ICPC Templates

Leachim

February 1, 2021

Contents

11 杂项   **93**

1 图论

1.1 最小生成树 Kruskal

```cpp
#include <cstdio>
#include <algorithm>

#define MAXN 5005
#define MAXM 200005
#define _for(i,a,b) for(int i=(a);i<=(b);i++)

using namespace std;

int read(){
    int ng=0,x=0;
    char ch=getchar();
    for(;ch<'0' || ch>'9';ch=getchar()) ng|=ch=='-';
    for(;ch>='0' && ch<='9';ch=getchar()) x=(x<<3)+(x<<1)+ch-'0';
    return ng?-x:x;
}

struct road
{
    int f,g,w;
}r[MAXM];

struct cmpf
{
    bool operator() (const road& a, const road& b) const {
        return a.w<b.w;
    }
};

int n,m,ans=0,fa[MAXN];

int find(int x){
    return fa[x]==x?x:fa[x]=find(fa[x]);
}

void kruskal(){
    int rx,ry;

    sort(r+1, r+m+1, cmpf());
    _for(i,1,m){
        rx=find(r[i].f);
        ry=find(r[i].g);
        if(rx==ry)
            continue;
        fa[rx]=ry;
        ans+=r[i].w;
    }
}

```

```
50  int main(){
51      n=read();m=read();
52      _for(i,1,m){
53          r[i].f=read();
54          r[i].g=read();
55          r[i].w=read();
56      }
57      _for(i,1,n)
58          fa[i]=i;
59      kruskal();
60      _for(i,2,n){
61          if(find(i)!=find(i-1)){
62              puts("orz");
63              return 0;
64          }
65      }
66      printf("%d", ans);
67  }
```

1.2 最小生成树 Prim

```
1   #include <cstdio>
2   #include <cstdlib>
3   #include <queue>
4
5   #define MAXN 5005
6   #define MAXM 200005
7
8   using namespace std;
9
10  struct edge
11  {
12      int v,to,next;
13  }e[MAXM<<1];
14
15  int n,m,f,g,w,tot=0,head[MAXN],ans=0,flag[MAXN];
16
17  void add(int x, int y, int z){
18      e[++tot].v=z;
19      e[tot].to=y;
20      e[tot].next=head[x];
21      head[x]=tot;
22  }
23
24  struct HeapNode
25  {
26      int v,u;
27      bool operator <(const HeapNode& a) const{
28          return v > a.v;
29      }
30  };
31
```

```
32  priority_queue<HeapNode> Q;
33
34  void prim(){
35      for(int i=1;i<=n;i++)
36          flag[i]=0;
37      flag[1]=1;
38      for(int p=head[1];p;p=e[p].next)
39          Q.push((HeapNode){e[p].v,e[p].to});
40      for(int i=1,u;i<n;i++){
41          while(flag[Q.top().u]){
42              Q.pop();
43              if(Q.empty()){
44                  puts("orz");
45                  exit(0);
46              }
47          }
48          ans+=Q.top().v;
49          u=Q.top().u;
50          Q.pop();
51          flag[u]=1;
52          for(int p=head[u];p;p=e[p].next)
53              if(!flag[e[p].to])
54                  Q.push((HeapNode){e[p].v,e[p].to});
55      }
56  }
57
58  int main(){
59      scanf("%d %d", &n, &m);
60      for(int i=1;i<=n;i++)
61          head[i]=0;
62      for(int i=1;i<=m;i++){
63          scanf("%d %d %d",&f,&g,&w);
64          add(f,g,w);
65          add(g,f,w);
66      }
67      prim();
68      printf("%d",ans);
69      return 0;
70  }
```

1.3   最短路 Djikstra

```
1   #include <bits/stdc++.h>
2   #define LL long long
3   using namespace std;
4   const int inf=0x3f3f3f3f;
5   const LL INF=0x3f3f3f3f3f3f3f3f;
6   const int MOD=998244353;
7   const int MAXN=2000005;
8
9   struct edge {
10      int to,next,w;
```

```
11  }e[MAXN<<1];
12
13  int tot,head[MAXN];
14
15  void add(int x,int y,int z) {
16      tot++;
17      e[tot].to=y;
18      e[tot].w=z;
19      e[tot].next=head[x];
20      head[x]=tot;
21  }
22
23  struct node {
24      int u;
25      LL dis;
26      bool operator<(const node& y) const{
27          return dis>y.dis;
28      }
29  };
30
31  LL dist[MAXN];
32  bool flag[MAXN];
33  priority_queue<node> Q;
34
35  void dij(int n,int s) {
36      memset(flag+1,0,n*sizeof(flag[0]));
37      memset(dist+1,0x3f,n*sizeof(dist[0]));
38      dist[s]=0;
39      Q.push({s,0LL});
40      while(!Q.empty()) {
41          while(!Q.empty() && flag[Q.top().u]) Q.pop();
42          if(Q.empty()) break;
43          int u=Q.top().u;
44          Q.pop();
45          flag[u]=1;
46          for(int p=head[u];p;p=e[p].next) {
47              int v=e[p].to,w=e[p].w;
48              if(flag[v]) continue;
49              if(dist[u]+w<dist[v]) {
50                  dist[v]=dist[u]+w;
51                  Q.push({v,dist[v]});
52              }
53          }
54      }
55  }
56
57  void solve() {
58      int n,m,s;
59      scanf("%d %d %d", &n, &m, &s);
60      tot=0;
61      memset(head+1,0,n*sizeof(head[0]));
62      for(int i=1;i<=m;i++) {
63          int f,g,w;
```

```
64          scanf("%d %d %d", &f, &g, &w);
65          add(f,g,w);
66      }
67      dij(n,s);
68      for(int i=1;i<=n;i++)
69          printf("%lld%c", dist[i], " \n"[i==n]);
70  }
71
72  int main() {
73      int T=1,cas=1;(void)(cas);
74      // scanf("%d", &T);
75      while(T--) {
76          // printf("Case #%d: ", cas++);
77          solve();
78      }
79      return 0;
80  }
```

1.4 最短路 SPFA

```
1   #include <cstdio>
2   #include <cstring>
3   #include <queue>
4
5   #define MAXN 100005
6   #define MAXM 500005
7   const int inf=0x3f3f3f3f;
8
9   using namespace std;
10
11  struct edge{
12      int v,to,next;
13  }e[MAXM];
14
15  int n,m,p,tot,head[MAXN],dist[MAXN];
16  bool flag[MAXN];
17
18  void add(int x,int y,int z){
19      tot++;
20      e[tot].v=z;
21      e[tot].to=y;
22      e[tot].next=head[x];
23      head[x]=tot;
24  }
25
26  void spfa(int x){
27      queue <int> Q;
28      memset(dist+1,inf,n*sizeof(dist[0]));
29      memset(flag+1,0,n*sizeof(flag[0]));
30
31      Q.push(x);
32      flag[x]=true;
```

```
33      dist[x]=0;
34      while(!Q.empty()){
35          int u=Q.front();
36          Q.pop();
37          flag[u]=false;
38          for(int q=head[u];q;q=e[q].next){
39              int v=e[q].to;
40              if(dist[u]+e[q].v<dist[v]){
41                  dist[v]=dist[u]+e[q].v;
42                  if(!flag[v]){
43                      Q.push(v);
44                      flag[v]=true;
45                  }
46              }
47          }
48      }
49  }
50
51  int main(){
52      scanf("%d %d %d",&n,&m,&p);
53      tot=0;
54      memset(head+1,0,n*sizeof(head[0]));
55      for(int i=1;i<=m;i++){
56          int f,g,w;
57          scanf("%d %d %d", &f, &g, &w);
58          add(f,g,w);
59      }
60      spfa(p);
61      for(int i=1;i<=n;i++)
62          printf("%d ", dist[i]==inf?2147483647:dist[i]);
63      return 0;
64  }
```

1.5 最近公共祖先 LCA_Doubling

```
1   #include <cstdio>
2   #include <cstring>
3   #include <algorithm>
4   #define MAXN 500005
5   #define MAXM 500005
6   #define MAXLN 25
7   using namespace std;
8
9   struct edge{
10      int to,next;
11  }e[MAXM<<1];
12
13  int tot,head[MAXN];
14
15  void add(int x,int y){
16      tot++;
17      e[tot].to=y;
```

```
18      e[tot].next=head[x];
19      head[x]=tot;
20  }
21
22  int dep[MAXN],st[MAXN][MAXLN];
23
24  void dfs(int cur,int fa){
25      dep[cur]=dep[fa]+1;
26      st[cur][0]=fa;
27      for(int l=1;(1<<l)<=dep[cur];l++)
28          st[cur][l]=st[st[cur][l-1]][l-1];
29
30      for(int p=head[cur];p;p=e[p].next){
31          if(e[p].to==fa) continue;
32          dfs(e[p].to,cur);
33      }
34  }
35
36  int lca(int x,int y){
37      if(dep[x]<dep[y]) swap(x,y);
38      for(int i=0;dep[x]-dep[y];i++)
39          if((dep[x]-dep[y])&(1<<i)) x=st[x][i];
40      if(x==y) return x;
41
42      int l;
43      for(l=1;(1<<l)<=dep[x];l++);
44      for(int i=l;i>=0;i--)
45          if(st[x][i]!=st[y][i])
46              x=st[x][i], y=st[y][i];
47      return st[x][0];
48  }
49
50  int n,m,s;
51
52  int main(){
53      scanf("%d %d %d", &n, &m, &s);
54      tot=0;
55      memset(head+1,0,n*sizeof(head[0]));
56      for(int i=1;i<n;i++){
57          int f,g;
58          scanf("%d %d", &f, &g);
59          add(f,g);
60          add(g,f);
61      }
62      dep[0]=0;
63      dfs(s,0);
64      for(int i=1;i<=m;i++){
65          int f,g;
66          scanf("%d %d", &f, &g);
67          printf("%d\n", lca(f,g));
68      }
69      return 0;
70  }
```

1.6 最近公共祖先 LCA_Tarjan

```cpp
#include <cstdio>
#define MAXN 500005
#define MAXM 500005

struct edge{
    int to,next;
}e[MAXM<<1],eq[MAXM<<1];

int n,m,s,tot,totq,head[MAXN],headq[MAXN],fa[MAXN],ans[MAXM];

void add(int x, int y){
    tot++;
    e[tot].to=y;
    e[tot].next=head[x];
    head[x]=tot;
}

void addq(int x, int y){
    tot++;
    eq[tot].to=y;
    eq[tot].next=headq[x];
    headq[x]=tot;
}

int find(int x){
    if(fa[x]==x)
        return x;
    return fa[x]=find(fa[x]);
}

void tarjan(int x){
    fa[x]=x;
    for(int p=head[x];p;p=e[p].next){
        int u=e[p].to;
        if(!fa[u]){
            tarjan(u);
            fa[u]=x;
        }
    }
    for(int p=headq[x];p;p=eq[p].next){
        int u=eq[p].to;
        if(fa[u]){
            ans[(p+1)>>1]=find(u);
        }
    }
}

int main(){
    scanf("%d %d %d", &n, &m, &s);
    for(int i=1;i<=n;i++){
        head[i]=0;
```

11

```
52        headq[i]=0;
53    }
54    tot=0;
55    for(int i=1;i<n;i++){
56        int f,g;
57        scanf("%d %d",&f,&g);
58        add(f,g);
59        add(g,f);
60    }
61    tot=0;
62    for(int i=1;i<=m;i++){
63        int f,g;
64        scanf("%d %d",&f,&g);
65        addq(f,g);
66        addq(g,f);
67    }
68    for(int i=1;i<=n;i++){
69        fa[i]=0;
70    }
71    tarjan(s);
72    for(int i=1;i<=m;i++)
73        printf("%d\n", ans[i]);
74    return 0;
75 }
```

1.7 判断负环 SPFA_Negtive_Cycle

```
1  #include <cstdio>
2  #include <cstring>
3  #include <queue>
4
5  #define MAXN 100005
6  #define MAXM 500005
7  const int inf=0x3f3f3f3f;
8
9  using namespace std;
10
11 struct edge{
12     int v,to,next;
13 }e[MAXM];
14
15 int n,m,tot,head[MAXN];
16 int dist[MAXN],cnt[MAXN];
17 bool flag[MAXN];
18
19 void add(int x,int y,int z){
20     tot++;
21     e[tot].v=z;
22     e[tot].to=y;
23     e[tot].next=head[x];
24     head[x]=tot;
25 }
```

```
26
27  bool spfa(int x){//负环return false
28      queue <int> Q;
29      memset(dist+1,inf,n*sizeof(dist[0]));
30      memset(flag+1,0,n*sizeof(flag[0]));
31      memset(cnt+1,0,n*sizeof(cnt[0]));
32
33      Q.push(x);
34      flag[x]=true;
35      dist[x]=0;
36      while(!Q.empty()){
37          int u=Q.front();
38          Q.pop();
39          flag[u]=false;
40          for(int q=head[u];q;q=e[q].next){
41              int v=e[q].to;
42              if(dist[u]+e[q].v<dist[v]){
43                  dist[v]=dist[u]+e[q].v;
44                  if(!flag[v]) {
45                      Q.push(v);
46                      flag[v]=true;
47                      if(++cnt[v]>=n){
48                          return false;
49                      }
50                  }
51              }
52          }
53      }
54      return true;
55  }
56
57  void solve() {
58      scanf("%d %d",&n,&m);
59      tot=0;
60      memset(head+1,0,n*sizeof(head[0]));
61      for(int i=1;i<=m;i++){
62          int f,g,w;
63          scanf("%d %d %d", &f, &g, &w);
64          add(f,g,w);
65          if(w>=0) add(g,f,w);
66      }
67      if(spfa(1)) printf("NO\n");
68      else printf("YES\n");
69  }
70
71  int main(){
72      int T=1;
73      scanf("%d", &T);
74      while(T--) {
75          solve();
76      }
77      return 0;
78  }
```

1.8 拓扑排序 Topological_Sort_Khan

```cpp
#include <cstdio>
#include <cstring>
#include <stack>
#define MAXN 100005
#define MAXM 200005

using namespace std;

struct edge{
    int to,next;
}e[MAXM];

int n,m,tot,head[MAXN],indgr[MAXN],list[MAXN];
bool flag[MAXN];

void add(int x,int y){
    tot++;
    e[tot].to=y;
    e[tot].next=head[x];
    head[x]=tot;
}

void khan(){
    stack<int> s;
    int cnt=0;
    memset(flag+1,0,n*sizeof(flag[0]));
    for(int i=1;i<=n;i++){
        if(!flag[i] && indgr[i]==0){
            s.push(i);
            flag[i]=true;
            while(!s.empty()){
                int u=s.top();s.pop();
                list[++cnt]=u;
                flag[u]=true;
                for(int p=head[u];p;p=e[p].next){
                    int v=e[p].to;
                    indgr[v]--;
                    if(indgr[v]==0)
                        s.push(v);
                }
            }
        }
    }
}

int main(){
    scanf("%d %d",&n,&m);
    memset(head+1,0,n*sizeof(head[0]));
    memset(indgr+1,0,n*sizeof(indgr[0]));
    tot=0;
    for(int i=1;i<=m;i++){
```

```
52        int f,g;
53        scanf("%d %d", &f, &g);
54        add(f,g);
55        indgr[g]++;
56    }
57    khan();
58    for(int i=1;i<=n;i++)
59        printf("%d ", list[i]);
60    return 0;
61 }
```

2 多项式

2.1 FFT 字符串匹配 String_Match_FFT

```cpp
#include <cstdio>
#include <cstring>
#include <cmath>
#include <algorithm>
#include <complex>
#define MAXN 300005
#define MAXL 550005

using namespace std;

const double PI=acos(-1);
complex<double> omg[MAXL],iomg[MAXL];

void init(int n){
    for(int i=0;i<n;i++){
        omg[i]=polar(1.0,2.0*PI*i/n);
        iomg[i]=conj(omg[i]);
    }
}

void FFT(int n,complex<double>* P,complex<double>* w){
    for(int i=0,j=0;i<n;i++){
        if(i<j) swap(P[i],P[j]);
        for(int l=n>>1;(j^=l)<l;l>>=1);
    }

    for(int i=2,l;i<=n;i<<=1){
        l=i>>1;
        complex<double> t;
        for(int j=0;j<n;j+=i){
            for(int k=0;k<l;k++){
                t=P[j+l+k]*w[n/i*k];
                P[j+l+k]=P[j+k]-t;
                P[j+k]=P[j+k]+t;
            }
        }
    }
}

int n,m,lim;
char s1[MAXN],s2[MAXN];
complex<double> A[MAXL],B[MAXL],tA[MAXL],tB[MAXL],ans[MAXL];

int main(){
    scanf("%d %d", &m, &n);
    scanf("%s %s",s1,s2);
    for(int i=0;i<m;i++) A[m-i-1]=(s1[i]=='*'?0:s1[i]-'a'+1);
    for(int i=0;i<n;i++) B[i]=(s2[i]=='*'?0:s2[i]-'a'+1);
    for(lim=1;lim<n-1;lim<<=1);
```

```
50    init(lim);
51    fill(ans,ans+lim,complex<double>(0.0,0.0));
52
53    for(int i=0;i<m;i++) tA[i]=A[i]*A[i]*A[i];
54    for(int i=0;i<n;i++) tB[i]=B[i];
55    fill(tA+m,tA+lim,complex<double>(0.0,0.0));
56    fill(tB+n,tB+lim,complex<double>(0.0,0.0));
57    FFT(lim,tA,omg);
58    FFT(lim,tB,omg);
59    for(int i=0;i<lim;i++) ans[i]+=tA[i]*tB[i];
60
61    for(int i=0;i<m;i++) tA[i]=A[i];
62    for(int i=0;i<n;i++) tB[i]=B[i]*B[i]*B[i];
63    fill(tA+m,tA+lim,complex<double>(0.0,0.0));
64    fill(tB+n,tB+lim,complex<double>(0.0,0.0));
65    FFT(lim,tA,omg);
66    FFT(lim,tB,omg);
67    for(int i=0;i<lim;i++) ans[i]+=tA[i]*tB[i];
68
69    for(int i=0;i<m;i++) tA[i]=A[i]*A[i];
70    for(int i=0;i<n;i++) tB[i]=B[i]*B[i];
71    fill(tA+m,tA+lim,complex<double>(0.0,0.0));
72    fill(tB+n,tB+lim,complex<double>(0.0,0.0));
73    FFT(lim,tA,omg);
74    FFT(lim,tB,omg);
75    for(int i=0;i<lim;i++) ans[i]-=complex<double>(2.0,0)*tA[i]*tB[i];
76
77    FFT(lim,ans,iomg);
78
79    int cnt=0;
80    for(int i=m-1;i<n;i++)
81        if((int)(ans[i].real()/lim+0.5)==0) cnt++;
82    printf("%d\n", cnt);
83    for(int i=m-1;i<n;i++)
84        if((int)(ans[i].real()/lim+0.5)==0) printf("%d ", i-m+2);
85    return 0;
86 }
```

