

数据结构

树状数组[区间修改单点查询]

```
1  int n,m;
2  int a[50005] = {0},c[50005]; //对应原数组和树状数组
3
4  int lowbit(int x){
5      return x&(-x);
6  }
7
8  void updata(int i,int k){    //在i位置加上k
9      while(i <= n){
10         c[i] += k;
11         i += lowbit(i);
12     }
13 }
14
15 int getSum(int i){           //求D[1 - i]的和, 即A[i]值
16     int res = 0;
17     while(i > 0){
18         res += c[i];
19         i -= lowbit(i);
20     }
21     return res;
22 }
23
24 int main(){
25     cin>>n;27     for(int i = 1; i <= n; i++){
26         cin>>a[i];
27         updata(i,a[i] - a[i-1]);    //输入初值的时候, 也相当于更新了值
28     }
29
30     //[x,y]区间内加上k
31     updata(x,k);    //A[x] - A[x-1]增加k
32     updata(y+1,-k);    //A[y+1] - A[y]减少k
33
34     //查询i位置的值
35     int sum = getsum(i);
36
37     return 0;
38 }
```

线段树[单点修改区间查询]

```
1  #include <cstdio>
2  #include <iostream>
3  #include <algorithm>
4  #include <cmath>
5  #include <cstring>
6  #include <map>
```

```

7  #include <set>
8  #include <queue>
9  #include <string>
10 #include <vector>
11 using namespace std;
12 typedef long long ll;
13 typedef unsigned long long ull;
14 const int INF = 0x7fffffff;
15 const int mod = 1e9+7;
16 const double eps = 1e-5;
17 const int N = 1e5+10;
18
19 void redirect(){
20     #ifdef LOCAL
21         freopen("test.txt", "r", stdin);
22     #endif
23 }
24 inline ll read(){
25     ll f=1,x=0;char ch;
26     do{ch=getchar();if(ch=='-')f=-1;}while(ch<'0' || ch>'9');
27     do{x=x*10+ch-'0';ch=getchar();}while(ch>='0' && ch<='9');
28     return x*f;
29 }
30
31 int n,k;
32 int pos[N];int a[N];
33
34 struct NOOD {
35     int l, r, add, Max;
36 }tree[N * 4 + 5];
37 void Build(int L, int R, int x) {
38     tree[x].l = L, tree[x].r = R, tree[x].Max = 0;
39     if(L == R) {
40         tree[x].Max = a[L];
41         return ;
42     }
43     int mid = (L + R) / 2;
44     Build(L, mid, x * 2);
45     Build(mid + 1, R, x * 2 + 1);
46     tree[x].Max = max(tree[x * 2].Max, tree[x * 2 + 1].Max);
47 }
48 void PushDown(int x) {
49     if(tree[x].add) {
50         tree[x * 2].Max = tree[x].add;
51         tree[x * 2 + 1].Max = tree[x].add;
52         tree[x * 2].add = tree[x].add;
53         tree[x * 2 + 1].add = tree[x].add;
54         tree[x].add = 0;
55     }
56 }
57 void Update(int L, int R, int add, int x) {
58     if(L <= tree[x].l && tree[x].r <= R) {
59         tree[x].add = add;
60         tree[x].Max = add;
61         return ;
62     }
63     PushDown(x);
64     int mid = (tree[x].l + tree[x].r) / 2;

```

```

65     if(L <= mid)Update(L, R, add, x * 2);
66     if(R > mid)Update(L, R, add, x * 2 + 1);
67     tree[x].Max = max(tree[x * 2].Max, tree[x * 2 + 1].Max);
68 }
69
70 int Query(int L, int R, int x) {
71     if(L <= tree[x].l && tree[x].r <= R)return tree[x].Max;
72     PushDown(x);
73     int mid = (tree[x].l + tree[x].r) / 2;
74     int res = 0;
75     if(L <= mid) res = max(res, Query(L, R, x * 2));
76     if(R > mid) res = max(res, Query(L, R, x * 2 + 1));
77     return res;
78 }
79
80 int nxt[N];int ans[N];
81
82 int dfs(int i){
83     if(nxt[i]==0||ans[i]!=1) return ans[i];
84     else return ans[i]=dfs(nxt[i])+1;
85 }
86
87 int main(){
88     redirect();
89     int T;scanf("%d",&T);
90     while(T--){
91         scanf("%d%d",&n,&k);
92         memset(nxt,0,sizeof(nxt));memset(tree, 0, sizeof(tree));
93         for(int i=1;i<=n;i++){
94             scanf("%d",&a[i]);pos[a[i]]=i;ans[i]=1;
95         }
96         Build(1, n, 1);
97         for(int i=n;i>=1;i--){
98             update(pos[i], pos[i] , 0, 1);
99             int big = Query(max(pos[i]-k,1), min(pos[i]+k,n), 1);
100             if(big!=0) nxt[i]=big;
101         }
102
103         for(int i=1;i<=n;i++){
104             int ans = dfs(i);printf("%d%c",ans,i==n?'\\n':' ');
105         }
106     }
107     return 0;
108 }
109
110
111 /*
112 ---linux compile---
113 g++ aa.cpp -o aa
114 ./ aa
115 -----
116 author:dragon_bra
117 */

```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  const int N = 2e5+10;
5
6  struct node{
7      int data;
8  }_a[N];
9
10 bool operator < (node const &_a,node const &_b){
11     return _a.data<_b.data;
12 }
13 bool operator > (node const &_a,node const &_b){
14     return _a.data>_b.data;
15 }
16 bool operator == (node const &_a,node const &_b){
17     return _a.data<_b.data;
18 }
19 bool operator != (node const &_a,node const &_b){
20     return _a.data<_b.data;
21 }
22
23 int n,t,_root,_sz;
24 int _fa[N],_s[N][2],_cnt[N],_size[N];ll _sum[N];
25
26 inline int ws(int x){return _s[_fa[x]][1]==x;}//which son
27 void setson(int son,int f,int w){//0-left,C;1-right,◆◆;
28     if(son!=0) _fa[son]=f;
29     if(f!=0) _s[f][w]=son;
30 }
31 void maintain(int x){
32     _size[x]=_size[_s[x][0]]+_size[_s[x][1]] + _cnt[x];
33     _sum[x]=_sum[_s[x][0]] + _sum[_s[x][1]] + (ll)_cnt[x]*_a[x].data;
34 }
35 void rot(int x){
36     int f=_fa[x]; int ff=_fa[f]; int w=ws(x); int wf=ws(f);
37     int p=_s[x][!w];
38     setson(p,f,w);
39     setson(x,ff,wf);
40     setson(f,x,!w);//!w
41     maintain(f);
42     maintain(x);
43 }
44 void splay(int x){
45     for(;;_fa[x];rot(x)) if(_fa[_fa[x]]&&ws(_fa[x])==ws(x))
46         rot(_fa[x]);//zig-zag or zig-zig
47     _root=x;
48 }
49 void insert(int now,node p){
50     if(_root==0){
51         _root=++_sz;
52         _a[_sz]=p;
53         _size[_sz]=_cnt[_sz]=1;
54         return;
55     }
56     while(_a[now]!=p){
57         _size[now]++;
58         if(p>_a[now]){

```

