxm], id[maxn], idx, size[maxm];**

**void init(int n, int m) {**

**idx = n + 1;**

**for(int i = 0; i <= n + m; i++)**

**par[i] = i, size[i] = 1;**

**for(int i = 0; i <= n; i++)**

**id[i] = i;**

**}**

**int find(int x) {**

**if(par[x] == x) return x;**

**return par[x] = find(par[x]);**

**}**

**int Find(int x) {**

**int root = find(id[x]);**

**if(!root) {**

**id[x] = idx++;**

**par[id[x]] = id[x];**

**size[id[x]] = 1;**

**root = id[x];**

**}**

**return root;**

**}**

**void unite(int x, int y) {**

**int l = Find(x);**

**int r = Find(y);**

**if(l == r) return;**

**par[l] = r;**

**size[r] += size[l];**

**size[l] = 0;**

**}**

**void disunion(int x) {**

**x = Find(x);**

**par[x] = 0;**

**}**

**int main() {**

**int n, m, u, v, t; char s[5];**

**scanf("%d", &t);**

**while(t--) {**

**scanf("%d %d", &n, &m);**

**init(n, m);**

**while(m--) {**

**scanf("%s", s);**

**if(s[0] == 'U') {**

**scanf("%d %d", &u, &v);**

**unite(u, v);**

**}**

**else if(s[0] == 'D') {**

**scanf("%d", &u);**

**disunion(u);**

**}**

**else if(s[0] == 'S') {**

**scanf("%d", &u);**

**u = Find(u);**

**printf("%d\n", size[u]);**

**}**

**else if(s[0] == 'F') {**

**scanf("%d %d", &u, &v);**

**u = Find(u), v = Find(v);**

**if(u == v) puts("Yes");**

**else puts("No");**

**}**

**}**

**}**

**return 0;**

**}**

# H

**较难题**

**首先，数据范围极大，要求出T也是存不下的，首先必须找到fib数列的循环节。**

**考虑到不能太难，此题可以暴力循环找循环节，16941960即是此模数的循环节。**

**科学的方法：**

**首先我们知道fib数列模p如果出现了连续的1，0就意味这着开始循环了，因为接下来的项就是1 1 2 3 5等等。**

**那么很显然如果在第k位第一次出现了1，0，那么对于以后的1，0都可以表示为k\*m。**

**那么，现在我们考虑如果fib数列模p在第pos位第一次出现了0，那么设0前面的那个数为a，则接下来的序列将是a，0，a，**

**a，2a，3a，5a，8a，....。可以看出a的系数就是一个fib数列，那么我们就可以得到fib(k+i)%p=a\*fib(i)%p，其中i满**

**足0<i<k，所以进一步可以得到fib(i)=[a^j\*fib(i-k\*j)]%p。**

**那么我们现在先来说说如何求fib数模一个正整数n的循环节长度：**

**对于这个问题，我们先对n进行素因子分解，得到：http://img.blog.csdn.net/20130903175725359?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvQUNkcmVhbWVycw==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/SouthEast，然后先对每一个形如p^k的数计算循环节，然后它们**

**的最小公倍数就是n的循环节长度（当然这里面涉及到CRT等等方面的基础）。那么现在问题就是计算p^k的循环节，这个问题**

**可以进一步简化成求G(p)\*p^(k-1)。其中G(p)表示fib数列模素数p的循环节长度，所以现在的问题是如何求fib数列模一个**

**素数p的循环节长度。**

**求fib数列模p(p是素数)的最小循环节方法：**

**暴力枚举fib[i]%p==0的最小的i，然后记录pos=i+1，设a为fib[i]%p==0的前一位数，即a=fib[i-1]**

**那么我们知道fib数列模p的最小循环节长度一定是pos\*x，那么也就是说现在要求一个最小的数x，满足http://img.blog.csdn.net/20130903180816578?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvQUNkcmVhbWVycw==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/SouthEast,**