2.2 FFT 递归 Fast_Fourier_Transform_Cooley-Tukey_Recursion

```
1  #include <cstdio>
2  #include <algorithm>
3  #include <complex>
4  #define MAXN 4000005
5
6  using namespace std;
7
8  complex<double> omg[MAXN],iomg[MAXN],temp[MAXN];
9
10 void init(int n){
11     double PI=acos(-1);
12     for(int i=0;i<n;i++){
```

```
13          omg[i]=polar(1.0,2.0*PI*i/n);
14          iomg[i]=conj(omg[i]);
15      }
16  }
17
18  void FFT(int n,complex<double>* buffer,int offset,int step,complex<double>* omg){
19      if(n==1) return;
20      int m=n>>1;
21      FFT(m,buffer,offset,step<<1,omg);
22      FFT(m,buffer,offset+step,step<<1,omg);
23      for(int i=0;i<m;i++){
24          int pos=2*i*step;
25          temp[i]=buffer[offset+pos]+omg[i*step]*buffer[offset+step+pos];
26          temp[i+m]=buffer[offset+pos]-omg[i*step]*buffer[offset+step+pos];
27      }
28      for(int i=0;i<n;i++)
29          buffer[offset+i*step]=temp[i];
30  }
31
32  int n,m,lim;
33  complex<double> A[MAXN],B[MAXN];
34
35  int main(){
36      scanf("%d %d", &n, &m);
37      for(lim=1;lim<=n+m;lim<<=1);
38      fill(A,A+lim,complex<double>(0.0,0.0));
39      fill(B,B+lim,complex<double>(0.0,0.0));
40      for(int i=0,t;i<n+1;i++){
41          scanf("%d", &t);
42          A[i]+=t;
43      }
44      for(int i=0,t;i<m+1;i++){
45          scanf("%d", &t);
46          B[i]+=t;
47      }
48      init(lim);
49      FFT(lim,A,0,1,omg);
50      FFT(lim,B,0,1,omg);
51      for(int i=0;i<lim;i++)
52          A[i]=A[i]*B[i];
53      FFT(lim,A,0,1,iomg);
54      for(int i=0;i<n+m+1;i++)
55          printf("%d ", (int)(A[i].real()/lim+0.5));
56      return 0;
57  }
```

2.3   FFT 递推 Fast_Fourier_Transform_Cooley-Tukey_Iteration

```
1  #include <bits/stdc++.h>
2  #define MAXN 2100005
3
4  using namespace std;
```

```
5
6   const double PI=acos(-1);
7
8   struct Complex{
9       double real,image;
10      Complex operator+(Complex y)const{
11          return {real+y.real,image+y.image};
12      }
13      Complex operator-(Complex y)const{
14          return {real-y.real,image-y.image};
15      }
16      Complex operator*(Complex y)const{
17          return {real*y.real-image*y.image,real*y.image+image*y.real};
18      }
19  };
20
21  void FFT(int n,Complex* P,int f){
22      for(int i=0,j=0;i<n;i++){
23          if(i<j) swap(P[i],P[j]);
24          for(int l=n>>1;(j^=l)<l;l>>=1);
25      }
26
27      for(int i=2,l;i<=n;i<<=1){
28          l=i>>1;
29          Complex wn={cos(2*PI/i),f*sin(2*PI/i)};
30          for(int j=0;j<n;j+=i){
31              Complex w={1,0};
32              for(int k=0;k<l;k++,w=w*wn){
33                  Complex t=P[j+l+k]*w;
34                  P[j+l+k]=P[j+k]-t;
35                  P[j+k]=P[j+k]+t;
36              }
37          }
38      }
39  }
40
41  int n,m,lim;
42  Complex A[MAXN],B[MAXN];
43
44  int main(){
45      scanf("%d %d", &n, &m);
46      for(lim=1;lim<=n+m;lim<<=1);
47      memset(A,0,lim*sizeof(A[0]));
48      memset(B,0,lim*sizeof(B[0]));
49      for(int i=0,t;i<n+1;i++){
50          scanf("%d", &t);
51          A[i]={1.0*t,0};
52      }
53      for(int i=0,t;i<m+1;i++){
54          scanf("%d", &t);
55          B[i]={1.0*t,0};
56      }
57      FFT(lim,A,1);
```

19

```
58      FFT(lim,B,1);
59      for(int i=0;i<lim;i++)
60          A[i]=A[i]*B[i];
61      FFT(lim,A,-1);
62      for(int i=0;i<n+m+1;i++)
63          printf("%d ", (int)(A[i].real/lim+0.5));
64      return 0;
65  }
```

2.4 NTT 递推 Number_Theoretic_Transforms

```
1  #include <cstdio>
2  #include <algorithm>
3  using namespace std;
4  #define LL long long
5  const int MAXN=2100005;
6  const int MOD=998244353;
7  const int RT=3;
8
9  LL binpow(LL x,LL y,LL mod){
10     LL r=1%mod;
11     while(y){
12         if(y&1) r=(r*x)%mod;
13         x=(x*x)%mod;
14         y>>=1;
15     }
16     return r;
17 }
18
19 void NTT(int n,int* P,int f){
20     for(int i=0,j=0;i<n;i++){
21         if(i<j) swap(P[i],P[j]);
22         for(int l=n>>1;(j^=l)<l;l>>=1);
23     }
24
25     for(int i=2,l;i<=n;i<<=1){
26         l=i>>1;
27         int wn=binpow(RT,(MOD-1)/i,MOD);
28         if(f==-1) wn=binpow(wn,MOD-2,MOD);
29         for(int j=0;j<n;j+=i){
30             int w=1;
31             for(int k=0;k<l;k++,w=1LL*w*wn%MOD){
32                 int t=1LL*P[j+l+k]*w%MOD;
33                 P[j+l+k]=1LL*(P[j+k]-t+MOD)%MOD;
34                 P[j+k]=1LL*(P[j+k]+t)%MOD;
35             }
36         }
37     }
38 }
39
40 int A[MAXN],B[MAXN];
41
```

```
42  int main(){
43      int n,m;
44      scanf("%d %d", &n, &m);
45      int lim;
46      for(lim=1;lim<=n+m;lim<<=1);
47      for(int i=0;i<n+1;i++){
48          scanf("%d", &A[i]);
49      }
50      for(int i=0;i<m+1;i++){
51          scanf("%d", &B[i]);
52      }
53      NTT(lim,A,1);
54      NTT(lim,B,1);
55      for(int i=0;i<lim;i++)
56          A[i]=1LL*A[i]*B[i]%MOD;
57      NTT(lim,A,-1);
58      int invn=binpow(lim,MOD-2,MOD);
59      for(int i=0;i<n+m+1;i++)
60          printf("%lld ", 1LL*A[i]*invn%MOD);
61      return 0;
62  }
```

2.5 多项式求逆 Polynomial_Inverse

```
1   #include <cstdio>
2   #include <algorithm>
3   #define LL long long
4   #define MAXN 270005
5   #define MOD 998244353
6   #define RT 3
7
8   using namespace std;
9
10  LL omg[MAXN],iomg[MAXN];
11
12  LL binpow(LL x,LL y,LL mod){
13      LL r=1%mod;
14      while(y){
15          if(y&1) r=(r*x)%mod;
16          x=(x*x)%mod;
17          y>>=1;
18      }
19      return r;
20  }
21
22  void init(int n){
23      omg[0]=iomg[0]=1;
24      omg[1]=binpow(RT,(MOD-1)/n,MOD);
25      iomg[1]=binpow(omg[1],MOD-2,MOD);
26      for(int i=2;i<n;i++){
27          omg[i]=omg[i-1]*omg[1]%MOD;
28          iomg[i]=iomg[i-1]*iomg[1]%MOD;
```

```
29         }
30  }
31
32  void NTT(int n,LL* P,LL* w){
33      for(int i=0,j=0;i<n;i++){
34          if(i<j) swap(P[i],P[j]);
35          for(int l=n>>1;(j^=l)<l;l>>=1);
36      }
37
38      for(int i=2,l;i<=n;i<<=1){
39          l=i>>1;
40          for(int j=0;j<n;j+=i){
41              for(int k=0;k<l;k++){
42                  LL t=P[j+l+k]*w[n/i*k]%MOD;
43                  P[j+l+k]=(P[j+k]-t+MOD)%MOD;
44                  P[j+k]=(P[j+k]+t)%MOD;
45              }
46          }
47      }
48  }
49
50  void poly_inv(int dgr,LL* X,LL* Y){
51      if(dgr==1){
52          Y[0]=binpow(X[0],MOD-2,MOD);
53      }
54      else{
55          poly_inv((dgr+1)>>1,X,Y);
56
57          static LL Z[MAXN];
58          int lim;
59          for(lim=1;lim<(dgr<<1);lim<<=1);
60          copy(X,X+dgr,Z);
61          fill(Z+dgr,Z+lim,0);
62          init(lim);
63
64          NTT(lim,Z,omg);
65          NTT(lim,Y,omg);
66          for(int i=0;i<lim;i++)
67              Y[i]=(2-Z[i]*Y[i]%MOD+MOD)*Y[i]%MOD;
68          NTT(lim,Y,iomg);
69          LL invlim=binpow(lim,MOD-2,MOD);
70          for(int i=0;i<dgr;i++)
71              Y[i]=Y[i]*invlim%MOD;
72          fill(Y+dgr,Y+lim,0);
73      }
74  }
75
76  int n;
77  LL A[MAXN],B[MAXN];
78
79  int main(){
80      scanf("%d", &n);
81      for(int i=0;i<n;i++)
```

```
82        scanf("%lld", &A[i]);
83    poly_inv(n,A,B);
84    for(int i=0;i<n;i++)
85        printf("%lld ", B[i]);
86    return 0;
87 }
```

3 字符串

3.1 字符串哈希 String_Hash

```cpp
#include <cstdio>
#include <string>

#define BASE 307
#define MOD 5555567

int hsh(string x){
    int h=0,len=x.length();
    for(int i=0;i<len;i++){
        h=(h*BASE+x[i])%MOD;
    }
    return h;
}

int main(){
    return 0;
}
```

3.2 马拉车 Manacher

```cpp
#include <cstdio>
#include <cstring>
#include <algorithm>
#define MAXN 11000005
using namespace std;

int r[MAXN<<1],mx;
char st[MAXN<<1];

void manacher(char *s){
    int len=strlen(s);
    st[0]='$';
    for(int i=0;i<len;i++){
        st[i<<1|1]='#';
        st[(i+1)<<1]=s[i];
    }
    len=len<<1|1;
    st[len]='#';
    st[len+1]='*';
    r[1]=1;
    mx=0;
    for(int i=2,mid=1;i<=len;i++){
        r[i]=min(mid+r[mid]-i,r[2*mid-i]);
        for(;st[i-r[i]]==st[i+r[i]];r[i]++);
        if(i+r[i]>mid+r[mid]) mid=i;
        mx=max(mx,r[i]-1);
    }
}
```

```
29
30  char s[MAXN];
31
32  int main(){
33      scanf("%s",s);
34      manacher(s);
35      printf("%d\n", mx);
36      return 0;
37  }
```

3.3 字符串匹配 KMP

```
1   #include <bits/stdc++.h>
2   using namespace std;
3   #define MAXN 1000005
4
5   int kmp(char *s1, char *s2, int *nxt){
6       int n=strlen(s1+1),m=strlen(s2+1);
7
8       nxt[0]=-1;
9       for(int i=1,k=-1;i<=m;i++){
10          //k初始值为-1. nxt数组可以查询boarder的boarder.
11          while(~k && s2[k+1]!=s2[i])
12              k=nxt[k];
13          nxt[i]=++k;
14      }
15
16      int cnt=0;
17      for(int i=1,k=0;i<=n;i++){
18          //匹配串前缀与模式串后缀比，上一位前缀的boarder的boarder也能匹配。
19          while(~k && s2[k+1]!=s1[i])
20              k=nxt[k];
21          if(m==++k){
22              cnt++;
23              printf("%d\n", i-m+1);
24          }
25      }
26      return cnt;
27  }
28
29  char s1[MAXN],s2[MAXN];
30  int nxt[MAXN];
31
32  int main(){
33      scanf("%s %s", s1+1, s2+1);
34      kmp(s1,s2,nxt);
35      int len=strlen(s2+1);
36      for(int i=1;i<=len;i++){
37          printf("%d ", nxt[i]);
38      }
39      return 0;
40  }
```

3.4 AC 自动机 AC-Automaton

```cpp
#include <cstdio>
#include <cstring>
#include <map>
#include <queue>
#define MAXN 1000006
using namespace std;

struct trie{
    int fail,mark,ch[26];
}tt[MAXN];

int tot;

void insert(char *s){
    int len=strlen(s+1), cur=0;
    for(int i=1;i<=len;i++){
        int& next=tt[cur].ch[s[i]-'a'];
        if(!next){
            next=++tot;
            tt[tot].mark=0;
        }
        cur=next;
    }
    tt[cur].mark++;
}

void getfail(){
    queue<int> Q;
    tt[0].fail=0;
    tt[0].mark=0;
    for(int i=0;i<26;i++)
        if(tt[0].ch[i]) tt[tt[0].ch[i]].fail=0, Q.push(tt[0].ch[i]);
    while(!Q.empty()){
        int u=Q.front(); Q.pop();
        for(int i=0;i<26;i++){
            if(!tt[u].ch[i]) continue;
            int k=tt[u].fail;
            while(k && !tt[k].ch[i])
                k=tt[k].fail;
            tt[tt[u].ch[i]].fail=tt[k].ch[i];
            // tt[tt[u].ch[i]].mark+=tt[tt[k].ch[i]].mark;//如果需要重复统计，fail累加标记
            Q.push(tt[u].ch[i]);
        }
    }
}

int query(char *s){
    int len=strlen(s+1),ans=0;
    for(int i=1,k=0;i<=len;i++){
        while(k && !tt[k].ch[s[i]-'a'])
            k=tt[k].fail;
```

```
52        k=tt[k].ch[s[i]-'a'];
53        ans+=tt[k].mark;
54
55        tt[k].mark=0;//清除该字符串的标记（只求
56    }
57    return ans;
58 }
59
60 int n;
61 char s[MAXN];
62
63 int main(){
64    scanf("%d", &n);
65    tot=0;
66    memset(tt,0,sizeof(tt));
67    for(int i=1;i<=n;i++){
68        scanf("%s", s+1);
69        insert(s);
70    }
71    scanf("%s", s+1);
72    getfail();
73    printf("%d\n", query(s));
74    return 0;
75 }
```

3.5 后缀数组 Suffix__Array

```
1  #include <cstdio>
2  #include <cstring>
3  #include <algorithm>
4  #define MAXN 1000005
5  using namespace std;
6
7  //sa:排名对应的前缀，rk:前缀的排名，tp:第二关键字排名对应的前缀，tax:排名对应的个数
8  //height:排名i与i-1后缀的LCP(最长公共前缀)
9  int sa[MAXN],r1[MAXN],r2[MAXN],tax[MAXN],height[MAXN];
10 int *rk=r1,*tp=r2;
11 char s[MAXN];
12
13 void rsort(int n,int m){
14    memset(tax,0,(m+1)*sizeof(tax[0]));
15    for(int i=1;i<=n;i++) tax[rk[i]]++;//当前排名装桶
16    for(int i=1;i<=m;i++) tax[i]+=tax[i-1];//计算桶的名次
17    for(int i=n;i>=1;i--) sa[tax[rk[tp[i]]]--]=tp[i];//按照第二关键字降序，分配排名。
18 }
19
20 void get_sa(char* s){
21    //O(nlogn)
22    int n=strlen(s+1), m=0;
23    for(int i=1;i<=n;i++)
24        m=max(m,rk[i]=s[i]),tp[i]=i;
25    rsort(n,m);
```

```
26    for(int k=1,p=0;p<n;k<<=1,m=p){
27        p=0;
28        //重制第二关键字
29        for(int i=n-k+1;i<=n;i++) tp[++p]=i; //后续为空，排前面
30        for(int i=1;i<=n;i++) if(sa[i]>k) tp[++p]=sa[i]-k; //按照第一关键字排第二关键字
31
32        rsort(n,m);
33
34        swap(tp,rk);
35        rk[sa[1]]=p=1;
36        for(int i=2;i<=n;i++){
37            rk[sa[i]]=(tp[sa[i]]==tp[sa[i-1]] && tp[sa[i]+k]==tp[sa[i-1]+k])?p:++p;
38        }
39    }
40
41    //利用height[rk[i+1]]>=height[rk[i]]-1
42    //O(n)
43    for(int i=1,k=0;i<=n;i++){
44        if(k) k--;
45        while(rk[i]>1 && s[i+k]==s[sa[rk[i]-1]+k]) k++;
46        height[rk[i]]=k;
47    }
48 }
49
50 int main(){
51    scanf("%s",s+1);
52    get_sa(s);
53    int len=strlen(s+1);
54    for(int i=1;i<=len;i++)
55        printf("%d ", sa[i]);
56    return 0;
57 }
```