```

58         if(_s[now][1]==0){
59             _a[++_sz]=p;
60             setson(_sz,now,1);
61         }
62         now=_s[now][1];
63     }
64     else{
65         if(_s[now][0]==0){
66             _a[++_sz]=p;
67             setson(_sz,now,0);
68         }
69         now=_s[now][0];
70     }
71 }
72 _size[now]++; _cnt[now]++;
73 splay(now);
74 }

```

数学

埃筛

```

1  //埃氏筛法
2  #define N 10000
3  int flag[N+1],p[N+1],pnum;
4  /*
5   flag[n] 表示n是否是素数，1是素数，0不是
6   prime   中是所有的素数按从小到大排列、
7   pnum    表示素数的个数
8   */
9  void CreatePrime(){
10     pnum=0;//初始化没有素数
11     //先将所有数看做素数，然后开始筛选
12     for(int i=0; i<=N; i++){
13         flag[i]=1;
14     }
15     //遍历筛去所有最大因数是i的合数
16     for(int i=2; i<=N; i++){
17         if(flag[i]==1){
18             //把素数记录下来
19             p[pnum++]=i;
20         }
21         //遍历已知素数表中比i的最小素因数小的素数，并筛去合数
22         for(int j=0; j<pnum && p[j]*i<=N; j++){
23             //筛去合数
24             flag[p[j]*i]=0;
25             if(i%p[j]==0)
26                 //找到i的最小素因数
27                 break;
28         }
29     }
30 }

```

大素数判定+泼辣的肉

```

1  #include<iostream>
2  #include<cstdio>
3  #include<cstring>
4  #include<algorithm>
5  #include<cstdlib>
6  using namespace std;
7  typedef long long ll;
8
9  const int S=20;
10
11 long long mult_mod(long long a,long long b,long long c)
12 {
13     a%=c;
14     b%=c;
15     long long ret=0;
16     while(b)
17     {
18         if(b&1){ret+=a;ret%=c;}
19         a<<=1;
20         if(a>=c)a%=c;
21         b>>=1;
22     }
23     return ret;
24 }
25
26 long long pow_mod(long long x,long long n,long long mod)
27 {
28     if(n==1)return x%mod;
29     x%=mod;
30     long long tmp=x;
31     long long ret=1;
32     while(n)
33     {
34         if(n&1) ret=mult_mod(ret,tmp,mod);
35         tmp=mult_mod(tmp,tmp,mod);
36         n>>=1;
37     }
38     return ret;
39 }
40
41 bool check(long long a,long long n,long long x,long long t)
42 {
43     long long ret=pow_mod(a,x,n);
44     long long last=ret;
45     for(int i=1;i<=t;i++)
46     {
47         ret=mult_mod(ret,ret,n);
48         if(ret==1&&last!=1&&last!=n-1) return true;//合数
49         last=ret;
50     }
51     if(ret!=1) return true;
52     return false;
53 }
54
55
56
57 bool Miller_Rabin(long long n)

```

```

58 {
59     if(n<2) return false;
60     if(n==2) return true;
61     if((n&1)==0) return false;
62     long long x=n-1;
63     long long t=0;
64     while((x&1)==0){x>>=1;t++;}
65     for(int i=0;i<S;i++)
66     {
67         long long a=rand()%(n-1)+1;
68         if(check(a,n,x,t))
69             return false;
70     }
71     return true;
72 }
73
74 long long factor[100];
75 int tol;
76
77 long long gcd(long long a,long long b)
78 {
79     if(a==0) return 1; //??????
80     if(a<0) return gcd(-a,b);
81     while(b)
82     {
83         long long t=a%b;
84         a=b;
85         b=t;
86     }
87     return a;
88 }
89
90 long long Pollard_rho(long long x,long long c)
91 {
92     long long i=1,k=2;
93     long long x0=rand()%x;
94     long long y=x0;
95     while(1)
96     {
97         i++;
98         x0=(mult_mod(x0,x0,x)+c)%x;
99         long long d=gcd(y-x0,x);
100         if(d!=1&&d!=x) return d;
101         if(y==x0) return x;
102         if(i==k){y=x0;k+=k;}
103     }
104 }
105
106 void findfac(long long n)
107 {
108     if(Miller_Rabin(n))
109     {
110         factor[tol++]=n;
111         return;
112     }
113     long long p=n;
114     while(p>=n){

```

```

115         if (Pollard_rho(p, rand()%(n-1)+1)!=0) p=Pollard_rho(p,rand()%(n-
116         1)+1);
117     }
118     findfac(p);
119     findfac(n/p);
120 }
121 int main(void)
122 {
123     int t;
124     cin >> t;
125     while(t--)
126     {
127         ll n;
128         scanf("%lld", &n);
129         if(Miller_Rabin(n)) printf("%lld\n", n);
130         else
131         {
132             tol = 0;
133             findfac(n);
134             ll ans = factor[0];
135             for(int i = 1; i < tol; i++)
136                 ans = min(ans, factor[i]);
137             printf("%lld\n", ans);
138         }
139     }
140     return 0;
141 }

```

第几个质数

```

1 //G++ 1560ms 6544k
2 #include <bits/stdc++.h>
3 #define ll long long
4 using namespace std;
5 ll f[340000],g[340000],n;
6 void init(){
7     ll i,j,m;
8     for(m=1;m*m<=n;++m)f[m]=n/m-1;
9     for(i=1;i<=m;++i)g[i]=i-1;
10    for(i=2;i<=m;++i){
11        if(g[i]==g[i-1])continue;
12        for(j=1;j<=min(m-1,n/i/i);++j){
13            if(i*j<m)f[j]-=f[i*j]-g[i-1];
14            else f[j]-=g[n/i/j]-g[i-1];
15        }
16        for(j=m;j>=i*i;--j)g[j]-=g[j/i]-g[i-1];
17    }
18 }
19 int main(){
20     while(scanf("%I64d",&n)!=EOF){
21         init();
22         cout<<f[1]<<endl;
23     }
24     return 0;
25 }

```



```

26  /*
27
28  o(n^3/4) 筛一个大质数是第几个质数
29  疑似 Meisell-Lehmer算法
30
31  */

```

费马小定理

$$\frac{a}{b} \% mod = a * b^{mod-2} \% mod$$

高精度