**求出x后，那么问题就解决了，剩下的就是合并等等。**

**本题找到循环节之后，我们就可以先把T求解出来对循环节取模，然后再用一般的矩阵快速幂加速求fib数列就行了，现在来求解T**

**注意到， 需要先拆除向上取整符号，把2带进去，变成，然后构造 M = ,可以用数学归纳法证明M是一个正整数，发现 < 1那么意味着M = T**

**于是去掉向上取整T = ,可以看出这是一个形如a[i] = L\*a[i-1] + K\*a[i-1]的a[i]的表达式，设特征方程,由特征方程的一些知识可以知道**

**L = ,**

**K = - = -1**

**那么a[i] = 3a[i-1] – a[i-2]，a[0] = 2, a[1] = 3但是很遗憾，虽然循环节求出来了，但是实际上f(0),f(1)是没有在循环节里面的，因为是从2开始的。我们需要把式子做一个变形，先减去2 那么递推式a[0] = 0, a[1] = 1, a[i] = 3a[i-1] – [i-2] + 2,构造3阶矩阵，然后用矩阵快速幂加速便可以求出，最后答案+2就是T的值，再利用矩阵快速幂求出f(T)即可。**

**时间复杂度 O(log(16941960));**

**//**

**// Created by Running Photon on 2016-03-21**

**// Copyright (c) 2015 Running Photon. All rights reserved.**

**//**

**#include <algorithm>**

**#include <cctype>**

**#include <cmath>**

**#include <cstdio>**

**#include <cstdlib>**

**#include <cstring>**

**#include <iomanip>**

**#include <iostream>**

**#include <map>**

**#include <queue>**

**#include <string>**

**#include <sstream>**

**#include <set>**

**#include <vector>**

**#include <stack>**

**#define ALL(x) x.begin(), x.end()**

**#define INS(x) inserter(x, x,begin())**

**#define ll long long**

**#define CLR(x) memset(x, 0, sizeof x)**

**using namespace std;**

**const ll MOD = 2238065148;**

**const ll inf = 0x3f3f3f3f;**

**const int maxn = 1e6 + 10;**

**const int maxv = 3;**

**const double eps = 1e-9;**

**ll mod;**

**struct Matrix {**

**ll mat[maxv][maxv];**

**Matrix() {CLR(mat);}**

**void clear() {**

**CLR(mat);**

**}**

**void print(int maxx) {**

**for(int i = 0; i < maxx; i++) {**

**for(int j = 0; j < maxx; j++) {**

**printf("%lld ", mat[i][j]);**

**}**

**puts("");**

**}**

**puts("");**

**}**

**Matrix operator\* (Matrix& rhs) {**

**Matrix ret;**

**ret.clear();**

**for(int i = 0; i < maxv; i++) {**

**for(int j = 0; j < maxv; j++) {**

**for(int k = 0; k < maxv; k++) {**

**ret.mat[i][j] = (ret.mat[i][j] + mat[i][k] \* rhs.mat[k][j] % mod + mod) % mod;**

**}**

**}**

**}**

**return ret;**

**}**

**Matrix operator^ (ll n) {**

**Matrix ret;**

**Matrix a = \*this;**

**ret.clear();**

**for(int i = 0; i < maxv; i++) {**

**ret.mat[i][i] = 1;**

**}**

**while(n) {**

**if(n & 1) ret = ret \* a;**

**a = a \* a;**

**n >>= 1;**

**}**

**return ret;**

**}**

**};**

**// d[i] = 3 \* d[i-1] - d[i-2] - 2 check oka**

**ll getT(ll m) {**

**if(m == 0) return 2;**

**if(m == 1) return 3;**

**if(m == 2) return 7;**

**Matrix res;**

**res.mat[0][0] = 3;**

**res.mat[0][1] = 1;**

**res.mat[1][0] = -1;**

**res.mat[2][0] = 2;**

**res.mat[2][2] = 1;**

**mod = 16941960;**

**res = res ^ (m - 2);**

**ll ans = (5LL \* res.mat[0][0] + res.mat[1][0] + res.mat[2][0] + mod) % mod;**

**return ans + 2;**

**}**

**ll solve(ll n) {**

**if(n < 1) return 0;**

**if(n <= 2) return 1;**

**Matrix res;**

**res.mat[0][0] = 1;**

**res.mat[0][1] = 1;**

**res.mat[1][0] = 1;**

**res.mat[1][1] = 0;**

**mod = MOD;**

**res = res ^ (n - 2);**

**ll ans = (res.mat[0][0] + res.mat[1][0] + mod) % mod;**

**return ans;**

**}**

**int main() {**

**#ifdef LOCAL**

**// freopen("1.in", "r", stdin);**

**// freopen("1.out","w",stdout);**

**#endif**

**// ios\_base::sync\_with\_stdio(0);**

**ll n;**

**int T;**

**scanf("%d", &T);**

**while(T--) {**

**cin >> n;**

**cout << getT(n) << endl;**

**// cout << solve(getT(n)) << endl;**

**}**

**return 0;**

**}**