3.6 后缀自动机 Suffix-Automaton

```
1  #include <bits/stdc++.h>
2  #define MAXN 1000005
3  #define LL long long
4  using namespace std;
5
6  struct SAM {
7      int len,link,cnt;
8      int ch[26];
9  }sam[MAXN<<1];
10
11 int sz,last;
12
13 void sam_init() {
14     sam[0].len=0;
15     sam[0].link=-1;
16     sam[0].cnt=0;
17     memset(sam[0].ch,0,sizeof(sam[0].ch));
```

```
18      sz=0;
19      last=0;
20  }
21
22  void sam_extend(int c) {
23      int cur=++sz;
24      sam[cur].len=sam[last].len+1;
25      memset(sam[cur].ch,0,sizeof(sam[cur].ch));
26
27      int p=last;
28      for(;~p && !sam[p].ch[c];p=sam[p].link)
29          sam[p].ch[c]=cur;
30
31      if(!~p) {
32          sam[cur].link=0;
33      } else {
34          int q=sam[p].ch[c];
35          if(sam[p].len+1==sam[q].len) {
36              sam[cur].link=q;
37          } else {
38              int clone=++sz;
39              sam[clone]=sam[q];
40              sam[clone].len=sam[p].len+1;
41              sam[clone].cnt=0;
42              sam[q].link=sam[cur].link=clone;
43              for(;~p && sam[p].ch[c]==q;p=sam[p].link)
44                  sam[p].ch[c]=clone;
45          }
46      }
47
48      last=cur;
49      sam[cur].cnt=1;
50  }
51
52  struct edge{
53      int to,next;
54  }e[MAXN<<1];
55
56  int tot,head[MAXN<<1];
57
58  void add(int x,int y) {
59      tot++;
60      e[tot].to=y;
61      e[tot].next=head[x];
62      head[x]=tot;
63  }
64
65  char s[MAXN];
66  LL ans;
67
68  void dfs(int x) {
69      for(int p=head[x];p;p=e[p].next) {
70          int u=e[p].to;
```

```
71          dfs(u);
72          sam[x].cnt+=sam[u].cnt;
73      }
74      if(sam[x].cnt!=1) ans=max(ans,1LL*sam[x].len*sam[x].cnt);
75  }
76
77  void solve() {
78      scanf("%s", s);
79      int len=strlen(s);
80      sam_init();
81      for(int i=0;i<len;i++)
82          sam_extend(s[i]-'a');
83      tot=0;
84      memset(head,0,(sz+1)*sizeof(head[0]));
85      for(int i=1;i<=sz;i++)
86          add(sam[i].link,i);
87      ans=0;
88      dfs(0);
89      printf("%lld\n", ans);
90  }
91
92  int main() {
93      int T=1,cas=1;
94      // scanf("%d", &T);
95      while(T--) {
96          // printf("Case #%d: ", cas++);
97          solve();
98      }
99      return 0;
100 }
```

3.7 广义后缀自动机 General_Suffix-Automaton

```
1   #include <bits/stdc++.h>
2   #define MAXN 1000005
3   #define LL long long
4   using namespace std;
5   // const int inf=0x3f3f3f3f;
6
7   struct SAM {
8       int len,link,cnt;
9       int ch[26];
10  }sam[MAXN<<1];
11
12  int sz,last;
13
14  void sam_init() {
15      sam[0].len=0;
16      sam[0].link=-1;
17      sam[0].cnt=0;
18      memset(sam[0].ch,0,sizeof(sam[0].ch));
19      sz=0;
```

```
20        last=0;
21   }
22
23   void sam_extend(int c) {
24       if(sam[last].ch[c] && sam[last].len+1==sam[sam[last].ch[c]].len) {
25           last=sam[last].ch[c];
26           sam[last].cnt++;
27           return;
28       }
29
30       int cur=++sz;
31       sam[cur].len=sam[last].len+1;
32       memset(sam[cur].ch,0,sizeof(sam[cur].ch));
33
34       int p=last;
35       for(;~p && !sam[p].ch[c];p=sam[p].link)
36           sam[p].ch[c]=cur;
37
38       if(!~p) {
39           sam[cur].link=0;
40       } else {
41           int q=sam[p].ch[c];
42           if(sam[p].len+1==sam[q].len) {
43               sam[cur].link=q;
44           } else {
45               int clone;
46               if(p==last) {
47                   clone=cur;
48               } else {
49                   clone=++sz;
50                   sam[cur].link=clone;
51               }
52
53               sam[clone]=sam[q];
54               sam[clone].len=sam[p].len+1;
55               sam[q].link=clone;
56               sam[clone].cnt=0;
57               for(;~p && sam[p].ch[c]==q;p=sam[p].link)
58                   sam[p].ch[c]=clone;
59           }
60       }
61
62       last=cur;
63       sam[cur].cnt=1;
64   }
65
66   char s[MAXN];
67
68   void solve() {
69       int n;
70       scanf("%d", &n);
71       sam_init();
72       for(int i=1;i<=n;i++) {
```

```
73        scanf("%s", s);
74        int len=strlen(s);
75        last=0;
76        for(int j=0;j<len;j++) {
77            sam_extend(s[j]-'a');
78        }
79    }
80
81    LL ans=0;
82    for(int i=1;i<=sz;i++) {
83        ans+=sam[i].len-sam[sam[i].link].len;
84    }
85
86    printf("%lld\n", ans);
87 }
88
89 int main() {
90    int T=1,cas=1;
91    // scanf("%d", &T);
92    while(T--) {
93        // printf("Case #%d: ", cas++);
94        solve();
95    }
96    return 0;
97 }
```

4 数据结构

4.1 并查集 Disjoin_Set_Union

```cpp
#include <cstdio>
#define maxn 10005

int n,m,a,b,c,fa[maxn];

void ini(){
    for(int i=1;i<=n;i++){
        fa[i]=i;
    }
}

int find(int x){
    if(fa[x] == x)
        return x;
    return fa[x]=find(fa[x]);
}

void join(int x, int y){
    fa[find(x)]=find(y);
}

int main(){
    scanf("%d %d", &n, &m);
    ini();
    for(int i=1; i<=m; i++){
        scanf("%d %d %d", &a, &b, &c);
        if(a==1)
            join(b,c);
        else if(find(b)==find(c))
            puts("Y");
        else puts("N");
    }
    return 0;
}
```

4.2 可撤销并查集 Disjoin_Set_Union_Withdrawable

```cpp
struct DSU{
    int fa[MAXN],rk[MAXN];
    vector<pair<int*,int>> stk;
    void init(int n) {
        stk.clear();
        for(int i=1;i<=n;i++) fa[i]=i,rk[i]=1;
    }
    int find(int x) {
        if(x==fa[x]) return x;
        return find(fa[x]);
    }
```

```
12    bool join(int x,int y) {
13        int rx=find(x),ry=find(y);
14        if(rx==ry) return false;
15
16        if(rk[rx]>rk[ry]) swap(rx,ry);
17        stk.emplace_back(fa+rx,rx);
18        fa[rx]=ry;
19        stk.emplace_back(rk+ry,rk[ry]);
20        rk[ry]+=rk[rx];
21        return true;
22    }
23    void withdraw() {
24        *stk.back().first=stk.back().second;
25        stk.pop_back();
26        *stk.back().first=stk.back().second;
27        stk.pop_back();
28    }
29 }dsu;
```

4.3　分块 Block_1

```
1  #include <cstdio>
2  #include <cstring>
3  #include <cmath>
4  #include <algorithm>
5  #define MAXN 50005
6
7  using namespace std;
8
9  int n,blo,v[MAXN],bl[MAXN],atag[MAXN];
10
11 void add(int l,int r,int x){
12     if(bl[l]==bl[r]){
13         for(int i=l;i<=r;i++)
14             v[i]+=x;
15     }
16     else{
17         for(int i=l;i<=bl[l]*blo;i++)
18             v[i]+=x;
19         for(int i=(bl[r]-1)*blo+1;i<=r;i++)
20             v[i]+=x;
21     }
22     for(int i=bl[l]+1;i<=bl[r]-1;i++)
23         atag[i]+=x;
24 }
25
26 int main(){
27     scanf("%d", &n);blo=sqrt(n);
28     for(int i=1;i<=n;i++)
29         scanf("%d", &v[i]);
30     for(int i=1;i<=n;i++)
31         bl[i]=(i-1)/blo+1;
```

```
32      memset(atag+1,0,bl[n]*sizeof(atag[0]));
33      for(int i=1;i<=n;i++){
34          int opt,l,r,c;
35          scanf("%d %d %d %d", &opt, &l, &r, &c);
36          if(opt==0){
37              add(l,r,c);
38          }
39          else printf("%d\n", v[r]+atag[bl[r]]);
40      }
41      return 0;
42  }
```

4.4 分块 Block_2

```
1   #include <cstdio>
2   #include <cstring>
3   #include <cmath>
4   #include <algorithm>
5   #include <vector>
6   #define MAXN 50005
7   #define MAXB 505
8
9   using namespace std;
10
11  int n,blo,v[MAXN],bl[MAXN],atag[MAXB];
12  vector<int> ve[MAXB];
13
14  void reset(int x){
15      ve[x].clear();
16      for(int i=(x-1)*blo+1;i<=min(x*blo,n);i++){
17          ve[x].push_back(v[i]);
18      }
19      sort(ve[x].begin(),ve[x].end());
20  }
21
22  void add(int l,int r,int x){
23      if(bl[l]==bl[r]){
24          for(int i=l;i<=r;i++)
25              v[i]+=x;
26          reset(bl[l]);
27      }
28      else{
29          for(int i=l;i<=bl[l]*blo;i++)
30              v[i]+=x;
31          reset(bl[l]);
32          for(int i=(bl[r]-1)*blo+1;i<=r;i++)
33              v[i]+=x;
34          reset(bl[r]);
35          for(int i=bl[l]+1;i<=bl[r]-1;i++)
36          atag[i]+=x;
37      }
38  }
```

```
39
40  int query(int l,int r,int x){
41      int cnt=0;
42      if(bl[l]==bl[r]){
43          for(int i=l;i<=r;i++)
44              if(v[i]+atag[bl[i]]<x) cnt++;
45      }
46      else{
47          for(int i=l;i<=bl[l]*blo;i++)
48              if(v[i]+atag[bl[i]]<x) cnt++;
49          for(int i=(bl[r]-1)*blo+1;i<=r;i++)
50              if(v[i]+atag[bl[i]]<x) cnt++;
51          for(int i=bl[l]+1;i<=bl[r]-1;i++)
52              cnt+=lower_bound(ve[i].begin(),ve[i].end(),x-atag[i])-ve[i].begin();
53      }
54      return cnt;
55  }
56
57  int main(){
58      scanf("%d", &n);blo=sqrt(n);
59      for(int i=1;i<=n;i++)
60          scanf("%d", &v[i]);
61      for(int i=1;i<=n;i++){
62          bl[i]=(i-1)/blo+1;
63          ve[bl[i]].push_back(v[i]);
64      }
65      memset(atag+1,0,bl[n]*sizeof(atag[0]));
66      for(int i=1;i<=bl[n];i++)
67          sort(ve[i].begin(),ve[i].end());
68
69      for(int i=1;i<=n;i++){
70          int opt,l,r,c;
71          scanf("%d %d %d %d", &opt, &l, &r, &c);
72          if(opt==0){
73              add(l,r,c);
74          }
75          else{
76              printf("%d\n", query(l,r,c*c));
77          }
78      }
79      return 0;
80  }
```

4.5 分块 Block_4

```
1  #include <cstdio>
2  #include <cstring>
3  #include <cmath>
4  #include <algorithm>
5  #define LL long long
6  #define MAXN 50005
7  #define MAXB 505
```

```
8
9  using namespace std;
10
11 int n,blo;
12 LL v[MAXN],bl[MAXN],atag[MAXB],sum[MAXB];
13
14 void add(int l,int r,int x){
15     if(bl[l]==bl[r]){
16         for(int i=l;i<=r;i++)
17             v[i]+=x,sum[bl[i]]+=x;
18     }
19     else{
20         for(int i=l;i<=bl[l]*blo;i++)
21             v[i]+=x,sum[bl[i]]+=x;
22         for(int i=(bl[r]-1)*blo+1;i<=r;i++)
23             v[i]+=x,sum[bl[i]]+=x;
24         for(int i=bl[l]+1;i<=bl[r]-1;i++)
25             atag[i]+=x;
26     }
27 }
28
29 LL query(int l,int r){
30     LL ans=0;
31     if(bl[l]==bl[r]){
32         for(int i=l;i<=r;i++)
33             ans+=v[i]+atag[bl[i]];
34     }
35     else{
36         for(int i=l;i<=bl[l]*blo;i++)
37             ans+=v[i]+atag[bl[i]];
38         for(int i=(bl[r]-1)*blo+1;i<=r;i++)
39             ans+=v[i]+atag[bl[i]];
40         for(int i=bl[l]+1;i<=bl[r]-1;i++)
41             ans+=sum[i]+atag[i]*blo;
42     }
43     return ans;
44 }
45
46 int main(){
47     scanf("%d", &n);blo=sqrt(n);
48     for(int i=1;i<=n;i++)
49         bl[i]=(i-1)/blo+1;
50     memset(atag+1,0,bl[n]*sizeof(atag[0]));
51     memset(sum+1,0,bl[n]*sizeof(sum[0]));
52     for(int i=1;i<=n;i++){
53         scanf("%lld", &v[i]);
54         sum[bl[i]]+=v[i];
55     }
56     for(int i=1;i<=n;i++){
57         int opt,l,r,c;
58         scanf("%d %d %d %d", &opt, &l, &r, &c);
59         if(opt==0){
60             add(l,r,c);
```

```
61          }
62          else printf("%lld\n", query(l,r)%(c+1));
63      }
64      return 0;
65  }
```

4.6   树状数组 Binary_Indexed_Tree

```cpp
1  #include <bits/stdc++.h>
2  #define LL long long
3  using namespace std;
4  const int MAXN=2000005;
5
6  struct BIT {
7      int n,b[MAXN];
8
9      int lowbit(int x){
10         return x&(-x);
11     }
12
13     void change(int x,int y){
14         for(;x<=n;x+=lowbit(x))
15             b[x]+=y;
16     }
17
18     int sum(int x){
19         int s=0;
20         for(;x>0;x-=lowbit(x))
21             s+=b[x];
22         return s;
23     }
24
25     void build(int len,int *a) {
26         n=len;
27         for(int i=1;i<=n;i++)
28             b[i]=a[i];
29         for(int x=1;x<<1<=n;x<<=1)
30             for(int i=x;i+x<=n;i+=x<<1)
31                 b[i+x]+=b[i];
32     }
33 }bit;
34
35 int a[MAXN];
36
37 void solve() {
38     int n,m;
39     scanf("%d %d", &n, &m);
40     for(int i=1;i<=n;i++)
41         scanf("%d", &a[i]);
42     bit.build(n,a);
43     for(int i=1;i<=m;i++) {
44         int opt,x,y;
```

```
45        scanf("%d %d %d", &opt, &x, &y);
46        if(opt==1) {
47            bit.change(x,y);
48        } else {
49            printf("%d\n", bit.sum(y)-bit.sum(x-1));
50        }
51    }
52 }
53
54 int main() {
55    int T=1,cas=1;(void)(cas);
56    // scanf("%d", &T);
57    while(T--) {
58        // printf("Case #%d: ", cas++);
59        solve();
60    }
61    return 0;
62 }
```

4.7 二维树状数组 2D_Binary_Indexed_Tree

```cpp
1  #include <bits/stdc++.h>
2  const int MAXN=2005;
3
4  struct BIT {
5      int n,m;
6      int a[MAXN][MAXN];
7
8      int lowbit(int x){
9          return x&(-x);
10     }
11
12     void change(int x,int y,int k){
13         for(int i=x;i<=n;i+=lowbit(i)){
14             for(int j=y;j<=m;j+=lowbit(j)){
15                 a[i][j]+=k;
16             }
17         }
18     }
19
20     int sum(int x,int y){
21         int s=0;
22         for(int i=x;i>0;i-=lowbit(i)){
23             for(int j=y;j>0;j-=lowbit(j)){
24                 s+=a[i][j];
25             }
26         }
27         return s;
28     }
29 }bit;
30
31 int main(){
```

```
32    int n,m,H;
33    scanf("%d %d %d",&n, &m, &H);
34    bit.n=n;
35    bit.m=m;
36    for(int i=1;i<=n;i++)
37        for(int j=1;j<=m;j++)
38            bit.a[i][j]=0;
39    for(int i=1;i<=H;i++) {
40        int x,y;
41        scanf("%d %d", &x, &y);
42        bit.change(1,1,1);
43        bit.change(x+1,1,-1);
44        bit.change(1,y+1,-1);
45        bit.change(x+1,y+1,1);//1-2+1
46    }
47    for(int i=1;i<=n;i++)
48        for(int j=1;j<=m;j++)
49            printf("%d\n", bit.sum(i,j));
50 }
```

4.8 线段树 Segment_Tree

```cpp
1  #include <cstdio>
2  #include <algorithm>
3  #define LL long long
4  #define MAXN 100005
5  using namespace std;
6
7  struct SGT{
8      LL sum[MAXN<<2],tag[MAXN<<2];
9
10     void pushup(int x) {sum[x]=sum[x<<1]+sum[x<<1|1];}
11     void pushdown(int x,int l,int r) {
12         int m=(l+r)>>1;
13         sum[x<<1]+=tag[x]*(m-l+1);
14         tag[x<<1]+=tag[x];
15         sum[x<<1|1]+=tag[x]*(r-m);
16         tag[x<<1|1]+=tag[x];
17         tag[x]=0;
18     }
19
20     void build(int x,int l,int r,LL *a) {
21         tag[x]=0;
22         if(l==r) sum[x]=a[l];
23         else {
24             int m=(l+r)>>1;
25             build(x<<1,l,m,a);
26             build(x<<1|1,m+1,r,a);
27             pushup(x);
28         }
29     }
30
```

```
31    void modify(int x,int l,int r,int ql,int qr,LL delta) {
32        if(ql<=l && r<=qr) {
33            sum[x]+=delta*(r-l+1);
34            tag[x]+=delta;
35            return;
36        }
37        if(tag[x]) pushdown(x,l,r);
38        int m=(l+r)>>1;
39        if(ql<=m) modify(x<<1,l,m,ql,qr,delta);
40        if(m<qr) modify(x<<1|1,m+1,r,ql,qr,delta);
41        // sum[x]=tag[x]*(r-l+1)+sum[x<<1]+sum[x<<1|1];
42        pushup(x);
43    }
44
45    LL query(int x,int l,int r,int ql,int qr){
46        if(ql<=l && r<=qr) return sum[x];
47
48        if(tag[x]) pushdown(x,l,r);
49        int m=(l+r)>>1;
50        LL res=0;
51        if(ql<=m) res+=query(x<<1,l,m,ql,qr);
52        if(m<qr) res+=query(x<<1|1,m+1,r,ql,qr);
53        // res+=tag[x]*(min(qr,r)-max(ql,l)+1);
54        return res;
55    }
56 }sgt;
57
58 LL a[MAXN];
59
60 int main(){
61     int n,m;
62     scanf("%d %d",&n,&m);
63     for(int i=1;i<=n;i++) scanf("%lld", &a[i]);
64     sgt.build(1,1,n,a);
65     for(int i=1;i<=m;i++){
66         int op,l,r;
67         scanf("%d", &op);
68         if(op==1){
69             LL k;
70             scanf("%d %d %lld",&l,&r,&k);
71             sgt.modify(1,1,n,l,r,k);
72         }
73         else{
74             scanf("%d %d",&l,&r);
75             printf("%lld\n", sgt.query(1,1,n,l,r));
76         }
77     }
78     return 0;
79 }
```