```

1  #include<iostream>
2  #include<string>
3  #include<cstring>
4  #include<cstdio>
5  using namespace std;
6  const int N = 1005;
7  struct bign
8  {
9      int len,s[N];
10     bign() { memset(s,0,sizeof(s)); len=1; }
11     bign(int num) { *this=num; }
12     bign(char *num) { *this=num; }
13     bign operator =(int num)
14     {
15         char c[N];
16         sprintf(c,"%d",num);
17         *this=c;
18         return *this;
19     }
20     bign operator =(const char *num)
21     {
22         len=strlen(num);
23         for (int i=0;i<len;i++) s[i]=num[len-1-i]-'0';
24         return *this;
25     }
26     string str()
27     {
28         string res="";
29         for (int i=0;i<len;i++) res=(char)(s[i]+'0')+res;
30         return res;
31     }
32     void clean()
33     {
34         while (len>1&&!s[len-1]) len--;
35     }
36     bign operator +(const bign &b)
37     {
38         bign c;
39         c.len=0;
40         for (int i=0,g=0;i<len||i<b.len;i++)
41         {
42             int x=g;

```

```

43         if (i<len) x+=s[i];
44         if (i<b.len) x+=b.s[i];
45         c.s[c.len++]=x%10;
46         g=x/10;
47     }
48     return c;
49 }
50 bign operator -(const bign &b)
51 {
52     bign c;
53     c.len=0;
54     int x;
55     for (int i=0,g=0;i<len;i++)
56     {
57         x=s[i]-g;
58         if (i<b.len) x-=b.s[i];
59         if (x>=0) g=0;
60         else{
61             x+=10;
62             g=1;
63         };
64         c.s[c.len++]=x;
65     }
66     c.clean();
67     return c;
68 }
69 bign operator *(const bign &b)
70 {
71     bign c;
72     c.len=len+b.len;
73     for (int i=0;i<len;i++) for (int j=0;j<b.len;j++)
74 c.s[i+j]+=s[i]*b.s[j];
75     for (int i=0;i<c.len-1;i++) { c.s[i+1]+=c.s[i]/10; c.s[i]%=10; }
76     c.clean();
77     return c;
78 }
79 bool operator <(const bign &b)
80 {
81     if (len!=b.len) return len<b.len;
82     for (int i=len-1;i>=0;i--)
83         if (s[i]!=b.s[i]) return s[i]<b.s[i];
84     return false;
85 }
86 bign operator +=(const bign &b)
87 {
88     *this=*this+b;
89     return *this;
90 }
91 bign operator -=(const bign &b)
92 {
93     *this=*this-b;
94     return *this;
95 }
96 istream& operator >>(istream &in,bign &x)
97 {
98     string s;
99     in>>s;

```

```

100     x=s.c_str();
101     return in;
102 }
103 ostream& operator <<(ostream &out,bign &x)
104 {
105     out<<x.str();
106     return out;
107 }
108 int main(){
109     bign a,b,c;
110     ios::sync_with_stdio(false);
111     cin>>a>>b;
112     //     cout<<a<<endl;
113     //     cout<<b<<endl;
114     c=a+b;
115     cout<<c<<endl;
116     return 0;
117 }

```

高精度除法

```

1  #include<iostream>
2  #include<algorithm>
3  using namespace std;
4  string div(string a,int b)//高精度a除以单精度b
5  {
6      string r,ans;
7      int d=0;
8      if(a=="0") return a;//特判
9      for(int i=0;i<a.size();i++)
10     {
11         r+=(d*10+a[i]-'0')/b+'0';//求出商
12         d=(d*10+(a[i]-'0'))%b;//求出余数
13     }
14     int p=0;
15     for(int i=0;i<r.size();i++)
16         if(r[i]!='0') {p=i;break;}
17     return r.substr(p);
18 }
19 int main()
20 {
21     string a;
22     int b;
23     while(cin>>a>>b)
24     {
25         cout<<div(a,b)<<endl;
26     }
27     return 0;
28 }

```

矩阵快速幂

```

1  #include <bits/stdc++.h>
2  using namespace std;
3

```

```

4   long long T,a,b,c,pp,mod;
5   long long n;
6
7   struct mat{
8       long long m[4][4];
9   };
10
11  mat mul(mat a,mat b){
12      mat ans;int i,j,k;
13      for(i=1;i<=3;i++)
14          for(j=1;j<=3;j++)
15              ans.m[i][j]=0;
16      for(i=1;i<=3;i++)
17          for(j=1;j<=3;j++)
18              for(k=1;k<=3;k++)
19                  ans.m[i][j]=(ans.m[i][j]+a.m[i][k]*b.m[k][j])%mod;
20      return ans;
21  }
22
23  mat matqp(mat t,long long p)
24  {
25      mat ans;
26      int i,j;
27      for(i=1;i<=3;i++)
28          for(j=1;j<=3;j++)
29              if(i==j)ans.m[i][j]=1;
30              else ans.m[i][j]=0;
31      while(p)
32      {
33          if(p&1)
34              ans=mul(ans,t);
35          t=mul(t,t);
36          p=p>>1;
37      }
38      return ans;
39  }
40
41  long long qp(long long a,long long p)
42  {
43      long long ans=1;
44      while(p){
45          if(p&1) {ans*=a;ans%=pp;}
46          a=a*a; a%=pp;
47          p=p>>1;
48      }
49      return ans;
50  }
51
52  int main(){
53      //scanf("%d",&T);
54      cin>>T;
55      while(T--){
56          {
57              //scanf("%I64d %d %d %d %d",&n,&a,&b,&c,&pp);
58              cin>>n>>a>>b>>c>>pp;
59              ///*
60              mod=pp-1;
61              /*/

```

```

62     mat base;
63     for(int i=1;i<=3;i++)
64         for(int j=1;j<=3;j++)
65             base.m[i][j]=0;
66     base.m[1][1]=c;base.m[1][2]=1;base.m[1][3]=1;base.m[2]
[1]=1;base.m[3][3]=1;
67     if(n==1){
68         cout<<1<<endl;
69     }
70     else{
71         mat out = matqp(base,n-2);
72         long long res = out.m[1][1]*b%mod + out.m[1][3]*b%mod;
73         //cout<<res<<endl;
74         long long ans = qp(a,res);
75         cout<<ans<<endl;
76     }
77 }
78
79 return 0;
80 }

```

扩展欧几里得

```

1  int extend_gcd( int a, int b, int &x, int &y ) {
2      if(b==0){
3          x=1;y=0;
4          return a;
5      }else{
6          int r = extend_gcd(b,a%b,y,x);
7          y-=x*(a/b);
8          return r;
9      }
10 }

```

欧拉函数