4.9 线段树 Segment_Tree_Multiply

```cpp
#include <bits/stdc++.h>
#define LL long long
using namespace std;
const int MAXN=100005;
const int MOD=571373;

struct SGT{
    LL sum[MAXN<<2],ts[MAXN<<2],tm[MAXN<<2];

    void pushup(int x) {sum[x]=(sum[x<<1]+sum[x<<1|1])%MOD;}
    void pushdown(int x,int l,int r) {
        int m=(l+r)>>1;
        sum[x<<1]=(sum[x<<1]*tm[x]%MOD + ts[x]*(m-l+1)%MOD)%MOD;
        sum[x<<1|1]=(sum[x<<1|1]*tm[x]%MOD + ts[x]*(r-m)%MOD)%MOD;
        tm[x<<1]=tm[x<<1]*tm[x]%MOD;
        tm[x<<1|1]=tm[x<<1|1]*tm[x]%MOD;
        ts[x<<1]=(ts[x<<1]*tm[x]%MOD+ts[x])%MOD;
        ts[x<<1|1]=(ts[x<<1|1]*tm[x]%MOD+ts[x])%MOD;
        tm[x]=1;
        ts[x]=0;
    }

    void build(int x,int l,int r,LL* a) {
        ts[x]=0,tm[x]=1;
        if(l==r) sum[x]=a[l];
        else {
            int m=(l+r)>>1;
            build(x<<1,l,m,a);
            build(x<<1|1,m+1,r,a);
            pushup(x);
        }
    }

    void plus(int x,int l,int r,int ql,int qr,LL d) {
        if(ql<=l && r<=qr) {
            sum[x]=(sum[x]+d*(r-l+1)%MOD)%MOD;
            ts[x]=(ts[x]+d)%MOD;
            return;
        }

        pushdown(x,l,r);
        int m=(l+r)>>1;
        if(ql<=m) plus(x<<1,l,m,ql,qr,d);
        if(m<qr) plus(x<<1|1,m+1,r,ql,qr,d);
        pushup(x);
    }

    void multi(int x,int l,int r,int ql,int qr,LL d) {
        if(ql<=l && r<=qr) {
            sum[x]=(sum[x]*d)%MOD;
            ts[x]=(ts[x]*d)%MOD;
            tm[x]=(tm[x]*d)%MOD;
            return;
```

```
54          }
55
56          pushdown(x,l,r);
57          int m=(l+r)>>1;
58          if(ql<=m) multi(x<<1,l,m,ql,qr,d);
59          if(m<qr) multi(x<<1|1,m+1,r,ql,qr,d);
60          pushup(x);
61      }
62
63      LL query(int x,int l,int r,int ql,int qr) {
64          if(ql<=l && r<=qr) {
65              return sum[x];
66          }
67
68          pushdown(x,l,r);
69          int m=(l+r)>>1;
70          LL res=0;
71          if(ql<=m) res=(res+query(x<<1,l,m,ql,qr))%MOD;
72          if(m<qr) res=(res+query(x<<1|1,m+1,r,ql,qr))%MOD;
73          return res;
74      }
75  }sgt;
76
77  LL a[MAXN];
78
79  int main(){
80      int n,m,p;
81      scanf("%d %d %d", &n, &m, &p);
82      for(int i=1;i<=n;i++) scanf("%lld", &a[i]);
83      sgt.build(1,1,n,a);
84      for(int i=1;i<=m;i++){
85          int op,l,r;
86          scanf("%d", &op);
87          if(op==1){
88              LL k;
89              scanf("%d %d %lld", &l, &r, &k);
90              sgt.multi(1,1,n,l,r,k);
91          } else if(op==2) {
92              LL k;
93              scanf("%d %d %lld", &l, &r, &k);
94              sgt.plus(1,1,n,l,r,k);
95          } else {
96              scanf("%d %d",&l,&r);
97              printf("%lld\n", sgt.query(1,1,n,l,r));
98          }
99      }
100     return 0;
101 }
```

4.10　扫描线 Scanline

```
1  #include <cstdio>
```

```
2   #include <algorithm>
3   #define LL long long
4   #define MAXN 100005
5   using namespace std;
6
7   struct line{
8       int x,y1,y2,sign;
9       bool operator<(line b)const{
10          if(x!=b.x) return x<b.x;
11          else return sign>b.sign;
12      }
13  }li[MAXN<<1];
14
15  struct node{
16      int le,ri;
17      int cnt,len;
18  }sgt[MAXN<<3];
19
20  int dy[MAXN<<1];
21
22  void pushup(int cur){
23      if(sgt[cur].cnt)
24          sgt[cur].len=dy[sgt[cur].ri]-dy[sgt[cur].le];
25      else if(sgt[cur].le<sgt[cur].ri-1)
26          sgt[cur].len=sgt[cur<<1].len+sgt[cur<<1|1].len;
27      else sgt[cur].len=0;
28  }
29
30  void build(int cur,int l,int r){
31      sgt[cur].le=l, sgt[cur].ri=r;
32      sgt[cur].cnt=sgt[cur].len=0;
33      if(l<r-1){
34          build(cur<<1,l,(l+r)>>1);
35          build(cur<<1|1,(l+r)>>1,r);
36      }
37  }
38
39  void modify(int cur,int l,int r,int sign){
40      if(l<=sgt[cur].le && sgt[cur].ri<=r){
41          sgt[cur].cnt+=sign;
42      }
43      else{
44          int mid=(sgt[cur].le+sgt[cur].ri)>>1;
45          if(l<mid) modify(cur<<1,l,r,sign);
46          if(r>mid) modify(cur<<1|1,l,r,sign);
47      }
48      pushup(cur);
49  }
50
51
52  int main(){
53      int n,cnt;
54      scanf("%d", &n);
```

```
55        cnt=0;
56        for(int i=1;i<=n;i++){
57            int x1,y1,x2,y2;
58            scanf("%d %d %d %d", &x1, &y1, &x2, &y2);
59            li[(i<<1)-1].x=x1, li[i<<1].x=x2;
60            li[(i<<1)-1].y1=li[i<<1].y1=y1;
61            li[(i<<1)-1].y2=li[i<<1].y2=y2;
62            li[(i<<1)-1].sign=1, li[i<<1].sign=-1;
63            dy[++cnt]=y1, dy[++cnt]=y2;
64        }
65        sort(dy+1,dy+cnt+1);
66        cnt=unique(dy+1,dy+cnt+1)-dy-1;
67        for(int i=1;i<=(n<<1);i++){
68            li[i].y1=lower_bound(dy+1,dy+cnt+1,li[i].y1)-dy;
69            li[i].y2=lower_bound(dy+1,dy+cnt+1,li[i].y2)-dy;
70        }
71        sort(li+1,li+(n<<1)+1);
72        build(1,1,cnt);
73        LL sum=0;
74        for(int i=1;i<(n<<1);i++){
75            modify(1,li[i].y1,li[i].y2,li[i].sign);
76            sum+=1LL*sgt[1].len*(li[i+1].x-li[i].x);
77        }
78        printf("%lld\n", sum);
79        return 0;
80  }
```

4.11   zkw 线段树 ZKW_Segment_Tree

```
1   #include <cstdio>
2   #include <cstring>
3   #define LL long long
4   #define MAXN 100005
5
6   struct node{
7       LL sum,tag;
8   }sgt[MAXN<<2];
9
10  int M;
11
12  int a[MAXN];
13
14  void built(int n){
15      for(M=1;M<n+2;M<<=1);
16      memset(sgt+M,0,M*sizeof(sgt[0]));
17      for(int i=1;i<=n;i++)
18          sgt[M+i].sum=a[i];
19      for(int i=M-1;i;i--)
20          sgt[i].sum=sgt[i<<1].sum+sgt[i<<1|1].sum;
21  }
22
23  void modify(int l,int r,LL del){
```

```
24      LL len=1,lc=0,rc=0;
25      for(l=l+M-1,r=r+M+1;l^r^1;l>>=1,r>>=1,len<<=1){
26          if(~l&1) sgt[l+1].tag+=del, lc+=len;
27          if(r&1) sgt[r-1].tag+=del, rc+=len;
28          sgt[l>>1].sum+=del*lc;
29          sgt[r>>1].sum+=del*rc;
30      }
31      for(lc+=rc,l>>=1;l;l>>=1)
32          sgt[l].sum+=del*lc;
33  }
34
35  LL query(int l,int r){
36      LL res=0,len=1,lc=0,rc=0;
37      for(l=l+M-1,r=r+M+1;l^r^1;l>>=1,r>>=1,len<<=1){
38          if(~l&1) res+=sgt[l+1].sum+sgt[l+1].tag*len, lc+=len;
39          if(r&1) res+=sgt[r-1].sum+sgt[r-1].tag*len, rc+=len;
40          res+=sgt[l>>1].tag*lc;
41          res+=sgt[r>>1].tag*rc;
42      }
43      for(lc+=rc,l>>=1;l;l>>=1)
44          res+=sgt[l].tag*lc;
45      return res;
46  }
47
48  int main(){
49      int n,m;
50      scanf("%d %d", &n, &m);
51      for(int i=1;i<=n;i++)
52          scanf("%d", &a[i]);
53      built(n);
54      for(int i=1;i<=m;i++){
55          int opt,x,y;
56          scanf("%d %d %d", &opt, &x, &y);
57          if(opt==1){
58              int k;
59              scanf("%d", &k);
60              modify(x,y,k);
61          }
62          else{
63              printf("%lld\n", query(x,y));
64          }
65      }
66      return 0;
67  }
```

4.12 李超线段树 Li-Chao_Segment_Tree

```
1  #include <cstdio>
2  #include <algorithm>
3  #define N 39989
4  #define MAXN 40005
5  #define MAXT 160005
```

```
 6  const double eps=1e-12;
 7  const double inf=1e9;
 8
 9  using namespace std;
10
11  struct line{
12      int l,r;
13      double k,b;
14      int id;
15  }sgt[MAXT];
16
17  double calc(line l,int x){return l.k*x+l.b;}
18
19  void modify(int cur,int l,int r,line li){
20      if(li.l<=l && r<=li.r){
21          if(calc(li,l)-calc(sgt[cur],l)>eps && calc(li,r)-calc(sgt[cur],r)>eps)
22              sgt[cur]=li;
23          else if(calc(li,l)-calc(sgt[cur],l)>eps || calc(li,r)-calc(sgt[cur],r)>eps){
24              int mid=(l+r)>>1;
25              if(calc(li,mid)-calc(sgt[cur],mid)>eps)
26                  swap(li,sgt[cur]);
27              if(calc(li,l)-calc(sgt[cur],l)>eps)
28                  modify(cur<<1,l,mid,li);
29              else modify(cur<<1|1,mid+1,r,li);
30          }
31      }
32      else{
33          int mid=(l+r)>>1;
34          if(li.l<=mid) modify(cur<<1,l,mid,li);
35          if(li.r>mid) modify(cur<<1|1,mid+1,r,li);
36      }
37  }
38
39  line query(int cur,int l,int r,int x){
40      if(l==r) return sgt[cur];
41      else{
42          int mid=(l+r)>>1;
43          line t;
44          if(x<=mid) t=query(cur<<1,l,mid,x);
45          else t=query(cur<<1|1,mid+1,r,x);
46          if(!t.id || calc(sgt[cur],x)-calc(t,x)>eps) return sgt[cur];
47          else return t;
48      }
49  }
50
51  void built(int cur,int l,int r){
52      sgt[cur].k=sgt[cur].b=0;
53      sgt[cur].l=1; sgt[cur].r=N;
54      sgt[cur].id=0;
55      if(l<r){
56          int mid=(l+r)>>1;
57          built(cur<<1,l,mid);
58          built(cur<<1|1,mid+1,r);
```

```
59        }
60  }
61
62  int n;
63
64  int main(){
65      scanf("%d", &n);
66      built(1,1,N);
67      int last=0,id=0;
68      for(int i=1;i<=n;i++){
69          int opt;
70          scanf("%d", &opt);
71          if(opt==0){
72              int x;
73              scanf("%d", &x);
74              x=(x+last-1)%N+1;
75              printf("%d\n", last=query(1,1,N,x).id);
76          }
77          else{
78              int x0,x1,y0,y1;
79              scanf("%d %d %d %d", &x0,&y0,&x1,&y1);
80              x0=(x0+last-1)%N+1;
81              x1=(x1+last-1)%N+1;
82              y0=(y0+last-1)%1000000000+1;
83              y1=(y1+last-1)%1000000000+1;
84              line t;
85              t.id=++id;
86              t.l=min(x0,x1); t.r=max(x0,x1);
87              t.k=x1==x0?0:(double)(y1-y0)/(x1-x0);
88              t.b=x1==x0?max(y0,y1):y0-t.k*x0;
89              modify(1,1,N,t);
90          }
91      }
92      return 0;
93  }
```

4.13   可并堆左偏树 Leftist__Tree

```
1   #include <cstdio>
2   #include <algorithm>
3   #define MAXN 100005
4
5   using namespace std;
6
7   int n,m;
8
9   struct node{
10      int rt,lc,rc,dis,v;
11  }lt[MAXN];
12
13  int find(int x){
14      if(lt[x].rt==x)
```

```
15      return x;
16    return lt[x].rt=find(lt[x].rt);
17  }
18
19  int merge(int x,int y){
20    if(!x || !y) return x+y;
21    if(lt[x].v>lt[y].v || (lt[x].v==lt[y].v && x>y)) swap(x,y);//后一个条件蜜汁优化？
22    lt[x].rc=merge(lt[x].rc,y);
23    lt[lt[x].rc].rt=x;
24    if(lt[lt[x].lc].dis<lt[lt[x].rc].dis) swap(lt[x].lc,lt[x].rc);
25    lt[x].dis=lt[lt[x].rc].dis+1;
26    return x;
27  }
28
29  void pop(int x){
30    lt[x].v=-1;
31    lt[lt[x].lc].rt=lt[x].lc;
32    lt[lt[x].rc].rt=lt[x].rc;
33    lt[x].rt=merge(lt[x].lc,lt[x].rc);
34  }
35
36  int main(){
37    scanf("%d %d", &n, &m);
38    for(int i=1;i<=n;i++){
39        scanf("%d", &lt[i].v);
40        lt[i].rt=i;
41        lt[i].lc=lt[i].rc=0;
42        lt[i].dis=0;
43    }
44    lt[0].dis=0;
45    for(int i=1;i<=m;i++){
46        int opt;
47        scanf("%d",&opt);
48        if(opt==1){
49            int x,y;
50            scanf("%d %d", &x, &y);
51            int rx=find(x),ry=find(y);
52            if(lt[x].v==-1||lt[y].v==-1||rx==ry)
53                continue;
54            merge(rx,ry);
55        }
56        else{
57            int x;
58            scanf("%d", &x);
59            if(lt[x].v==-1)
60                printf("-1\n");
61            else{
62                int rx=find(x);
63                printf("%d\n", lt[rx].v);
64                pop(rx);
65            }
66        }
67    }
```

```
68      return 0;
69 }
```

4.14 Splay 树 Splay_Tree

```
1  #include <cstdio>
2  #define MAXN 100005
3  const int inf=0x3f3f3f3f;
4
5  int root,len;
6
7  struct node{
8      int v,fa,ch[2],size,cnt;
9  }sp[MAXN];
10
11 int getch(int x) {return sp[sp[x].fa].ch[1]==x;}
12 void pushup(int x) {sp[x].size=sp[x].cnt+sp[sp[x].ch[0]].size+sp[sp[x].ch[1]].size;}
13
14 void rotate(int x){
15     int f=sp[x].fa, ff=sp[f].fa;
16     int k=getch(x);
17     sp[ff].ch[getch(f)]=x; sp[x].fa=ff;
18     sp[sp[x].ch[k^1]].fa=f; sp[f].ch[k]=sp[x].ch[k^1];
19     sp[x].ch[k^1]=f; sp[f].fa=x;
20     pushup(f); pushup(x);
21 }
22
23 void splay(int x,int goal=0){
24     for(int f;(f=sp[x].fa)!=goal;rotate(x)){
25         if(sp[f].fa!=goal)
26             rotate(getch(x)==getch(f)?f:x);
27     }
28     if(!goal) root=x;
29 }
30
31 void insert(int x){
32     int cur=root,f=0;
33     while(cur&&sp[cur].v!=x){
34         f=cur;
35         cur=sp[cur].ch[x>sp[cur].v];
36     }
37     if(cur)
38         sp[cur].cnt++;
39     else{
40         cur=++len;
41         sp[f].ch[x>sp[f].v]=cur;
42         sp[cur].ch[0]=sp[cur].ch[1]=0;
43         sp[cur].fa=f;
44         sp[cur].v=x;
45         sp[cur].cnt=sp[cur].size=1;
46     }
47     splay(cur);
```

```
 48  }
 49
 50  void find(int x){
 51      int cur=root;
 52      while(x!=sp[cur].v && sp[cur].ch[x>sp[cur].v])
 53          cur=sp[cur].ch[x>sp[cur].v];
 54      splay(cur);
 55  }
 56
 57  int kth(int x){
 58      if(sp[root].size<x) return 0;
 59      int cur=root;
 60      while(1){
 61          if(x<=sp[sp[cur].ch[0]].size)
 62              cur=sp[cur].ch[0];
 63          else if(x>sp[sp[cur].ch[0]].size+sp[cur].cnt){
 64              x-=sp[sp[cur].ch[0]].size+sp[cur].cnt;
 65              cur=sp[cur].ch[1];
 66          }
 67          else return sp[cur].v;
 68      }
 69  }
 70
 71  int pre(int x){
 72      find(x);
 73      if(x>sp[root].v) return root;
 74      int cur=sp[root].ch[0];
 75      while(sp[cur].ch[1])
 76          cur=sp[cur].ch[1];
 77      return cur;
 78  }
 79
 80  int succ(int x){
 81      find(x);
 82      if(x<sp[root].v) return root;
 83      int cur=sp[root].ch[1];
 84      while(sp[cur].ch[0])
 85          cur=sp[cur].ch[0];
 86      return cur;
 87  }
 88
 89  void erase(int x){
 90      int last=pre(x),next=succ(x),del;
 91      splay(last);splay(next,last);
 92      del=sp[next].ch[0];
 93      if(sp[del].cnt>1){
 94          sp[del].cnt--;
 95          splay(del);
 96      }
 97      else{
 98          sp[next].ch[0]=0;
 99          sp[del].fa=0;
100          sp[del]=sp[len];
```

```
101        int f=sp[del].fa;
102        sp[f].ch[(sp[f].ch[1]==len)]=del;
103        sp[sp[del].ch[0]].fa=del;
104        sp[sp[del].ch[1]].fa=del;
105        if(root==len) root=del;
106        len--;
107
108    }
109 }
110
111
112 int n;
113 int main(){
114     scanf("%d", &n);
115     root=0;len=0;
116     insert(-inf);insert(inf);
117     sp[0].size=0;
118     for(int i=1;i<=n;i++){
119         int opt,x;
120         scanf("%d %d", &opt, &x);
121         if(opt==1){
122             insert(x);
123         }
124         else if(opt==2){
125             erase(x);
126         }
127         else if(opt==3){
128             find(x);
129             printf("%d\n", sp[sp[root].ch[0]].size);
130         }
131         else if(opt==4){
132             printf("%d\n", kth(x+1));
133         }
134         else if(opt==5){
135             printf("%d\n", sp[pre(x)].v);
136         }
137         else{
138             printf("%d\n", sp[succ(x)].v);
139         }
140     }
141     return 0;
142 }
```

4.15   Splay 树 Splay_Tree_Flip

```
1 #include <cstdio>
2 #include <algorithm>
3 #define MAXN 100005
4 const int inf=0x3f3f3f3f;
5
6 using namespace std;
7
```

```
 8  int root,len,a[MAXN];
 9
10  struct node{
11      int v,fa,ch[2],size,cnt,tag;
12  }sp[MAXN];
13
14  int getch(int x) {return sp[sp[x].fa].ch[1]==x;}
15  void pushup(int x) {sp[x].size=sp[x].cnt+sp[sp[x].ch[0]].size+sp[sp[x].ch[1]].size;}
16
17  void pushdown(int x){
18      if(sp[x].tag){
19          sp[sp[x].ch[0]].tag^=1;
20          sp[sp[x].ch[1]].tag^=1;
21          swap(sp[x].ch[0],sp[x].ch[1]);
22          sp[x].tag=0;
23      }
24  }
25
26  void rotate(int x){
27      int f=sp[x].fa, ff=sp[f].fa;
28      int k=getch(x);
29      sp[ff].ch[getch(f)]=x; sp[x].fa=ff;
30      sp[sp[x].ch[k^1]].fa=f; sp[f].ch[k]=sp[x].ch[k^1];
31      sp[x].ch[k^1]=f; sp[f].fa=x;
32      pushup(f); pushup(x);
33  }
34
35  void splay(int x,int goal=0){
36      for(int f;(f=sp[x].fa)!=goal;rotate(x)){
37          if(sp[f].fa!=goal)
38              rotate(getch(x)==getch(f)?f:x);
39      }
40      if(!goal) root=x;
41  }
42
43  int find(int x){
44      int cur=root;
45      while(1){
46          pushdown(cur);
47          if(x<=sp[sp[cur].ch[0]].size)
48              cur=sp[cur].ch[0];
49          else if(x>sp[sp[cur].ch[0]].size+sp[cur].cnt){
50              x-=sp[sp[cur].ch[0]].size+sp[cur].cnt;
51              cur=sp[cur].ch[1];
52          }
53          else return cur;
54      }
55  }
56
57  int built(int f,int l,int r){
58      if(l>r) return 0;
59      int mid=(l+r)>>1, cur=++len;
60      sp[cur].fa=f;
```

```
61      sp[cur].cnt=1;
62      sp[cur].v=a[mid];
63      sp[cur].tag=0;
64      sp[cur].ch[0]=built(cur,l,mid-1);
65      sp[cur].ch[1]=built(cur,mid+1,r);
66      pushup(cur);
67      return cur;
68  }
69
70  void flip(int l,int r){
71      int last=find(l-1),next=find(r+1);
72      splay(last);splay(next,last);
73      sp[sp[sp[root].ch[1]].ch[0]].tag^=1;
74  }
75
76  void dfs(int cur){
77      pushdown(cur);
78      if(sp[cur].ch[0]) dfs(sp[cur].ch[0]);
79      if(sp[cur].v!=-inf && sp[cur].v!=inf) printf("%d ", sp[cur].v);
80      if(sp[cur].ch[1]) dfs(sp[cur].ch[1]);
81  }
82
83  int n,m;
84
85  int main(){
86      scanf("%d %d", &n, &m);
87      for(int i=1;i<=n;i++) a[i+1]=i;
88      a[1]=-inf;a[n+2]=inf;
89      len=0;
90      root=built(0,1,n+2);
91      sp[0].size=0;
92      for(int i=1;i<=m;i++){
93          int l,r;
94          scanf("%d %d", &l, &r);
95          flip(l+1,r+1);
96      }
97      dfs(root);
98      return 0;
99  }
```

4.16 Splay 树 Splay_Tree_Dye&Flip

```
1   #include <cstdio>
2   #include <cstring>
3   #include <algorithm>
4   #define MAXN 500005
5   const int inf=0x3f3f3f3f;
6   using namespace std;
7
8   struct node{
9       int v,fa,ch[2],cnt;//basic
10      int size,sum,lm,rm,mm;//pushup
```

```
11        int flip,color;//pushdown
12   }sp[MAXN];
13
14   int a[MAXN],len,root,recy[MAXN],rlen;
15
16   int getch(int x){return sp[sp[x].fa].ch[1]==x;}
17
18   void pushup(int x){
19        int lc=sp[x].ch[0],rc=sp[x].ch[1];
20        sp[x].size=sp[lc].size+sp[rc].size+sp[x].cnt;
21        sp[x].sum=sp[lc].sum+sp[rc].sum+sp[x].v;
22        sp[x].lm=max(sp[lc].lm, sp[lc].sum+sp[x].v+sp[rc].lm);
23        sp[x].rm=max(sp[rc].rm, sp[rc].sum+sp[x].v+sp[lc].rm);
24        sp[x].mm=max(max(sp[lc].mm,sp[rc].mm),sp[lc].rm+sp[x].v+sp[rc].lm);
25   }
26
27   void pushdown(int x){
28        int lc=sp[x].ch[0],rc=sp[x].ch[1];
29        if(sp[x].color!=inf){
30            if(lc){
31                sp[lc].v=sp[lc].color=sp[x].color;
32                sp[lc].sum=sp[lc].size*sp[x].color;
33            }
34            if(rc){
35                sp[rc].v=sp[rc].color=sp[x].color;
36                sp[rc].sum=sp[rc].size*sp[x].color;
37            }
38            if(sp[x].color>0){
39                if(lc) sp[lc].lm=sp[lc].rm=sp[lc].mm=sp[lc].sum;
40                if(rc) sp[rc].lm=sp[rc].rm=sp[rc].mm=sp[rc].sum;
41            }
42            else{
43                if(lc) {sp[lc].lm=sp[lc].rm=0; sp[lc].mm=sp[lc].v;}
44                if(rc) {sp[rc].lm=sp[rc].rm=0; sp[rc].mm=sp[rc].v;}
45            }
46            sp[x].color=inf;
47            sp[x].flip=0;
48        }
49        else if(sp[x].flip){
50            if(lc){
51                sp[lc].flip^=1;
52                swap(sp[lc].ch[0],sp[lc].ch[1]);
53                swap(sp[lc].lm,sp[lc].rm);
54            }
55            if(rc){
56                sp[rc].flip^=1;
57                swap(sp[rc].ch[0],sp[rc].ch[1]);
58                swap(sp[rc].lm,sp[rc].rm);
59            }
60            sp[x].flip=0;
61        }
62   }
63
```

```
64  void rotate(int x){
65      int f=sp[x].fa, ff=sp[f].fa;
66      int k=getch(x);
67      sp[ff].ch[getch(f)]=x; sp[x].fa=ff;
68      sp[sp[x].ch[k^1]].fa=f; sp[f].ch[k]=sp[x].ch[k^1];
69      sp[x].ch[k^1]=f; sp[f].fa=x;
70      pushup(f); pushup(x);
71  }
72
73  void splay(int x,int goal=0){
74      for(int f;(f=sp[x].fa)!=goal;rotate(x)){
75          if(sp[f].fa!=goal)
76              rotate(getch(x)==getch(f)?f:x);
77      }
78      if(!goal) root=x;
79  }
80
81  int find(int x){
82      int cur=root;
83      while(1){
84          pushdown(cur);
85          if(x<=sp[sp[cur].ch[0]].size)
86              cur=sp[cur].ch[0];
87          else if(x>sp[sp[cur].ch[0]].size+sp[cur].cnt){
88              x-=sp[sp[cur].ch[0]].size+sp[cur].cnt;
89              cur=sp[cur].ch[1];
90          }
91          else return cur;
92      }
93  }
94
95  int built(int f,int l,int r){
96      if(l>r) return 0;
97      int mid=(l+r)>>1, cur=rlen?recy[rlen--]:++len;
98      sp[cur].v=a[mid];
99      sp[cur].fa=f;
100     sp[cur].cnt=1;
101     sp[cur].flip=0;
102     sp[cur].color=inf;
103     sp[cur].ch[0]=built(cur,l,mid-1);
104     sp[cur].ch[1]=built(cur,mid+1,r);
105     pushup(cur);
106     return cur;
107 }
108
109 void insert(int pos,int tot){
110     int l=find(pos),r=find(pos+1);
111     splay(l);splay(r,l);
112     sp[r].ch[0]=built(r,1,tot);
113     pushup(r); pushup(l);
114 }
115
116 void recycle(int x){
```

```
117 |     if(!x) return;
118 |     recycle(sp[x].ch[0]);
119 |     recycle(sp[x].ch[1]);
120 |     sp[sp[x].fa].ch[getch(x)]=0;
121 |     recy[++rlen]=x;
122 | }
123 |
124 | void erase(int pos,int tot){
125 |     int l=find(pos-1),r=find(pos+tot);
126 |     splay(l);splay(r,l);
127 |     recycle(sp[r].ch[0]);
128 |     pushup(r); pushup(l);
129 | }
130 |
131 | void dye(int pos,int tot,int c){
132 |     int l=find(pos-1),r=find(pos+tot);
133 |     splay(l);splay(r,l);
134 |     int x=sp[r].ch[0];
135 |     sp[x].color=c;
136 |     sp[x].v=c;
137 |     sp[x].sum=sp[x].size*c;
138 |     if(c>0)
139 |         sp[x].lm=sp[x].rm=sp[x].mm=sp[x].sum;
140 |     else{
141 |         sp[x].lm=sp[x].rm=0;
142 |         sp[x].mm=sp[x].v;
143 |     }
144 |     pushup(r); pushup(l);
145 | }
146 |
147 | void reverse(int pos,int tot){
148 |     int l=find(pos-1),r=find(pos+tot);
149 |     splay(l);splay(r,l);
150 |     int x=sp[r].ch[0];
151 |     sp[x].flip^=1;
152 |     swap(sp[x].ch[0],sp[x].ch[1]);
153 |     swap(sp[x].lm,sp[x].rm);
154 |     pushup(r); pushup(l);
155 | }
156 |
157 | int getsum(int pos,int tot){
158 |     int l=find(pos-1),r=find(pos+tot);
159 |     splay(l);splay(r,l);
160 |     return sp[sp[r].ch[0]].sum;
161 | }
162 |
163 | int n,m;
164 |
165 | int main(){
166 |     scanf("%d %d", &n, &m);
167 |     for(int i=1;i<=n;i++){
168 |         scanf("%d", &a[i+1]);
169 |     }
```

```
170        memset(sp,0,sizeof(sp[0]));
171        sp[0].mm=a[1]=a[n+2]=-inf;
172        rlen=0;
173        len=0;
174        root=built(0,1,n+2);
175        for(int i=1;i<=m;i++){
176            char opt[10];
177            scanf("%s", opt);
178            if(opt[0]=='I'){//Insert
179                int pos,tot;
180                scanf("%d %d", &pos, &tot);
181                for(int i=1;i<=tot;i++)
182                    scanf("%d", &a[i]);
183                insert(pos+1,tot);
184            }
185            else if(opt[0]=='D'){//Delete
186                int pos,tot;
187                scanf("%d %d", &pos, &tot);
188                erase(pos+1,tot);
189            }
190            else if(opt[2]=='K'){//Make-Same
191                int pos,tot,c;
192                scanf("%d %d %d", &pos, &tot, &c);
193                dye(pos+1,tot,c);
194            }
195            else if(opt[0]=='R'){//Reverse
196                int pos,tot;
197                scanf("%d %d", &pos, &tot);
198                reverse(pos+1,tot);
199            }
200            else if(opt[0]=='G'){//Get-Sum
201                int pos,tot;
202                scanf("%d %d", &pos, &tot);
203                printf("%d\n", getsum(pos+1,tot));
204            }
205            else if(opt[0]=='M'){//Max-Sum
206                printf("%d\n", sp[root].mm);
207            }
208        }
209        return 0;
210 }
```

5 数论

5.1 乘法逆元 Multiplicative_Inverse_Modulo

```
/*
    Coded with Leachim's ACM Template.
    No errors. No warnings. ~~
*/
#include <bits/stdc++.h>
#pragma GCC diagnostic ignored "-Wunused-const-variable"
#pragma GCC diagnostic ignored "-Wsign-conversion"
#pragma GCC diagnostic ignored "-Wsign-compare"
#define LL long long
using namespace std;
const int inf=0x3f3f3f3f;
const LL INF=0x3f3f3f3f3f3f3f3f;
const double eps=1e-7;
const int dx[4]={1,-1,0,0};
const int dy[4]={0,0,1,-1};
const int RT=3;
const int MOD=998244353;
const int MAXN=20000005;

int inv[MAXN],f[MAXN],fi[MAXN];

void pre_inv(int n,int p) {
    inv[1]=1;
    f[0]=fi[0]=f[1]=fi[1]=1;
    for(int i=2;i<=n;i++) {
        inv[i]=1LL*(p-p/i)*inv[p%i]%p;
        f[i]=1LL*f[i-1]*i%p;
        fi[i]=1LL*fi[i-1]*inv[i]%p;
    }
}

void solve() {
    int n,p;
    scanf("%d %d", &n, &p);
    pre_inv(n,p);
    for(int i=1;i<=n;i++)
        printf("%d\n", inv[i]);
}

int main() {
    int T=1,cas=1;(void)(cas);
    // scanf("%d", &T);
    while(T--) {
        // printf("Case #%d: ", cas++);
        solve();
    }
    return 0;
}
```

5.2 卢卡斯 Lucas

```cpp
#include <cstdio>
#define LL long long
#define MAXP 100005

int f[MAXP],fi[MAXP];

LL binpow(LL x,LL y,LL m){
    LL r=1%m;
    for(;y;y>>=1,x=x*x%m)
        if(y&1) r=r*x%m;
    return r;
}

void pre(LL p){
    f[0]=1;
    for(int i=1;i<=p-1;i++)
        f[i]=1LL*f[i-1]*i%p;
    fi[p-1]=binpow(f[p-1],p-2,p);
    for(int i=p-1;i;i--)
        fi[i-1]=1LL*fi[i]*i%p;
}

LL C(LL x,LL y,LL p) {
    if(x<y) return 0;
    return 1LL*f[x]*fi[y]%p*fi[x-y]%p;
}

LL lucas(LL x,LL y,LL p){
    if(!y) return 1;
    return C(x%p,y%p,p)*lucas(x/p,y/p,p)%p;
}

int main(){
    int T;
    scanf("%d", &T);
    while(T--){
        LL n,m,p;
        scanf("%lld %lld %lld", &n, &m, &p);
        pre(p);
        printf("%lld\n", lucas(n+m,m,p));
    }
    return 0;
}
```

5.3 拓展欧几里得 Exgcd

```cpp
#include <cstdio>
#include <cmath>
#define LL long long

```

```
5   LL exgcd(LL a,LL b,LL &x,LL &y){
6       if(!b || !a){
7           x=(a!=0);y=(b!=0);
8           return a+b;
9       }
10      LL g=exgcd(b,a%b,y,x);
11      y=y-(a/b)*x;
12      return g;
13  }
14
15  int main(){
16      int T;
17      scanf("%d", &T);
18      while(T--){
19          LL a,b,c,x,y,g;
20          scanf("%lld %lld %lld", &a, &b, &c);
21          g=exgcd(a,b,x,y);
22          if(c%g){
23              printf("-1\n");
24              continue;
25          }
26          a/=g;b/=g;c/=g;x*=c;y*=c;
27          LL kl=ceil((double)(-x+1)/b),kr=floor((double)(y-1)/a);
28          if(kr<kl){
29              printf("%lld %lld\n",(x+kl*b),(y-kr*a));
30          }
31          else{
32              printf("%lld %lld %lld %lld %lld\n",kr-kl+1,(x+kl*b),(y-kr*a),(x+kr*b),(y-kl*
                    a));
33          }
34      }
35      return 0;
36  }
```

5.4 拓展欧拉定理 Ex_Euler_Theorem-Automaton

```
1   #include <bits/stdc++.h>
2   using namespace std;
3   #define LL long long
4   #define MAXN 100000005
5
6   LL binpow(LL x,LL y,LL m){
7       int r=1%m;
8       x%=m;
9       while(y){
10          if(y&1) r=1LL*r*x%m;
11          x=1LL*x*x%m;
12          y>>=1;
13      }
14      return r;
15  }
16
```

```
17  void solve() {
18      int a,m;
19      scanf("%d %d", &a, &m);
20      int mm=m,phi=m;
21      for(int i=2;i*i<=m;i++) {
22          if(mm%i==0){
23              while(mm%i==0) mm/=i;
24              phi=phi/i*(i-1);
25          }
26      }
27      if(mm>1) phi=phi/mm*(mm-1);
28      char c;
29      while(!isdigit(c=getchar()));
30      int b=c-'0';
31      bool flag=0;
32      while(isdigit(c=getchar())){
33          b=10*b+c-'0';
34          if(b>=phi) b%=phi, flag=1;
35      }
36      if(flag) b+=phi;
37      printf("%lld\n", binpow(a,b,m));
38
39  }
40
41  int main() {
42      int T=1;
43      // scanf("%d", &T);
44      while(T--) {
45          solve();
46      }
47      return 0;
48  }
```