```

1  int phi(int x)
2  {
3      int ans = x;
4      for(int i = 2;i*i<=x;i++)
5      {
6          if(x%i==0)
7          {
8              ans = ans/i*(i-1);
9              while(x%i==0) x/=i;
10         }
11     }
12     if(x>1)
13         ans=ans/x*(x-1);
14     return ans;
15 }

```

欧拉筛

```

1 void init() {
2     phi[1] = 1;
3     for (int i = 2; i < MAXN; ++i) {
4         if (!vis[i]) {
5             phi[i] = i - 1;
6             pri[cnt++] = i;
7         }
8         for (int j = 0; j < cnt; ++j) {
9             if (1ll * i * pri[j] >= MAXN) break;
10            vis[i * pri[j]] = 1;
11            if (i % pri[j]) {
12                phi[i * pri[j]] = phi[i] * (pri[j] - 1);
13            } else {
14                phi[i * pri[j]] = phi[i] * pri[j];
15                break;
16            }
17        }
18    }
19 }

```

线性基

```

1 #include <bits/stdc++.h>
2 #define N 51
3 #define ll long long
4 using namespace std;
5
6 //给n个数，输出n个数里异或和的最大值
7
8 int n;
9 ll ans;
10 ll a[N], p[101];
11
12 inline ll read()
13 {
14     char ch = getchar();
15     ll x = 0, f = 1;
16     while(ch > '9' || ch < '0')
17     {
18         if(ch == '-')
19             f = -1;
20         ch = getchar();
21     }
22     while(ch >= '0' && ch <= '9')
23     {
24         x = x * 10 + ch - '0';
25         ch = getchar();
26     }
27     return x * f;
28 }
29
30 void Get_LB(ll x)
31 {
32     for(int i = 62; i >= 0; i--)
33     {
34         if(!(x >> (1ll)i))

```

```

35         continue;
36         if(!p[i])
37         {
38             p[i] = x;
39             break;
40         }
41         x ^= p[i];
42     }
43 }
44
45 int main()
46 {
47     n = read();
48     for(int i = 1; i <= n; i++)
49         Get_LB(a[i] = read());
50     for(int i = 62; i >= 0; i--)
51         if((ans ^ p[i]) > ans)
52             ans ^= p[i];
53     cout << ans;
54
55     return 0;
56 }

```

圓和矩形的面积交

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  #define INF 0x3f3f3f3f
4  #define eps 1e-17
5  #define pi acos(-1.0)
6  typedef long long ll;
7
8  void redirect() {
9      #ifdef LOCAL
10         freopen("1.in", "r", stdin);
11         freopen("1.out", "w", stdout);
12     #endif
13 }
14
15 int dcmp(double x){
16     if(fabs(x)<eps)return 0;
17     return x>0?1:-1;
18 }
19 struct Point{
20     double x,y;
21     Point(double _x=0,double _y=0){
22         x=_x;y=_y;
23     }
24 };
25 Point operator + (const Point &a,const Point &b){
26     return Point(a.x+b.x,a.y+b.y);
27 }
28 Point operator - (const Point &a,const Point &b){
29     return Point(a.x-b.x,a.y-b.y);
30 }
31 Point operator * (const Point &a,const double &p){

```

```

32     return Point(a.x*p,a.y*p);
33 }
34 Point operator / (const Point &a,const double &p){
35     return Point(a.x/p,a.y/p);
36 }
37 bool operator < (const Point &a,const Point &b){
38     return a.x<b.x||(dcmp(a.x-b.x)==0&&a.y<b.y);
39 }
40 bool operator == (const Point &a,const Point &b){
41     return dcmp(a.x-b.x)==0&&dcmp(a.y-b.y)==0;
42 }
43 double Dot(Point a,Point b){
44     return a.x*b.x+a.y*b.y;
45 }
46 double Length(Point a){
47     return sqrt(Dot(a,a));
48 }
49 double Angle(Point a,Point b){
50     return acos(Dot(a,b)/Length(a)/Length(b));
51 }
52 double angle(Point a){
53     return atan2(a.y,a.x);
54 }
55 double Cross(Point a,Point b){
56     return a.x*b.y-a.y*b.x;
57 }
58 Point vecunit(Point a){
59     return a/Length(a);
60 }
61 Point Normal(Point a){
62     return Point(-a.y,a.x)/Length(a);
63 }
64 Point Rotate(Point a,double rad){
65     return Point(a.x*cos(rad)-a.y*sin(rad),a.x*sin(rad)+a.y*cos(rad));
66 }
67 double Area2(Point a,Point b,Point c){
68     return Length(Cross(b-a,c-a));
69 }
70 bool OnSegment(Point p,Point a1,Point a2){
71     return dcmp(Cross(a1-p,a2-p))==0&&dcmp(Dot(a1-p,a2-p))<=0;
72 }
73 struct Line{
74     Point p,v;
75     double ang;
76     Line(){};
77     Line(Point p,Point v):p(p),v(v){
78         ang=atan2(v.y,v.x);
79     }
80     bool operator < (const Line &L) const {
81         return ang<L.ang;
82     }
83     Point point(double d){
84         return p+(v*d);
85     }
86 };
87 bool OnLeft(const Line &L,const Point &p){
88     return Cross(L.v,p-L.p)>=0;
89 }

```



```

90 Point GetLineIntersection(Point p,Point v,Point q,Point w){
91     Point u=p-q;
92     double t=Cross(w,u)/Cross(v,w);
93     return p+v*t;
94 }
95 Point GetLineIntersection(Line a,Line b){
96     return GetLineIntersection(a.p,a.v,b.p,b.v);
97 }
98 double PolyArea(vector<Point> p){
99     int n=p.size();
100    double ans=0;
101    for(int i=1;i<n-1;i++)
102        ans+=Cross(p[i]-p[0],p[i+1]-p[0]);
103    return fabs(ans)/2;
104 }
105 struct Circle{
106     Point c;
107     double r;
108     Circle(){}
109     Circle(Point c, double r):c(c), r(r){}
110     Point point(double a) { // 0 0 0 0 0 P L 0 0 0 0 0 0 0
111         return Point(c.x+cos(a)*r, c.y+sin(a)*r);
112     }
113 };
114
115 bool InCircle(Point x,Circle c){
116     return dcmp(c.r-Length(c.c-x))>=0;
117 }
118 bool OnCircle(Point x,Circle c){
119     return dcmp(c.r-Length(c.c-x))==0;
120 }
121 int getSegCircleIntersection(Line L,Circle C,Point *sol){
122     Point nor=Normal(L.v);
123     Line p1=Line(C.c,nor);
124     Point ip=GetLineIntersection(p1,L);
125     double dis=Length(ip-C.c);
126     if(dcmp(dis-C.r)>0)return 0;
127     Point dx=vecunit(L.v)*sqrt(C.r*C.r-dis*dis);
128     int ret=0;
129     sol[ret]=ip+dx;
130     if(OnSegment(sol[ret],L.p,L.point(1)))ret++;
131     sol[ret]=ip-dx;
132     if(OnSegment(sol[ret],L.p,L.point(1)))ret++;
133     return ret;
134 }
135 double SegCircleArea(Circle C,Point a,Point b){
136     double a1=angle(a-C.c);
137     double a2=angle(b-C.c);
138     double da=fabs(a1-a2);
139     if(da>pi)da=pi*2-da;
140     return dcmp(Cross(b-C.c,a-C.c))*da*C.r*C.r/2.0;
141 }
142 double PolyCircleArea(Circle C,Point *p,int n){
143     double ret=0;
144     Point sol[2];
145     p[n]=p[0];
146     for(int i=0;i<n;i++){
147         double t1,t2;