5.5 中国剩余定理 Chinese_Remainder_Theorem

```
1   /*
2       中国剩余定理
3       x%ai=bi;
4       sigma(ai*PI/ai*inv(PI/ai,ai))%PI;
5   */
6   #include <bits/stdc++.h>
7   #define LL long long
8   using namespace std;
9   const int MAXN=2000005;
10
11  LL exgcd(LL a,LL b,LL &x,LL &y){
12      if(!b || !a){
13          x=(a!=0);y=(b!=0);
14          return a+b;
15      }
16      LL g=exgcd(b,a%b,y,x);
17      y=y-(a/b)*x;
```

```
18        return g;
19  }
20
21  LL inv(LL a,LL m) {
22      LL x,y;
23      exgcd(a,m,x,y);
24      return (x%m+m)%m;
25  }
26
27  int a[MAXN],b[MAXN];
28
29  void solve() {
30      int n;
31      scanf("%d", &n);
32      LL sum=1;
33      for(int i=1;i<=n;i++) {//x%a==b
34          scanf("%d %d", &a[i], &b[i]);
35          sum=sum*a[i];
36      }
37      LL ans=0;
38      for(int i=1;i<=n;i++) {
39          ans=(ans+(__int128)b[i]*(sum/a[i])%sum*inv(sum/a[i],a[i])%sum)%sum;
40      }
41      printf("%lld\n", ans);
42  }
```

5.6 欧拉筛 Eular_Sieve

```
1   #include <cstdio>
2   #include <cstring>
3   #define MAXN 2000005
4
5   int cnt,p[MAXN];
6   bool inp[MAXN];
7   int phi[MAXN],mu[MAXN];
8
9   void eular_sieve(int n){
10      cnt=0;
11      memset(inp,0,(n+1)*sizeof(inp[0]));
12      inp[0]=inp[1]=1;
13      phi[1]=1;
14      mu[1]=1;
15      for(int i=2;i<=n;i++){
16          if(!inp[i]) p[++cnt]=i, phi[i]=i-1, mu[i]=-1;
17          for(int j=1;j<=cnt && i*p[j]<=n;j++){
18              inp[i*p[j]]=1;
19              if(i%p[j]){
20                  phi[i*p[j]]=phi[i]*(p[j]-1);
21                  mu[i*p[j]]=-mu[i];
22              }
23              else{
24                  phi[i*p[j]]=phi[i]*p[j];
```

```
25            mu[i*p[j]]=0;
26            break;
27        }
28      }
29    }
30 }
31
32 int main(){
33    int n,m;
34    scanf("%d %d", &n, &m);
35    eular_sieve(n);
36    for(int i=1;i<=m;i++){
37        int k;
38        scanf("%d",&k);
39        printf("%d\n", p[k]);
40    }
41 }
```

5.7 杜教筛 Dujiao_Sieve

```
1 #include <cstdio>
2 #include <cstring>
3 #define LL long long
4 #define MAXR R+5
5 #define R 2000000//r=n^(2/3)
6
7 int n,r,p[MAXR],cnt;
8 bool inp[MAXR];
9
10 LL phi[MAXR],sphi[MAXR],sphir[MAXR],mu[MAXR],smu[MAXR],smur[MAXR];
11
12 void pre(){
13    memset(inp+1,0,r*sizeof(inp[0]));
14    inp[0]=inp[1]=1;
15    cnt=0;
16    sphi[1]=phi[1]=1;
17    smu[1]=mu[1]=1;
18    for(int i=2;i<=r;i++){
19        if(!inp[i]) p[++cnt]=i, phi[i]=i-1, mu[i]=-1;
20        for(int j=1;j<=cnt&&i*p[j]<=r;j++){
21            inp[i*p[j]]=1;
22            if(i%p[j]){
23                phi[i*p[j]]=phi[i]*(p[j]-1);
24                mu[i*p[j]]=-mu[i];
25            }
26            else{
27                phi[i*p[j]]=phi[i]*p[j];
28                mu[i*p[j]]=0;
29                break;
30            }
31        }
32        sphi[i]=sphi[i-1]+phi[i];
```

```
33        smu[i]=smu[i-1]+mu[i];
34    }
35 }
36
37 LL sumphi(LL x){
38    if(x<=r) return sphi[x];
39    else if(sphir[n/x]) return sphir[n/x];
40    LL &sx=sphir[n/x];
41    sx=x*(x+1)/2;
42    for(LL i=2;i<=x;i++){
43        LL t=x/i,j=x/t;
44        sx-=(j-i+1)*sumphi(t);
45        i=j;
46    }
47    return sx;
48 }
49
50 LL summu(LL x){
51    if(x<=r) return smu[x];
52    else if(smur[n/x]) return smur[n/x];
53    LL &sx=smur[n/x];
54    sx=1;
55    for(LL i=2;i<=x;i++){
56        LL t=x/i,j=x/t;
57        sx-=(j-i+1)*summu(t);
58        i=j;
59    }
60    return sx;
61 }
62
63 int main(){
64    int T;
65    scanf("%d", &T);
66    r=R;
67    pre();
68    while(T--){
69        scanf("%d", &n);
70        memset(sphir+1,0,(n/r)*sizeof(sphir[0]));
71        memset(smur+1,0,(n/r)*sizeof(smur[0]));
72        printf("%lld %lld\n", sumphi(n), summu(n));
73    }
74    return 0;
75 }
```

5.8 求原根 Get_Primitive_Root

```
1 #include <cstdio>
2 #include <vector>
3
4 using namespace std;
5
6 int p;
```

```
7  vector<int> v;
8
9  int binpow(int x,int y,int mod){
10     int r=1%mod;
11     while(y){
12         if(y&1) r=(1LL*r*x)%mod;
13         x=(1LL*x*x)%mod;
14         y>>=1;
15     }
16     return r;
17 }
18
19 int main(){
20     scanf("%d", &p);
21     int pp=p-1;
22     for(int i=2;i*i<=pp;i++){
23         if(pp%i==0){
24             v.push_back(i);
25             while(pp%i==0) pp/=i;
26         }
27     }
28     if(pp>1) v.push_back(pp);
29     for(int g=2;;g++){
30         bool isg=true;
31         for(int d:v){
32             if(binpow(g,(p-1)/d,p)==1){
33                 isg=false;
34                 break;
35             }
36         }
37         if(isg){
38             printf("%d\n", g);
39             break;
40         }
41     }
42     return 0;
43 }
```

5.9 贝祖引理 Bezout_Lemma

```
1  #include <cstdio>
2
3  int gcd(int x,int y){
4      if(!x || !y) return x+y;
5      return gcd(y,x%y);
6  }
7
8  int n;
9
10 int main(){
11     scanf("%d", &n);
12     int g;
```

```
13      scanf("%d", &g);
14      if(g<0)g=-g;
15      for(int i=2;i<=n;i++){
16          int t;
17          scanf("%d", &t);
18          if(t<0) t=-t;
19          g=gcd(g,t);
20      }
21      printf("%d\n", g);
22      return 0;
23  }
```

5.10 除法分块 Block_Division

```
1   #include <cstdio>
2   #define LL long long
3
4   LL n;
5
6   int main(){
7       scanf("%lld", &n);
8       LL ans=0;
9       for(LL i=1;i<=n;i++){
10          LL t=n/i,j=n/t;
11          ans+=(j-i+1)*t;
12          i=j;
13      }
14      printf("%lld\n", ans);
15      return 0;
16  }
```

6 组合数学

6.1 康托展开 Cantor

```
/*
    康托 (Cantor) 展开: 求全排列是第几个。
*/
#include <bits/stdc++.h>
#define LL long long
using namespace std;
const int MAXN=2000005;
const int MOD=998244353;

struct BIT {
    int n,b[MAXN];

    int lowbit(int x){
        return x&(-x);
    }

    void change(int x,int y){
        for(;x<=n;x+=lowbit(x))
            b[x]+=y;
    }

    int sum(int x){
        int s=0;
        for(;x>0;x-=lowbit(x))
            s+=b[x];
        return s;
    }
}bit;

int a[MAXN],f[MAXN];

void solve() {
    int n;
    scanf("%d", &n);
    for(int i=1;i<=n;i++)
        scanf("%d", &a[i]);

    f[0]=1;
    for(int i=1;i<=n;i++)
        f[i]=1LL*f[i-1]*i%MOD;

    bit.n=n;

    int ans=1;
    for(int i=1;i<n;i++) {
        bit.change(a[i],1);
        ans=(ans+1LL*f[n-i]*(a[i]-bit.sum(a[i]))%MOD)%MOD;
    }
    printf("%d\n", ans);
```

```
50  }
51
52  int main() {
53      int T=1,cas=1;(void)(cas);
54      // scanf("%d", &T);
55      while(T--) {
56          // printf("Case #%d: ", cas++);
57          solve();
58      }
59      return 0;
60  }
```

6.2   波利亚 Pólya

```
1   /*
2       Polya定理
3       本质不同个数=sigma(颜色个数^等价置换的划分个数)/(等价置换总个数)
4
5       例题: n条个点围成一圈, m种颜色。求染色旋转本质不同个数。
6       ans = sigma(m^gcd(n,i))/n
7           = sigma(m^d * phi(n/d))/n ----- 莫比乌斯反演
8   */
9   #include <bits/stdc++.h>
10  #define LL long long
11  using namespace std;
12  const int MOD=1e9+7;
13  const int MAXN=2000005;
14
15  int binpow(int x,int y,int m) {
16      int r=1%m;
17      while(y) {
18          if(y&1) r=1LL*r*x%m;
19          x=1LL*x*x%m;
20          y>>=1;
21      }
22      return r;
23  }
24
25  int phi(int x) {
26      int r=x;
27      for(int i=2;i*i<=x;i++) {
28          if(x%i==0) {
29              r=r/i*(i-1);
30              while(x%i==0) x/=i;
31          }
32      }
33      if(x>1) r=r/x*(x-1);
34      return r;
35  }
36
37  void solve() {
38      int n,m; //n个点, m种颜色。
```

```
39    scanf("%d", &n);
40    m=n;
41    int ans=0;
42    for(int i=1;i*i<=n;i++) {
43        if(n%i==0) {
44            ans=(ans+1LL*binpow(m,i,MOD)*phi(n/i)%MOD)%MOD;
45            if(i*i!=n) {
46                ans=(ans+1LL*binpow(m,n/i,MOD)*phi(i)%MOD)%MOD;
47            }
48        }
49    }
50    ans=1LL*ans*binpow(n,MOD-2,MOD)%MOD;
51    printf("%d\n", ans);
52 }
53
54 int main() {
55    int T=1,cas=1;(void)(cas);
56    scanf("%d", &T);
57    while(T--) {
58        // printf("Case #%d: ", cas++);
59        solve();
60    }
61    return 0;
62 }
```

6.3 卡特兰数 Catalan

1. 有 $2n$ 个人排成一行进入剧场。入场费 5 元。其中只有 n 个人有一张 5 元钞票，另外 n 人只有 10 元钞票，剧院无其它钞票，问有多少中方法使得只要有 10 元的人买票，售票处就有 5 元的钞票找零？

2. 一位大城市的律师在她住所以北 n 个街区和以东 n 个街区处工作。每天她走 $2n$ 个街区去上班。如果他从不穿越（但可以碰到）从家到办公室的对角线，那么有多少条可能的道路？

3. 在圆上选择 $2n$ 个点，将这些点成对连接起来使得所得到的 n 条线段不相交的方法数？

4. 对角线不相交的情况下，将一个凸多边形区域分成三角形区域的方法数？

5. 一个栈（无穷大）的进栈序列为 $1, 2, 3, \cdots, n$ 有多少个不同的出栈序列？

6. n 个结点可构造多少个不同的二叉树？

7. n 个不同的数依次进栈，求不同的出栈结果的种数？

8. n 个 $+1$ 和 n 个 -1 构成 $2n$ 项 $a_1, a_2, \cdots, a_{2n}$，其部分和满足 $a_1 + a_2 + \cdots + a_k \geq 0 (k = 1, 2, 3, \cdots, 2n)$ 对与 n 该数列为？

H0	H1	H2	H3	H4	H5	H6
1	1	2	5	14	42	132

关于 Catalan 数的常见公式：

$$H_n = \begin{cases} \sum_{i=1}^{n} H_{i-1}H_{n-i} & n \geq 2, n \in \mathbf{N}_+ \\ 1 & n = 0, 1 \end{cases}$$

$$H_n = \frac{H_{n-1}(4n-2)}{n+1}$$

$$H_n = \binom{2n}{n} - \binom{2n}{n-1}$$

$$H_n = \frac{\binom{2n}{n}}{n+1}(n \geq 2, n \in \mathbf{N}_+)$$

6.4 斯特林数 Stirling

6.4.1 第一类斯特林数

（斯特林轮换数）

$\begin{bmatrix} n \\ k \end{bmatrix}$ 表示将 n 个两两不同的元素，划分为 k 个非空圆排列的方案数。

递推式

$$\begin{bmatrix} n \\ k \end{bmatrix} = \begin{bmatrix} n-1 \\ k-1 \end{bmatrix} + (n-1) \begin{bmatrix} n-1 \\ k \end{bmatrix}$$

边界是 $\begin{bmatrix} n \\ 0 \end{bmatrix} = [n=0]$ 。

6.4.2 第二类斯特林数

（斯特林子集数）

$\begin{Bmatrix} n \\ k \end{Bmatrix}$ 表示将 n 个两两不同的元素，划分为 k 个非空子集的方案数。

递推式

$$\begin{Bmatrix} n \\ k \end{Bmatrix} = \begin{Bmatrix} n-1 \\ k-1 \end{Bmatrix} + k \begin{Bmatrix} n-1 \\ k \end{Bmatrix}$$

边界是 $\begin{Bmatrix} n \\ 0 \end{Bmatrix} = [n=0]$ 。

6.4.3 上升幂与普通幂的相互转化

我们记上升阶乘幂 $x^{\overline{n}} = \prod_{k=0}^{n-1}(x+k)$ 。

则可以利用下面的恒等式将上升幂转化为普通幂：

$$x^{\overline{n}} = \sum_k \begin{bmatrix} n \\ k \end{bmatrix} x^k$$

如果将普通幂转化为上升幂，则有下面的恒等式：

$$x^n = \sum_k \begin{Bmatrix} n \\ k \end{Bmatrix} (-1)^{n-k} x^{\overline{k}}$$

6.4.4 下降幂与普通幂的相互转化

我们记下降阶乘幂 $x^{\underline{n}} = \dfrac{x!}{(x-n)!} = \prod_{k=0}^{n-1}(x-k)$ 。

则可以利用下面的恒等式将普通幂转化为下降幂：

$$x^n = \sum_k \left\{ {n \atop k} \right\} x^{\underline{k}}$$

如果将下降幂转化为普通幂，则有下面的恒等式：

$$x^{\underline{n}} = \sum_k \left[{n \atop k} \right] (-1)^{n-k} x^k$$

6.5 范德蒙德卷积 Vandermonde_Convolution

$$\sum_{i=0}^{k} \binom{n}{i} \binom{m}{k-i} = \binom{n+m}{k}$$

从数量分别为 n 和 m 的石堆中总共选取 k 个石子。

7 网络流

7.1 最大费用流 Minimum-Cost_Flow_Edmonds-Karp

```cpp
#include <cstdio>
#include <cstring>
#include <queue>
#include <algorithm>
#define MAXN 5005
#define MAXM 50005

const int inf = 0x3f3f3f3f;

using namespace std;

int n,m,s,t,tot,head[MAXN],dis[MAXN],maxflow,mincost;
bool inque[MAXN];

struct edge{//残量网络 residual network
    int to,cf,next,dis;//Cf:residual capacity
}e[(MAXM<<1)+1];

struct node{
    int fr,edge;
}pre[MAXN];

void add(int x,int y,int f,int d){
    tot++;
    e[tot].cf=f;
    e[tot].dis=d;
    e[tot].to=y;
    e[tot].next=head[x];
    head[x]=tot;
}

bool spfa(){//SPFA
    memset(dis+1,inf,n*sizeof(dis[0]));
    memset(inque+1,0,n*sizeof(inque[0]));
    pre[t].fr=0;
    queue<int> q;
    dis[s]=0;
    q.push(s);
    inque[s]=1;
    while(!q.empty()){
        int u=q.front();q.pop();
        inque[u]=0;
        for(int p=head[u];p;p=e[p].next){
            int v=e[p].to;
            if(e[p].cf && dis[v]>dis[u]+e[p].dis){
                dis[v]=dis[u]+e[p].dis;
                pre[v].fr=u;
                pre[v].edge=p;
                if(!inque[v]){
```

```
50                    q.push(v);
51                    inque[v]=1;
52                }
53            }
54        }
55    }
56    return pre[t].fr!=0;
57 }
58
59 int min_flow(){
60     int mn=inf;
61     for(int u=t;u!=s;u=pre[u].fr){
62         mn=min(mn,e[pre[u].edge].cf);
63     }
64     for(int u=t;u!=s;u=pre[u].fr){
65         e[pre[u].edge].cf-=mn;
66         e[pre[u].edge^1].cf+=mn;
67     }
68     return mn;
69 }
70
71 void edmonds_karp(){
72     maxflow=0,mincost=0;
73     while(spfa()){
74         int flow=min_flow();
75         maxflow+=flow;
76         mincost+=flow*dis[t];
77     }
78 }
79
80 int main(){
81     scanf("%d %d %d %d", &n, &m, &s, &t);
82     tot=1;
83     memset(head+1,0,n*sizeof(head[0]));
84     for(int i=1;i<=m;i++){
85         int f,g,w,d;
86         scanf("%d %d %d %d",&f,&g,&w,&d);
87         add(f,g,w,d);
88         add(g,f,0,-d);
89     }
90     edmonds_karp();
91     printf("%d %d\n",maxflow,mincost);
92     return 0;
93 }
```