```

```

148     int cnt=getSegCircleIntersection(Line(p[i],p[i+1]-p[i]),C,sol);
    // 计算两圆交点个数
149     if(cnt==0){ // 两圆相离或内含，无交点
150 
151         if(!InCircle(p[i],C)||!InCircle(p[i+1],C))ret+=SegCircleArea(C,p[i],p[i+1]);
        // 若两圆不相交且都不在圆内，则面积为两圆面积之和
152     }
153     else ret+=Cross(p[i+1]-C.c,p[i]-C.c)/2; // 若在圆内，则为弦长平方的一半
154     if(cnt==1){
155         if(InCircle(p[i],C)&&!InCircle(p[i+1],C)||OnCircle(p[i+1],C))ret+=Cross(sol[0]-C.c,p[i]-C.c)/2,ret+=SegCircleArea(C,sol[0],p[i+1]); // 若一圆在圆内，另一圆与圆相切，则面积为两圆面积之和
156         else ret+=SegCircleArea(C,p[i],sol[0]),ret+=Cross(p[i+1]-C.c,sol[0]-C.c)/2; // 若一圆在圆内，另一圆与圆相交，则面积为两圆面积之和
157     }
158     if(cnt==2){
159         if((p[i]<p[i+1])^(sol[0]<sol[1]))swap(sol[0],sol[1]);
160         ret+=SegCircleArea(C,p[i],sol[0]);
161         ret+=Cross(sol[1]-C.c,sol[0]-C.c)/2;
162         ret+=SegCircleArea(C,sol[1],p[i+1]);
163     }
164     return fabs(ret);
165 }
166 Point p[5];
167 int main(){
168     redirect();
169     double R,x1,y1,x2,y2,x3,y3;
170     cin>>x1>>y1>>R>>x2>>y2>>x3>>y3;
171 
172     Circle C=Circle(Point(x1,y1),R);
173     if(x2>x3)swap(x2,x3);
174     if(y2>y3)swap(y2,y3);
175     p[0]=Point(x2,y2);
176     p[2]=Point(x3,y3);
177     p[1]=Point(x3,y2);
178     p[3]=Point(x2,y3);
179     double ans=PolyCircleArea(C,p,4);
180     if(ans < -eps) ans = -ans;
181     printf("%.4lf\n",ans);
182 
183     return 0;
184 }

```

Min25

```
1  /*
2  * @ author: dragon_bra
3  * @ email: tommy514@foxmail.com
4  * @ data: 2020-09-20 13:59
5  */
6  // n以内素数和
7  #include <algorithm>
8  #include <cmath>
9  #include <cstdio>
```

```

10 #include <cstdlib>
11 #include <cstring>
12 #include <iostream>
13 #include <sstream>
14 #include <map>
15 #include <set>
16 #include <queue>
17 #include <vector>
18 using namespace std;
19
20 const int N = 2e5 + 10;
21
22 typedef long long ll;
23
24 void redirect() {
25     #ifdef LOCAL
26         freopen("in.txt", "r", stdin);
27         freopen("out.txt", "w", stdout);
28     #endif
29 }
30
31 int T; ll n, K;
32
33 namespace Min25 {
34
35     ll prime[N], id1[N], id2[N], flag[N], ncnt, m;
36
37     ll g[N], sum[N], a[N], T, n;
38
39     inline int ID(ll x) {
40         return x <= T ? id1[x] : id2[n / x];
41     }
42
43     inline ll calc(ll x) {
44         if (x % 2) return (x+1)/2 % K * x % K;
45         else return x/2 % K * (x+1) % K;
46         // return x * (x + 1) / 2 - 1;
47     }
48
49     inline ll f(ll x) {
50         return x;
51     }
52
53     inline void init() {
54         T = sqrt(n + 0.5);
55         ncnt = 0; m = 0;
56         memset(flag, 0, sizeof flag);
57         memset(sum, 0, sizeof sum);
58         memset(prime, 0, sizeof prime);
59         memset(a, 0, sizeof a);
60         for (int i = 2; i <= T; i++) {
61             if (!flag[i]) prime[++ncnt] = i, sum[ncnt] = (sum[ncnt - 1] +
62 i)%K;
63             for (int j = 1; j <= ncnt && i * prime[j] <= T; j++) {
64                 flag[i * prime[j]] = 1;
65                 if (i % prime[j] == 0) break;
66             }
67         }
68     }
69 }

```

```

67         for (ll l = 1; l <= n; l = n / (n / l) + 1) {
68             a[++m] = n / l;
69             if (a[m] <= T) id1[a[m]] = m; else id2[n / a[m]] = m;
70             g[m] = calc(a[m]) % K;
71         }
72         for (int i = 1; i <= ncnt; i++)
73             for (int j = 1; j <= m && (ll)prime[i] * prime[i] <= a[j];
j++) {
74
75                 g[j] = (g[j] - (ll)prime[i] * (g[id(a[j] / prime[i])] -
sum[i - 1] + K) % K + K) % K;
76             }
77         }
78
79         inline ll solve(ll x) {
80             if (x <= 1) return x;
81             return n = x, init(), g[id(n)];
82         }
83
84     }
85
86     int main() {
87         redirect();
88
89         scanf("%d", &T);
90         while (T--) {
91             scanf("%lld %lld", &n, &K);
92             n = n+1;
93             ll ans = 0;
94             if (n%2) {
95                 ans = (n+1)/2 % K * n % K;
96             } else {
97                 ans = n/2 % K * (n+1) % K;
98             }
99             ans += Min25::solve(n) - 5;
100             ans %= K;
101             printf("%lld\n", ans);
102         }
103     }

```

Zeller Formula

```

1  int Day(int year, int month, int day){
2      int ret = 0;
3      int c, y, m, d;
4      if(month <= 2){
5          c = ( year - 1 ) / 100;
6          y = ( year - 1 ) % 100;
7          m = month + 12;
8          d = day;
9      }
10     else{
11         c = year / 100;
12         y = year % 100;
13         m = month;
14         d = day;

```

```

15     }
16     ret = y + y / 4 + c / 4 - 2 * c + 26 * ( m + 1 ) / 10 + d - 1;
17     ret = ret >= 0 ? ( ret % 7 ) : ( ret % 7 + 7 );
18     return ret;
19 }

```

网络流

二分图最大流

```

1  const int maxn = 200005;
2  const int INF = 0x3f3f3f3f;
3
4  struct Edge
5  {
6      int from, to, flow, cap;
7      Edge(int x, int y, int f, int c) : from(x), to(y), flow(f), cap(c) {}
8  };
9
10 vector<Edge> edges;
11 vector<int> G[maxn];
12 int cur[maxn], d[maxn];
13 int S,T;
14 int cnt;
15
16 inline void addedge(int from, int to, int cap)
17 {
18     edges.push_back(Edge(from, to, 0, cap));
19     edges.push_back(Edge(to, from, 0, 0));
20     int m = edges.size();
21     G[from].push_back(m - 2);
22     G[to].push_back(m - 1);
23 }
24
25 int dfs(int u, int a)
26 {
27     if (u == T || a == 0)
28     {
29         return a;
30     }
31     int flow = 0, f;
32     for (int &i = cur[u]; i < G[u].size(); i++)
33     {
34         Edge &e = edges[G[u][i]];
35         if (d[e.to] > d[u] && (f = dfs(e.to, min(a, e.cap - e.flow))) > 0)
36         {
37             flow += f;
38             e.flow += f;
39             edges[G[u][i] ^ 1].flow -= f;
40             a -= f;
41             if (a == 0)
42             {
43                 break;
44             }
45         }
46     }
47     return flow;
48 }

```

```

46     }
47     if (a)
48     {
49         d[u] = -1;
50     }
51     return flow;
52 }
53
54 bool bfs()
55 {
56     memset(d, -1, (T + 1) * sizeof(int));
57     queue<int> q;
58     q.push(S);
59     d[S] = 0;
60     while (!q.empty())
61     {
62         int u = q.front();
63         q.pop();
64         for (int i = 0; i < G[u].size(); i++)
65         {
66             Edge &e = edges[G[u][i]];
67             if (d[e.to] == -1 && e.cap > e.flow)
68             {
69                 d[e.to] = d[u] + 1;
70                 q.push(e.to);
71             }
72         }
73     }
74     return d[T] != -1;
75 }
76
77 int max_flow()
78 {
79     int ans = 0;
80     while (bfs())
81     {
82         memset(cur, 0, (T+1)*sizeof(int));
83         ans += dfs(S, INF);
84     }
85     return ans;
86 }

```

Dinic (Node版本)

```

1 //以下是网络流模板
2 struct Edge{
3     int to,nxt,w;
4 }e[M<<1];
5 int head[N],ecnt;
6 void AddEdge(int u,int v,int w) {
7     e[ecnt]=(Edge){v,head[u],w};
8     head[u]=ecnt++;
9 }
10 void Link(int u,int v,int w){ AddEdge(u,v,w),AddEdge(v,u,0); }
11 #define erep(u,i) for(int i=head[u];~i;i=e[i].nxt)
12

```

```

13 int dis[N];
14 int Bfs(){
15     static queue <int> que;
16     rep(i,1,vc) dis[i]=INF;
17     que.push(S),dis[S]=0;
18     while(!que.empty()) {
19         int u=que.front(); que.pop();
20         erep(u,i) {
21             int v=e[i].to,w=e[i].w;
22             if(!w || dis[v]<=dis[u]+1) continue;
23             dis[v]=dis[u]+1,que.push(v);
24         }
25     }
26     return dis[T]<INF;
27 }
28
29 int Dfs(int u,int flowin) {
30     if(u==T) return flowin;
31     int flowout=0;
32     erep(u,i) {
33         int v=e[i].to,w=e[i].w;
34         if(dis[v]!=dis[u]+1 || !w) continue;
35         int t=Dfs(v,min(flowin-flowout,w));
36         flowout+=t,e[i].w-=t,e[i^1].w+=t;
37         if(flowin==flowout) break;
38     }
39     if(!flowout) dis[u]=0;
40     return flowout;
41 }
42
43 int Dinic(){
44     int ans=0;
45     while(Bfs()) ans+=Dfs(S,INF);
46     return ans;
47 }

```

字符串

```

1  #include <cstdio>
2  #include <iostream>
3  #include <algorithm>
4  #include <cmath>
5  #include <cstring>
6  #include <map>
7  #include <set>
8  #include <queue>
9  #include <string>
10 #include <vector>
11 using namespace std;
12 typedef long long ll;
13 typedef unsigned long long ull;
14 const int INF = 0x7fffffff;
15 const int mod = 1e9+7;
16 const double eps = 1e-5;
17 const int N = 1e6+10;
18

```

```

19 void redirect() {
20     #ifdef LOCAL
21         //freopen("test.txt","r",stdin);
22         //freopen("out.txt","w",stdout);
23     #endif
24 }
25 inline ll read() {
26     ll f=1,x=0;char ch;
27     do {ch=getchar(); if(ch=='-') f=-1;} while (ch<'0' || ch>'9');
28     do {x=x*10+ch-'0'; ch=getchar(); } while (ch>='0' && ch<='9');
29     return x*f;
30 }
31
32 struct Trie {
33     int next[N][26],fail[N],end[N];
34     int root,L;
35     int newnode(){
36         for(int i=0;i<26;i++)
37             next[L][i] = -1;
38         end[L++] = 0;
39         return L-1;
40     }
41     void init(){
42         L = 0;
43         root = newnode();
44     }
45     void insert(char buf[]){
46         int len = strlen(buf);
47         int now = root;
48         for(int i=0;i<len;i++){
49             if(next[now][buf[i]-'a'] == -1)
50                 next[now][buf[i]-'a'] = newnode();
51             now = next[now][buf[i]-'a'];
52         }
53         end[now]++;
54     }
55     void build(){
56         queue<int>Q;
57         fail[root] = root;
58         for(int i=0;i<26;i++)
59             if(next[root][i] == -1)
60                 next[root][i] = root;
61         else{
62             fail[next[root][i]] = root;
63             Q.push(next[root][i]);
64         }
65         while( !Q.empty() ) {
66             int now = Q.front();
67             Q.pop();
68             for(int i=0;i<26;i++)
69                 if(next[now][i] == -1)
70                     next[now][i] = next[fail[now]][i];
71                 else{
72                     fail[next[now][i]] = next[fail[now]][i];
73                     Q.push(next[now][i]);
74                 }
75         }
76     }