7.2   最大流 Maximum_Flow_Edmonds-Karp

```
1 #include <cstdio>
2 #include <cstring>
3 #include <queue>
4 #include <algorithm>
5 #define MAXN 10005
```

```cpp
#define MAXM 100005

using namespace std;

const int inf = 0x3f3f3f3f;

int n,m,s,t,tot,head[MAXN],vis[MAXN];

struct edge{
    int to,cf,next;
}e[MAXM<<1];

struct node{
    int fr,edge;
}pre[MAXN];

void add(int x,int y,int z){
    tot++;
    e[tot].cf=z;
    e[tot].to=y;
    e[tot].next=head[x];
    head[x]=tot;
}

bool find_augment(){
    memset(pre+1,0,n*sizeof(pre[0]));
    memset(vis+1,0,n*sizeof(vis[0]));
    queue<int> q;
    vis[s]=1;
    q.push(s);
    while(!q.empty()){
        int u=q.front();q.pop();
        for(int p=head[u];p;p=e[p].next){
            int v=e[p].to;
            if(!vis[v] && e[p].cf){
                pre[v].fr=u;
                pre[v].edge=p;
                vis[v]=1;
                q.push(v);
                if(v==t) return true;
            }
        }
    }
    return false;
}

int min_flow(){
    int mn=inf;
    for(int u=t;u!=s;u=pre[u].fr){
        mn=min(mn,e[pre[u].edge].cf);
    }
    for(int u=t;u!=s;u=pre[u].fr){
        e[pre[u].edge].cf-=mn;
```

```
59          e[pre[u].edge^1].cf+=mn;
60      }
61      return mn;
62  }
63
64  int edmonds_karp(){
65      int flow=0;
66      while(find_augment()){
67          flow+=min_flow();
68      }
69      return flow;
70  }
71
72  int main(){
73      scanf("%d %d %d %d", &n, &m, &s, &t);
74      tot=1;
75      memset(head+1,0,n*sizeof(head[0]));
76      for(int i=1;i<=m;i++){
77          int f,g,w;
78          scanf("%d %d %d",&f,&g,&w);
79          add(f,g,w);
80          add(g,f,0);
81      }
82      printf("%d\n", edmonds_karp());
83      return 0;
84  }
```

7.3 最大流 Maximum_Flow_Dinic

```
1   #include <cstdio>
2   #include <cstring>
3   #include <queue>
4   #include <algorithm>
5   #define MAXN 10005
6   #define MAXM 100005
7
8   const int inf = 0x3f3f3f3f;
9
10  using namespace std;
11
12  int n,m,s,t,tot,head[MAXN],lb[MAXN],cur[MAXN];
13
14  struct edge{//残量网络 residual network
15      int to,cf,next;//Cf:residual capacity
16  }e[(MAXM<<1)+1];
17
18  void add(int x,int y,int z){
19      tot++;
20      e[tot].cf=z;
21      e[tot].to=y;
22      e[tot].next=head[x];
23      head[x]=tot;
```

```
24  }
25
26  bool label_vertex(){//BFS
27      memset(lb+1,0,n*sizeof(lb[0]));
28      queue<int> q;
29      lb[s]=1;
30      q.push(s);
31      while(!q.empty()){
32          int u=q.front();q.pop();
33          for(int p=head[u];p;p=e[p].next){
34              int v=e[p].to;
35              if(e[p].cf && !lb[v]){
36                  lb[v]=lb[u]+1;
37                  q.push(v);
38                  if(v==t) return true;
39              }
40          }
41      }
42      return false;
43  }
44
45  int multi_augment(int u,int lim){//DFS 多路增广
46      if(u == t) return lim;
47
48      int used=0;
49      for(int& p=cur[u];p;p=e[p].next){
50          int v=e[p].to;
51          if(e[p].cf && lb[v]==lb[u]+1){
52              int rest=multi_augment(v,min(lim-used,e[p].cf));
53              used+=rest;
54              e[p].cf-=rest;
55              e[p^1].cf+=rest;
56              if(used==lim) break;
57          }
58      }
59      return used;
60  }
61
62  int dinic(){
63      int flow=0;
64      while(label_vertex()){//BFS 标记
65          for(int i=1;i<=n;i++) cur[i]=head[i];//当前弧优化
66          flow+=multi_augment(s,inf);//DFS 顺着标记找增广路
67      }
68      return flow;
69  }
70
71  int main(){
72      scanf("%d %d %d %d", &n, &m, &s, &t);
73      tot=1;
74      memset(head+1,0,n*sizeof(head[0]));
75      for(int i=1;i<=m;i++){
76          int f,g,w;
```

```
77        scanf("%d %d %d",&f,&g,&w);
78        add(f,g,w);
79        add(g,f,0);
80      }
81      printf("%d\n", dinic());
82      return 0;
83  }
```

7.4   二分图最大匹配 Bipartite_Graph_Maximum_Matching_Dinic

```
1   #include <cstdio>
2   #include <cstring>
3   #include <queue>
4   #define MAXN 2005
5   #define MAXM 1000005
6
7   using namespace std;
8
9   const int inf=0x3f3f3f3f;
10
11  struct edge{
12      int to,cf,next;
13  }e[MAXM<<1];
14
15  int n,n1,n2,m,s,t;
16  int tot,head[MAXN],cur[MAXN],lbl[MAXN];
17
18  void add(int x,int y,int z){
19      tot++;
20      e[tot].to=y;
21      e[tot].cf=z;
22      e[tot].next=head[x];
23      head[x]=tot;
24  }
25
26  bool bfs(){
27      memset(lbl+1,0,n*sizeof(lbl[0]));
28      lbl[t]=1;
29      queue<int> q;
30      q.push(t);
31      while(!q.empty()){
32          int u=q.front();q.pop();
33          for(int p=head[u];p;p=e[p].next){
34              int v=e[p].to;
35              if(e[p^1].cf && !lbl[v]){
36                  lbl[v]=lbl[u]+1;
37                  q.push(v);
38                  if(v==s) return true;
39              }
40          }
41      }
42      return lbl[s]!=0;
```

```
43  }
44
45  int dfs(int u,int lim){
46      if(u==t)return lim;
47
48      int used=0;
49      for(int& p=cur[u];p;p=e[p].next){
50          int v=e[p].to;
51          if(e[p].cf && lbl[v]==lbl[u]-1){
52              int rest=dfs(v,min(lim-used,e[p].cf));
53              used+=rest;
54              e[p].cf-=rest;
55              e[p^1].cf+=rest;
56              if(used==lim) break;
57          }
58      }
59      return used;
60  }
61
62  int dinic(){
63      int flow=0;
64      while(bfs()){
65          for(int i=1;i<=n;i++)
66              cur[i]=head[i];
67          flow+=dfs(s,inf);
68      }
69      return flow;
70  }
71
72  int main(){
73      scanf("%d %d %d", &n1, &n2, &m);
74      n=n1+n2+2;//n个点
75      s=n-1;t=n;
76      tot=1;
77      memset(head+1,0,n*sizeof(head[0]));
78      for(int i=1;i<=m;i++){
79          int f,g;
80          scanf("%d %d",&f,&g);
81          if(f>n1 || g>n2) continue;
82          add(f,n1+g,1);
83          add(n1+g,f,0);
84      }
85      for(int i=1;i<=n1;i++){
86          add(s,i,1);
87          add(i,s,0);
88      }
89      for(int i=n1+1;i<=n1+n2;i++){
90          add(i,t,1);
91          add(t,i,0);
92      }
93      printf("%d\n", dinic());
94      return 0;
95  }
```

7.5   二分图最大匹配 Bipartite_Graph_Maximum_Matching_Hungarian

```
1   #include <cstdio>
2   #include <cstring>
3   #define MAXN 1005
4   #define MAXM 1000005
5
6   struct node{
7       int to,next;
8   }e[MAXM];
9
10  int n1,n2,m,head[MAXN],tot,dfn[MAXN],mat[MAXN];
11
12  void add(int x,int y){
13      tot++;
14      e[tot].to=y;
15      e[tot].next=head[x];
16      head[x]=tot;
17  }
18
19  bool augment(int x,int stamp){
20      for(int p=head[x];p;p=e[p].next){
21          int u=e[p].to;
22          if(dfn[u] == stamp) continue;
23          dfn[u]=stamp;
24          if(!mat[u] || augment(mat[u],stamp)){
25              mat[u]=x;
26              return true;
27          }
28      }
29      return false;
30  }
31
32  int match(){
33      memset(mat+1,0,n2*sizeof(mat[0]));
34      memset(dfn+1,0,n2*sizeof(dfn[0]));
35      int cnt=0;
36      for(int i=1;i<=n1;i++){
37          if(augment(i,i))
38              cnt++;
39      }
40      return cnt;
41  }
42
43  int main(){
44      scanf("%d %d %d", &n1,&n2,&m);
45      tot=0;
46      memset(head+1,0,n1*sizeof(head[0]));
47      for(int i=1;i<=m;i++){
48          int f,g;
49          scanf("%d %d", &f, &g);
50          if(f>n1 || g>n2)
51              continue;
```

```
52          add(f,g);
53      }
54      printf("%d\n", match());
55      return 0;
56  }
```

7.6   二分图最大匹配 Bipartite_Graph_Maximum_Matching_Hopcroft-Karp

```
1   #include <cstdio>
2   #include <cstring>
3   #include <queue>
4   #define MAXN 2005
5   #define MAXM 1000005
6
7   const int inf=0x3f3f3f3f;
8
9   using namespace std;
10
11  int n1,n2,n,m,tot,head[MAXN];
12  int mat[MAXN],lb[MAXN],dfn[MAXN];
13
14  struct edge{
15      int to,next;
16  }e[MAXM];
17
18  void add(int x,int y){
19      tot++;
20      e[tot].to=y;
21      e[tot].next=head[x];
22      head[x]=tot;
23  }
24
25  bool bfs(){
26      memset(lb+1,0,n*sizeof(lb[0]));
27      queue<int> q;
28      for(int i=1;i<=n1;i++){
29          if(!mat[i]){
30              q.push(i);
31              lb[i]=1;
32          }
33      }
34      int dis=inf;
35      while(!q.empty()){
36          int u=q.front();q.pop();
37          for(int p=head[u];p;p=e[p].next){
38              int v=e[p].to;
39              if(!lb[v]){
40                  lb[v]=lb[u]+1;
41                  if(!mat[v]) dis=lb[v];
42                  else if(lb[v]<dis){
43                      lb[mat[v]]=lb[v]+1;
```

```cpp
44                    q.push(mat[v]);
45                }
46            }
47        }
48    }
49    return dis!=inf;
50 }
51
52 bool dfs(int u,int stamp){
53     for(int p=head[u];p;p=e[p].next){
54         int v=e[p].to;
55         if(dfn[v]!=stamp && lb[v]==lb[u]+1){
56             dfn[v]=stamp;
57             if(!mat[v] || (lb[mat[v]]==lb[v]+1 && dfs(mat[v],stamp))){
58                 mat[v]=u;
59                 mat[u]=v;
60                 return true;
61             }
62         }
63     }
64     return false;
65 }
66
67 int hopcroft_karp(){
68     int cnt=0,stamp=0;
69     memset(dfn+1,0,n*sizeof(dfn[0]));
70     memset(mat+1,0,n*sizeof(mat[0]));
71     while(bfs()){
72         stamp++;
73         for(int i=1;i<=n1;i++){
74             if(!mat[i] && dfs(i,stamp)){
75                 cnt++;
76             }
77         }
78     }
79     return cnt;
80 }
81
82 int main(){
83     scanf("%d %d %d", &n1, &n2, &m);
84     n=n1+n2;
85     memset(head+1,0,n*sizeof(head[0]));
86     tot=0;
87     for(int i=1;i<=m;i++){
88         int f,g;
89         scanf("%d %d",&f,&g);
90         if(f>n1 || g>n2) continue;
91         add(f,n1+g);
92     }
93     printf("%d\n", hopcroft_karp());
94     return 0;
95 }
```

8 计算几何

8.1 计算几何 Compuational_Geometry

```cpp
#include <bits/stdc++.h>
#define MAXN 2000005
#define LL long long
using namespace std;
const double PI=acos(-1.0);
const double inf=1e100;
const double eps=1e-7;

int sgn(double d) {
    if(abs(d)<eps) return 0;
    if(d>0) return 1;
    return -1;
}

int dcmp(double x,double y) {
    if(abs(x-y)<eps) return 0;
    if(x>y) return 1;
    return -1;
}

struct Point{
    double x,y;
    Point(double x=0, double y=0):x(x),y(y){}

    Point operator + (const Point& B) const{
        return Point(x+B.x,y+B.y);
    }
    Point operator - (const Point& B) const{
        return Point(x-B.x,y-B.y);
    }
    Point operator * (const double k) const{
        return Point(x*k,y*k);
    }
    Point operator / (const double k) const{
        return Point(x/k,y/k);
    }
    bool operator < (const Point B) {
        if(dcmp(x,B.x)==0)
            return dcmp(y,B.y)<0;
        else return dcmp(x,B.x)<0;
    }

    double operator * (const Point& B) const{//点积
        return x*B.x+y*B.y;
    }
    double operator ^ (const Point& B) const {//叉积
        return x*B.y-y*B.x;
    }
};
```

```
50
51   typedef Point Vector;
52
53   double Length(Vector A) {
54       return sqrt(A*A);
55   }
56
57   double Angle(Vector A, Vector B) {//弧度
58       return acos(A*B/Length(A)/Length(B));
59   }
60
61   double Area2(Vector A, Vector B){//求平行四边形面积
62       return A^B;
63   }
64
65   Vector Rotate(Vector A, double rad) {//逆时针
66       return Vector(A.x*cos(rad)-A.y*sin(rad), A.x*sin(rad)+A.y*cos(rad));
67   }
68
69   Vector Normal(Vector A) {//逆时针转90度，单位法向量
70       double L=Length(A);
71       return Vector(-A.y/L, A.x/L);
72   }
73
74   bool ToLeftTest(Vector A, Vector B) {//B是不是在A左边
75       return sgn(A^B)>0;
76   }
77
78   struct Line {//点向式+两点。既可以line也可以seg
79       Point p1,p2;//p1->p2
80       Vector v;
81       Line(Point p1, Point p2):p1(p1),p2(p2),v((p2-p1)/Length(p2-p1)){}
82       Point point(double t) {//给t求点
83           return p1+v*t;
84       }
85   };
86   typedef Line Segment;
87
88   bool OnLine(Point P,Line l) {//判断点P是否在直线L上
89       return sgn((P-l.p1)^l.v);
90   }
91
92   Point GetIntersection(Line l1, Line l2) {//求直线交点
93       double t = (l2.v^(l1.p1-l2.p1))/(l1.v^l2.v);
94       return l1.point(t);
95   }
96
97   double DistanceToLine(Point P, Line l) {//点到直线距离
98       return abs(l.v^(P-l.p1));
99   }
100
101  Point GetProjection(Point P, Line l) {//求投影点
102      return l.point((P-l.p1)*l.v);
```

```
103 | }
104 |
105 | bool OnSegment(Point P, Segment s) {
106 |     return (OnLine(P,s) && sgn((s.p1-P)*(s.p2-P))<0);
107 | }
108 |
109 | bool InSegmentIntersection(Segment s1, Segment s2) {//不允许端点相交
110 |     double c1=(s1.p2-s1.p1)^(s2.p1-s1.p1), c2=(s1.p2-s1.p1)^(s2.p2-s1.p1);
111 |     double c3=(s2.p2-s2.p1)^(s1.p1-s2.p1), c4=(s2.p2-s2.p1)^(s1.p2-s2.p1);
112 |     return (sgn(c1)*sgn(c2)<0 && sgn(c3)*sgn(c4)<0);
113 | }
114 |
115 | void solve() {
116 |
117 | }
118 |
119 | int main() {
120 |     int T=1,cas=1;
121 |     // scanf("%d", &T);
122 |     while(T--) {
123 |         // printf("Case #%d: ", cas++);
124 |         solve();
125 |     }
126 |     return 0;
127 | }
```

9 优化算法

9.1 wqs 二分 + 决策单调性 dp

```
/*
    决策单调性 + WQS二分

    给定数列{a}。设立k个关键点。
    让sigma min|a_i-b_k| 最小（每个a_i到最近的关键点距离之和）
*/
#include <bits/stdc++.h>
#define LL long long
using namespace std;
const int MAXN=2000005;

LL a[MAXN],f[MAXN];
int cnt[MAXN],lb[MAXN],rb[MAXN],p[MAXN];

LL w(int x,int y){
    return a[y]+a[x]-a[(x+y+1)>>1]-a[(x+y)>>1];
}

int k,n;

int calc(LL d) {
    int h,t;
    h=t=1;
    lb[t]=1,rb[t]=n,p[t]=0;
    f[0]=cnt[0]=0;
    for(int i=1;i<=n;i++) {
        f[i]=f[p[h]]+w(p[h],i)+d;
        cnt[i]=cnt[p[h]]+1;

        while(rb[h]<=i) h++;
        if(h<=t && lb[h]<=i) lb[h]=i+1;

        if(h<=t && f[i]+w(i,n)>f[p[t]]+w(p[t],n)) continue;
        while(h<=t && f[i]+w(i,lb[t])<=f[p[t]]+w(p[t],lb[t])) t--;
        if(h<=t) {
            int l=lb[t],r=rb[t]+1;
            while(l<r) {
                int mid=(l+r)>>1;
                if(f[i]+w(i,mid)<=f[p[t]]+w(p[t],mid)) r=mid;
                else l=mid+1;
            }
            rb[t]=l-1;
            ++t;
            lb[t]=l,rb[t]=n,p[t]=i;
        } else {
            ++t;
            lb[t]=i,rb[t]=n,p[t]=i;
        }
    }
```

```
50      return cnt[n];
51  }
52
53  void solve() {
54      scanf("%d %d", &n, &k);
55      a[0]=0;
56      for(int i=1;i<=n;i++) {
57          scanf("%lld", &a[i]);
58          a[i]+=a[i-1];
59      }
60
61      LL l=0,r=a[n];
62      while(l<r) {
63          LL mid=(l+r+1)>>1;
64          if(calc(mid)>=k) l=mid;
65          else r=mid-1;
66      }
67      calc(l);
68
69      printf("%lld\n", f[n]-1LL*k*l);
70  }
71
72  int main() {
73      int T=1,cas=1;(void)(cas);
74      // scanf("%d", &T);
75      while(T--) {
76          // printf("Case #%d: ", cas++);
77          solve();
78      }
79      return 0;
80  }
```

9.2 斜率优化 Slope_Optimization

```
1   #include <bits/stdc++.h>
2   #define LL long long
3   using namespace std;
4   const int MAXN=2000005;
5
6   //似乎会爆LL
7   LL c[MAXN],dp[MAXN];
8   LL X[MAXN],Y[MAXN];
9
10  void solve() {
11      int n,L;
12      scanf("%d %d", &n, &L);
13      L++;
14      c[0]=0;
15      for(int i=1;i<=n;i++) {
16          scanf("%lld", &c[i]);
17          c[i]+=c[i-1]+1;//c[i] = sum(c[j])+i;
18      }
```

```
19      int h,t;
20      h=t=1;
21      dp[0]=0;
22      X[t]=c[0];
23      Y[t]=dp[0]+c[0]*c[0];
24      for(int i=1;i<=n;i++) {
25          while(h<t && Y[h+1]-Y[h]<=2*(X[h+1]-X[h])*(c[i]-L)) h++;
26          dp[i]=Y[h]-2*(c[i]-L)*X[h]+(c[i]-L)*(c[i]-L);
27          LL x=c[i],y=dp[i]+c[i]*c[i];
28          while(h<t && (y-Y[t])*(X[t]-X[t-1])<=(Y[t]-Y[t-1])*(x-X[t])) t--;
29          t++;
30          X[t]=x,Y[t]=y;
31      }
32      printf("%lld\n", dp[n]);
33  }
34
35  int main() {
36      int T=1,cas=1;(void)(cas);
37      // scanf("%d", &T);
38      while(T--) {
39          // printf("Case #%d: ", cas++);
40          solve();
41      }
42      return 0;
43  }
```