```



```

77     int query(char buf[]){
78         int len = strlen(buf);
79         int now = root;
80         int res = 0;
81         for(int i=0;i<len;i++){
82             now = next[now][buf[i]-'a'];
83             int temp = now;
84             while( temp != root ) {
85                 res += end[temp];
86                 end[temp] = 0;
87                 temp = fail[temp];
88             }
89         }
90         return res;
91     }
92     void debug(){
93         for(int i = 0;i < L;i++){
94             printf("id=%3d,fail=%3d,end=%3d,chi=%3d",i,fail[i],end[i]);
95             for(int j = 0;j < 26;j++)
96                 printf("%2d",next[i][j]);
97             printf("\n");
98         }
99     }
100 };
101 char buf[N];
102 Trie ac;
103
104 int main() {
105     redirect();
106     int T; scanf("%d",&T);
107     int n;
108     while ( T-- ) {
109         scanf("%d",&n);
110         ac.init();
111         for(int i=0;i<n;i++){
112             scanf("%s",buf);
113             ac.insert(buf);
114         }
115         ac.build();
116         scanf("%s",buf);
117         printf("%d\n",ac.query(buf));
118     }
119 }
120
121 /*
122 -----
123 author:dragon_bra
124 -----
125 */

```

KMP

```

1 void makeNext(string s) {
2     int i = 0, k = -1;
3     next[0] = -1;
4     int len = strlen(s);

```

```

5     while (i < len-1) {
6         while (k >= 0 && s[i] != s[k]) k = next[k];
7         i ++; k ++;
8         if (s[i] == s[k]) next[i] = next[k];
9         else next[i] = k;
10    }
11 }
12
13 int kmpMatch(string t, string p) {
14     int i = 0, j = 0;
15     int len_1 = strlen(t), len2 = strlen(p);
16     while (i < len_1 && j < len_2) {
17         if (i == -1 || p[i] == c[j]) {
18             i ++; j ++;
19         } else {
20             i = next[i];
21         }
22     }
23     if (i >= len_1) return j - len_1 + 1;
24     else return 0;
25 }

```

Manachar

```

1  /*
2  * @ author: dragon_bra
3  * @ email: tommy514@foxmail.com
4  * @ data: 2020-05-16 15:19
5  */
6
7  #include <algorithm>
8  #include <cmath>
9  #include <cstdio>
10 #include <cstdlib>
11 #include <cstring>
12 #include <iostream>
13 #include <sstream>
14 #include <map>
15 #include <set>
16 #include <queue>
17 #include <vector>
18
19 using namespace std;
20
21 typedef long long ll;
22 const int INF = 0x3f3f3f3f;
23 const int mod = 1e9+7;
24 const double eps = 1e-5;
25 const int N = 2e5 + 10;
26
27 void redirect() {
28     #ifdef LOCAL
29         freopen("in.txt", "r", stdin);
30         freopen("out.txt", "w", stdout);
31     #endif
32 }

```

```

33
34 int p[N*2];
35 char str[N*2],t[N*2];
36
37 int Manacher(char *str,int len){
38     // 初始化部分
39     t[0] = '$';t[1] = '#';
40     int tot = 2;
41     for(int i=0; i<len; i++){
42         t[tot++]=str[i];
43         t[tot++]='#';
44     }
45
46     int mx = 0,id = 0,reslen = 0,resCenter = 0;
47     for(int i=0; i<tot; i++){
48         if(i<mx) p[i] = min(p[2*id - i] , mx - i); // 2*id - i = id - (i-
id); j和i关于id对称;
49         else p[i] = 1; // i比mx大了,也就是当前最大的回文串够不着它了
50
51         while( t[i+p[i]] == t[i-p[i]] ) p[i] ++; // 计算i为中心大时候,最大的回
文字串有多大
52         if(p[i]+i > mx){
53             mx = i + p[i];
54             id = i;
55         }
56
57         if(reslen < p[i]) {
58             reslen = p[i], resCenter = i;
59         }
60
61     }
62     return reslen;
63 }
64
65 int main(){
66     while(~scanf("%s", str)){
67         int len = strlen(str);
68         printf("%d\n",Manacher(str,len)-1);
69     }
70     return 0;
71 }

```

DFS

DSU (树上启发式合并)

```

1  /*
2
3  DSU-on-tree
4  树上启发式合并
5  重点: {
6      dfs1(): 找出所有节点的重儿子, 记录每个节点的子树大小
7      dfs2(): 搜索下去更新答案,
8              如果是重儿子,
9              将兄弟所有的集合合并到重儿子, 并将重儿子的答案合并到父亲节点

```

```

10         else 如果是轻儿子
11             寻找他的重儿子并先把答案合并到自己
12     }
13
14     */
15     #include <bits/stdc++.h>
16     using namespace std;
17
18     typedef long long ll;
19     const int N = 1e5 + 5;
20
21     void redirect() {
22         #ifdef LOCAL
23             freopen("1.in", "r", stdin);
24             freopen("1.out", "w", stdout);
25         #endif
26     }
27
28     int n, f[N];
29     int son[N], size[N];
30     ll ans[N], rans[N];
31
32     vector<int> G[N];
33     set<ll> S[N];
34
35     void merge(int a, int b) {
36         while(!S[b].empty()){
37             ll t = *(S[b].begin()); S[b].erase(t);
38
39             ll up=0, low=0;
40
41             if( S[a].upper_bound(t) == S[a].begin() ) {
42                 up = *S[a].begin();
43                 ans[a] += (up - t) * (up - t);
44             } else if( S[a].upper_bound(t) == S[a].end() ) {
45                 low = *(--S[a].lower_bound(t));
46                 ans[a] += (t - low) * (t - low);
47             } else {
48                 up = *(S[a].upper_bound(t)); low = *(--S[a].lower_bound(t));
49                 ans[a] -= (up - low) * (up - low); ans[a] += (up - t) *
50                 (up - t); ans[a] += (t - low) * (t - low);
51             }
52             S[a].insert(t);
53         }
54     }
55
56     void dfs1(ll u, ll fa) { //记录了所有子树的size 和 每个节点的重儿子
57         size[u] = 1;
58         for (auto v:G[u]) {
59             dfs1(v, u);
60             size[u] += size[v];
61             if (size[v] > size[son[u]]) son[u] = v;
62         }
63     }
64
65     void dfs2(ll u, ll fa, bool keep, bool isson){

```

```

66     for( auto v:G[u] ) {
67         if( v!=son[u] ){
68             dfs2(v,u,0,0);
69         }
70     }
71
72     if( son[u] ) {
73         dfs2(son[u],u,1,1);
74     }
75
76     if( keep ) {
77         for( auto v:G[fa] ) {
78             if( u==v ) continue;
79             merge( u, v );
80         }
81
82         if( S[fa].size() < S[u].size() ) S[fa].swap(S[u]),
swap(ans[fa],ans[u]);
83         merge( fa, u );
84         rans[fa] = ans[fa];
85     }
86 }
87
88 int main() {
89     redirect();
90
91     scanf("%d",&n); f[1] = 1; S[1].insert(1);
92     for(11 i=2;i<=n;i++){
93         scanf("%d",&f[i]);
94         G[ f[i] ].push_back(i); S[i].insert(i);
95     }
96
97     dfs1(1,1);
98     dfs2(1,1,0,0);
99
100    for(11 i=1;i<=n;i++) {
101        printf("%11d\n",rans[ i ]);
102    }
103
104    return 0;
105 }
106
107 /*
108 -----
109 author:dragon_bra
110 -----
111 */

```

STL&杂项

优先队列

```

1  #include<iostream>
2  #include<vector>
3  #include<queue>

```

```

4   using namespace std;
5   int tmp[100];
6   struct cmp1{
7       bool operator()(int x,int y)
8       {
9           return x>y;//小的优先级高 ,从小到大排
10      }
11  };
12  struct cmp2{
13      bool operator()(const int x,const int y)
14      {
15          return tmp[x]>tmp[y];
16      }
17  };
18  struct node{
19      int x,y;
20      friend bool operator<(node a,node b)
21      {
22          return a.x>b.x;//按x从小到大排
23      }
24  };
25  priority_queue<int>q1;
26  priority_queue<int,vector<int>,cmp1>q2;
27  priority_queue<int,vector<int>,cmp2>q3;
28  priority_queue<node>q4;
29  int main()
30  {
31      int i,j,k,m,n;
32      int x,y;
33      node a;
34      while(cin>>n)
35      {
36          for(int i=0;i<n;i++)
37          {
38              cin>>a.y>>a.x;
39              q4.push(a);
40          }
41          cout<<endl;
42          while(!q4.empty())
43          {
44              cout<<q4.top().y<<" "<<q4.top().x<<" "<<endl;
45              q4.pop();
46          }
47          cout<<endl;
48
49          int t;
50          for(i=0;i<n;i++)
51          {
52              cin>>t;
53              q2.push(t);
54          }
55          while(!q2.empty())
56          {
57              cout<<q2.top()<<endl;
58              q2.pop();
59          }
60          cout<<endl;
61      }

```

```
62     return 0;
63 }
```

exmu

```
1  #include <cstdio>
2  #include <iostream>
3  #include <algorithm>
4  #include <cmath>
5  #include <cstring>
6  #include <map>
7  #include <set>
8  #include <queue>
9  #include <string>
10 #include <vector>
11 using namespace std;
12 typedef long long ll;
13 typedef unsigned long long ull;
14 const int INF = 0x7fffffff;
15 const int mod = 1e9+7;
16 const double eps = 1e-5;
17 const int N = 1e5+10;
18
19 void redirect() {
20     #ifdef LOCAL
21         freopen("test.txt", "r", stdin);
22         //freopen("out.txt", "w", stdout);
23     #endif
24 }
25 inline ll read() {
26     ll f=1, x=0; char ch;
27     do {ch=getchar(); if(ch=='-') f=-1;} while (ch<'0' || ch>'9');
28     do {x=x*10+ch-'0'; ch=getchar(); } while (ch>='0' && ch<='9');
29     return x*f;
30 }
31
32 int main() {
33     //redirect();
34     cout<<"Hello world."<<endl;
35 }
36
37 /*
38 -----
39 author:dragon_bra
40 -----
41 */
```

LIS

```
1  /*
2  * @ author: dragon_bra
3  * @ email: tommy514@foxmail.com
4  * @ data: 2020-07-25 12:12
5  */
6
```

```

7  #include <algorithm>
8  #include <cmath>
9  #include <cstdio>
10 #include <cstdlib>
11 #include <cstring>
12 #include <iostream>
13 #include <sstream>
14 #include <map>
15 #include <set>
16 #include <queue>
17 #include <vector>
18
19 using namespace std;
20
21 typedef long long ll;
22 const int INF = 0x3f3f3f3f;
23 const int mod = 1e9+7;
24 const double eps = 1e-5;
25 const int N = 1e3 + 10;
26
27 void redirect() {
28     #ifdef LOCAL
29         freopen("in.txt", "r", stdin);
30         freopen("out.txt", "w", stdout);
31     #endif
32 }
33
34 int n, a[N];
35 int f[N];
36
37 int lis(int x) {
38     f[0] = -INF;
39     int s = 0, t;
40     for(int i = 1; i <= n; i++) {
41         t = a[i+x-1];
42         if(t > f[s]) f[++s] = t;
43         else {
44             int l = 1, r = s, m;
45             while(l <= r) {
46                 m = (l+r)/2;
47                 if(t > f[m]) l = m+1;
48                 else r = m-1;
49             }
50             f[l] = t;
51         }
52     }
53     return s;
54 }
55
56 int main() {
57     redirect();
58
59     cin >> n;
60     for (int i = 1; i <= n; i++) {
61         cin >> a[i];
62         a[i+n] = a[i];
63     }
64

```



```

65     int mx = 0;
66     for (int i=1; i<=n; i++) {
67         mx = max(mx, lis(i));
68     }
69
70     cout << n - mx << endl;
71 }

```

Tarjan

```

1
2 void tarjan(int i) {
3     int j;
4     DFN[i]=LOW[i]=++Dindex;
5     instack[i]=true;
6     Stap[++Stop]=i;
7     for (edge *e=V[i];e=e->next)
8     {
9         j=e->t;
10        if (!DFN[j])
11        {
12            tarjan(j);
13            if (LOW[j]<LOW[i])
14                LOW[i]=LOW[j];
15        }
16        else if (instack[j] && DFN[j]<LOW[i])
17            LOW[i]=DFN[j];
18    }
19    if (DFN[i]==LOW[i])
20    {
21        Bcnt++;
22        do
23        {
24            j=Stap[Stop--];
25            instack[j]=false;
26            Belong[j]=Bcnt;
27        }
28        while (j!=i);
29    }
30 }
31 void solve()
32 {
33     int i;
34     Stop=Bcnt=Dindex=0;
35     memset(DFN,0,sizeof(DFN));
36     for (i=1;i<=N;i++)
37         if (!DFN[i])
38             tarjan(i);
39 }

```