10 离线算法

10.1 莫队算法 Mo

```cpp
/*
    Coded with Leachim's ACM Template.
    No errors. No warnings. ~~
*/
#include <bits/stdc++.h>
#pragma GCC diagnostic ignored "-Wunused-const-variable"
#pragma GCC diagnostic ignored "-Wsign-conversion"
#pragma GCC diagnostic ignored "-Wsign-compare"
#define LL long long
using namespace std;
const int inf=0x3f3f3f3f;
const LL INF=0x3f3f3f3f3f3f3f3f;
const double eps=1e-7;
const int dx[4]={1,-1,0,0};
const int dy[4]={0,0,1,-1};
const int MOD=998244353;
const int MAXN=2000005;

LL gcd(LL x,LL y) {
    if(!x || !y) return x+y;
    return gcd(y,x%y);
}

/*
    莫队:
    sqrt(m)个块
    块大小n/sqrt(m);
    卡常大小n/sqrt(m*2/3);
*/

struct query {
    int l,r,id;
}q[MAXN];

int a[MAXN],cnt[MAXN];
int block,bl[MAXN];
LL ans1[MAXN],ans2[MAXN],sum;

bool cmp(const query &x, const query &y) {
    if(bl[x.l]!=bl[y.l])
        return bl[x.l]<bl[y.l];
    else if(bl[x.l]&1)
        return x.r<y.r;
    else return x.r>y.r;
}

void modify(int x,int f) {
    sum-=cnt[a[x]]*cnt[a[x]];
    cnt[a[x]]+=f;
```

```
50        sum+=cnt[a[x]]*cnt[a[x]];
51    }
52
53    void solve() {
54        int n,m;
55        scanf("%d %d", &n, &m);
56        block=n/sqrt(m*2/3);
57        for(int i=1;i<=n;i++) {
58            scanf("%d", &a[i]);
59            bl[i]=(i-1)/block+1;
60        }
61        for(int i=1;i<=m;i++) {
62            scanf("%d %d", &q[i].l, &q[i].r);
63            q[i].id=i;
64        }
65        sort(q+1,q+1+m,cmp);
66        memset(cnt+1,0,n*sizeof(cnt[0]));
67        sum=0;
68        for(int i=1,l=1,r=0;i<=m;i++) {
69            for(;r<q[i].r;r++) modify(r+1,1);
70            for(;r>q[i].r;r--) modify(r,-1);
71            for(;l<q[i].l;l++) modify(l,-1);
72            for(;l>q[i].l;l--) modify(l-1,1);
73            ans1[q[i].id]=sum-(r-l+1);
74            ans2[q[i].id]=1LL*(q[i].r-q[i].l+1)*(q[i].r-q[i].l);
75        }
76        for(int i=1;i<=m;i++) {
77            LL g=gcd(ans1[i],ans2[i]);
78            if(!g) printf("0/1\n");
79            else printf("%lld/%lld\n", ans1[i]/g, ans2[i]/g);
80        }
81    }
82
83    int main() {
84        int T=1,cas=1;(void)(cas);
85        // scanf("%d", &T);
86        while(T--) {
87            // printf("Case #%d: ", cas++);
88            solve();
89        }
90        return 0;
91    }
```

10.2   线段树分治 Segment_Tree_Partition

```
1    /*
2        可撤销并查集+线段树分治
3        离线操作: 增边, 删边, 询问图是否联通
4    */
5    #include <bits/stdc++.h>
6    #define LL long long
7    using namespace std;
```

```
8    const int MAXN=5005;
9    const int MAXM=500005;
10
11   struct DSU{
12       int fa[MAXN],rk[MAXN];
13       vector<pair<int*,int>> stk;
14       void init(int n) {
15           stk.clear();
16           for(int i=1;i<=n;i++) fa[i]=i,rk[i]=1;
17       }
18       int find(int x) {
19           if(x==fa[x]) return x;
20           return find(fa[x]);
21       }
22       bool join(int x,int y) {
23           int rx=find(x),ry=find(y);
24           if(rx==ry) return false;
25
26           if(rk[rx]>rk[ry]) swap(rx,ry);
27           stk.emplace_back(fa+rx,rx);
28           fa[rx]=ry;
29           stk.emplace_back(rk+ry,rk[ry]);
30           rk[ry]+=rk[rx];
31           return true;
32       }
33       void withdraw() {
34           *stk.back().first=stk.back().second;
35           stk.pop_back();
36           *stk.back().first=stk.back().second;
37           stk.pop_back();
38       }
39   }dsu;
40
41   int qf[MAXM],qg[MAXM];
42
43   struct SGT{
44       vector<pair<int,int>> eg[MAXM<<2];
45
46       void modify(int x,int l,int r,int ql,int qr,int f,int g) {
47           if(ql<=l && r<=qr) {
48               eg[x].emplace_back(f,g);
49               return;
50           }
51           int m=(l+r)>>1;
52           if(ql<=m) modify(x<<1,l,m,ql,qr,f,g);
53           if(m<qr) modify(x<<1|1,m+1,r,ql,qr,f,g);
54       }
55
56       void dfs(int x,int l,int r) {
57           int cnt=0;
58           for(auto pr:eg[x])
59               if(dsu.join(pr.first,pr.second))
60                   cnt++;
```

```
61          if(l==r) {
62              if(qf[l]) printf("%c\n", "NY"[dsu.find(qf[l])==dsu.find(qg[l])]);
63          } else {
64              int m=(l+r)>>1;
65              dfs(x<<1,l,m);
66              dfs(x<<1|1,m+1,r);
67          }
68          while(cnt--) dsu.withdraw();
69      }
70  }sgt;
71
72  int mp[MAXN][MAXN];
73
74  void solve() {
75      int n,m;
76      scanf("%d %d", &n, &m);
77      memset(qf+1,0,m*sizeof(qf[0]));
78      memset(qg+1,0,m*sizeof(qg[0]));
79      for(int i=1;i<=m;i++) {
80          int op,f,g;
81          scanf("%d %d %d", &op, &f, &g);
82          if(op==0) {
83              mp[f][g]=mp[g][f]=i;
84          } else if(op==1) {
85              sgt.modify(1,1,m+1,mp[f][g],i,f,g);
86              mp[f][g]=mp[g][f]=0;
87          } else qf[i]=f,qg[i]=g;
88      }
89
90      for(int f=1;f<n;f++) {
91          for(int g=f+1;g<=n;g++) {
92              if(mp[f][g]) {
93                  sgt.modify(1,1,m+1,mp[f][g],m+1,f,g);
94                  mp[f][g]=mp[g][f]=0;
95              }
96          }
97      }
98
99      dsu.init(n);
100     sgt.dfs(1,1,m+1);
101 }
102
103 int main() {
104     int T=1,cas=1;(void)(cas);
105     // scanf("%d", &T);
106     while(T--) {
107         // printf("Case #%d: ", cas++);
108         solve();
109     }
110     return 0;
111 }
```

11 杂项

11.1 快读快写 Fast__Read&Write

```cpp
#include <cstdio>

int read(){
    int ng=0,x=0;
    char ch=getchar();
    for(;ch<'0' || ch>'9';ch=getchar()) ng|=ch=='-';
    for(;ch>='0' && ch<='9';ch=getchar()) x=(x<<3)+(x<<1)+ch-'0';
    return ng?-x:x;
}

int write(){

}

int main(){
    return 0;
}
```

11.2 快速幂 Fast__Power

```cpp
#include <cstdio>

#define LL long long

LL a,b,n;

#include <cstdio>

#define LL long long

LL a,b,n;

LL binpow(LL x,LL y,LL m){
    LL r=1%m;
    for(;y;y>>=1,x=x*x%m)
        if(y&1) r=r*x%m;
    return r;
}

int main(){
    scanf("%lld %lld %lld",&a,&b,&n);
    printf("%lld^%lld mod %lld=%lld\n",a,b,n,binpow(a,b,n));
    return 0;
}


int main(){
    scanf("%lld %lld %lld",&a,&b,&n);
```

```
29      printf("%lld^%lld mod %lld=%lld\n",a,b,n,binpow(a,b,n));
30      return 0;
31  }
```

11.3 矩阵快速幂 Matrix__Fast__Power

```
1   #include <cstdio>
2   #include <cstring>
3   #define LL long long
4   #define MAXN 105
5   #define MOD 1000000007
6
7   LL n,k;
8
9   struct Matrix{
10      LL num[MAXN][MAXN];
11
12      Matrix(int opt){
13          memset(num,0,sizeof(num));
14          if(opt==1)
15              for(int i=1;i<=n;i++)
16                  num[i][i]=1;
17      }
18
19      Matrix operator*(const Matrix &y) const{
20          Matrix x=*this,ans(0);
21          for(int i=1;i<=n;i++){
22              for(int j=1;j<=n;j++){
23                  for(int k=1;k<=n;k++){
24                      ans.num[i][j]=(ans.num[i][j]+x.num[i][k]*y.num[k][j]%MOD)%MOD;
25                  }
26              }
27          }
28          return ans;
29      }
30  };
31
32  Matrix binpow(Matrix x,LL y){
33      Matrix r(1);
34      while(y){
35          if(y&1) r=r*x;
36          x=x*x;
37          y>>=1;
38      }
39      return r;
40  }
41
42  int main(){
43      scanf("%lld %lld", &n, &k);
44      Matrix A(0);
45      for(int i=1;i<=n;i++){
46          for(int j=1;j<=n;j++){
```

```
47              scanf("%lld", &A.num[i][j]);
48          }
49      }
50      A=binpow(A,k);
51      for(int i=1;i<=n;i++){
52          for(int j=1;j<=n;j++){
53              printf("%lld ", A.num[i][j]);
54          }
55          puts("");
56      }
57      return 0;
58 }
```

11.4 矩阵加速 Matrix_Acceleration

```
1  #include <cstdio>
2  #include <cstring>
3  #define LL long long
4  #define MAXN 5
5  #define MOD 1000000007
6
7  struct Matrix{
8      int num[MAXN][MAXN];
9      int n,m;
10
11     Matrix(int nn,int mm){
12         n=nn; m=mm;
13         memset(num,0,sizeof(num));
14     }
15     Matrix(int nn){
16         n=m=nn;
17         memset(num,0,sizeof(num));
18         for(int i=1;i<=nn;i++) num[i][i]=1;
19     }
20
21     Matrix operator*(const Matrix &y) const{
22         Matrix x=*this,ans(x.n,y.m);
23         if(x.m!=y.n) return ans;
24         for(int i=1;i<=x.n;i++){
25             for(int j=1;j<=y.m;j++){
26                 for(int k=1;k<=x.m;k++){
27                     ans.num[i][j]=(ans.num[i][j]+1LL*x.num[i][k]*y.num[k][j]%MOD)%MOD;
28                 }
29             }
30         }
31         return ans;
32     }
33 };
34
35 Matrix binpow(Matrix x,LL y){
36     Matrix r(x.n);
37     while(y){
```

```
38        if(y&1) r=r*x;
39        x=x*x;
40        y>>=1;
41    }
42    return r;
43 }
44
45 int main(){
46    int T;
47    scanf("%d", &T);
48    while(T--){
49        int n;
50        scanf("%d", &n);
51        if(n<=3) printf("1\n");
52        else{
53            Matrix x(3,3);
54            x.num[1][1]=x.num[2][1]=x.num[1][3]=x.num[3][2]=1;
55            x=binpow(x,n-3);
56            Matrix y(3,1);
57            for(int i=1;i<=3;i++) y.num[i][1]=1;
58            printf("%d\n", (x*y).num[1][1]);
59        }
60    }
61    return 0;
62 }
```

11.5 最长公共子序列 Longest_Increasing_Subsequence

```
1  #include <cstdio>
2  #include <functional>
3  #include <algorithm>
4  #include <map>
5  #include <utility>
6  #define MAXN 100005
7
8  using namespace std;
9
10 int n,a[MAXN],len;
11 map <int,int> mp,rmp;
12
13 int main(){
14    scanf("%d", &n);
15    for(int i=1,t;i<=n;i++){
16        scanf("%d", &t);
17        mp.insert(make_pair(t,i));
18    }
19    a[len=0]=0;
20    map <int,int>::iterator ite;
21    for(int i=1,t,v;i<=n;i++){
22        scanf("%d",&t);
23        ite=mp.find(t);
24        v=ite->second;
```

```
25        if(v>a[len])
26            a[++len]=v;
27        else{
28            int *p=upper_bound(a+1,a+len+1,v,less<int>());
29            *p=v;
30        }
31    }
32    printf("%d\n",len);
33    return 0;
34 }
```

11.6 模拟退火 Simulated_Annealing

```
1  #include <bits/stdc++.h>
2  #define MAXN 1005
3
4  int n;
5
6  double ansx,ansy;
7
8  struct point{
9      double x,y,w;
10 }p[MAXN];
11
12 double f(double x,double y){
13     double sum=0.0;
14     for(int i=1;i<=n;i++){
15         double dx=x-p[i].x,dy=y-p[i].y;
16         sum+=sqrt(dx*dx+dy*dy)*p[i].w;
17     }
18     return sum;
19 }
20
21 void SA(){
22     double T=3000,d=0.999,tt=1e-15;
23     while(T>tt){
24         double newx=ansx+(2*rand()-RAND_MAX)*T;
25         double newy=ansy+(2*rand()-RAND_MAX)*T;
26         double delta=f(newx,newy)-f(ansx,ansy);
27         if(delta<0 || exp(-delta/T)*RAND_MAX>rand()){
28             ansx=newx;
29             ansy=newy;
30         }
31         T*=d;
32     }
33 }
34
35 int main(){
36     srand(time(0));
37     scanf("%d", &n);
38     for(int i=1;i<=n;i++){
39         scanf("%lf %lf %lf", &p[i].x, &p[i].y, &p[i].w);
```

```
40 │     }
41 │     ansx=ansy=0.0;
42 │     for(int i=1;i<=n;i++){
43 │         ansx+=p[i].x;
44 │         ansy+=p[i].y;
45 │     }
46 │     ansx/=n;
47 │     ansy/=n;
48 │     SA();
49 │     printf("%.3lf %.3lf\n", ansx, ansy);
50 │     return 0;
51 │ }
```

11.7 快速沃尔什变换 Fast_Walsh_Transform

```
 1 │ /*
 2 │     Coded with Leachim's ACM Template.
 3 │     No errors. No warnings. ~~
 4 │ */
 5 │ #include <bits/stdc++.h>
 6 │ #pragma GCC diagnostic ignored "-Wunused-const-variable"
 7 │ #pragma GCC diagnostic ignored "-Wsign-conversion"
 8 │ #pragma GCC diagnostic ignored "-Wsign-compare"
 9 │ #define LL long long
10 │ using namespace std;
11 │ const int inf=0x3f3f3f3f;
12 │ const double eps=1e-7;
13 │ const int dx[4]={1,-1,0,0};
14 │ const int dy[4]={0,0,1,-1};
15 │ const int MAXN=2000005;
16 │ const int MOD=998244353;
17 │
18 │ void FWTor(int n,int* P,int f) {
19 │     for (int l=1;(l<<1)<=n;l<<=1)
20 │         for(int i=0;i<n;i+=(l<<1))
21 │             for(int j=0;j<l;j++)
22 │                 P[i+j+l]=((P[i+j+l]+P[i+j]*f)%MOD+MOD)%MOD;
23 │ }
24 │
25 │ void FWTand(int n,int* P,int f) {
26 │     for (int l=1;(l<<1)<=n;l<<=1)
27 │         for(int i=0;i<n;i+=(l<<1))
28 │             for(int j=0;j<l;j++)
29 │                 P[i+j]=((P[i+j]+P[i+j+l]*f)%MOD+MOD)%MOD;
30 │ }
31 │
32 │ void FWTxor(int n,int* P,int f) {
33 │     for (int l=1;(l<<1)<=n;l<<=1)
34 │         for(int i=0;i<n;i+=(l<<1))
35 │             for(int j=0;j<l;j++) {
36 │                 int t=P[i+j];
37 │                 P[i+j]=(1LL*f*(t+P[i+j+l])%MOD+MOD)%MOD;
```

```
38              P[i+j+l]=(1LL*f*(t-P[i+j+l])%MOD+MOD)%MOD;
39          }
40 }
41
42 int A[MAXN],B[MAXN],C[MAXN];
43
44 void solve() {
45     int n;
46     scanf("%d", &n);
47     int lim=1<<n;
48     for(int i=0;i<lim;i++) scanf("%d", &A[i]);
49     for(int i=0;i<lim;i++) scanf("%d", &B[i]);
50
51     FWTor(lim,A,1);FWTor(lim,B,1);
52     for(int i=0;i<lim;i++) C[i]=1LL*A[i]*B[i]%MOD;
53     FWTor(lim,A,-1);FWTor(lim,B,-1);FWTor(lim,C,-1);
54     for(int i=0;i<lim;i++) printf("%d ", C[i]);
55     puts("");
56
57     FWTand(lim,A,1);FWTand(lim,B,1);
58     for(int i=0;i<lim;i++) C[i]=1LL*A[i]*B[i]%MOD;
59     FWTand(lim,A,-1);FWTand(lim,B,-1);FWTand(lim,C,-1);
60     for(int i=0;i<lim;i++) printf("%d ", C[i]);
61     puts("");
62
63     FWTxor(lim,A,1);FWTxor(lim,B,1);
64     for(int i=0;i<lim;i++) C[i]=1LL*A[i]*B[i]%MOD;
65     FWTxor(lim,A,(MOD+1)/2);FWTxor(lim,B,(MOD+1)/2);FWTxor(lim,C,(MOD+1)/2);
66     for(int i=0;i<lim;i++) printf("%d ", C[i]);
67     puts("");
68 }
69
70 int main() {
71     int T=1,cas=1;(void)(cas);
72     // scanf("%d", &T);
73     while(T--) {
74         // printf("Case #%d: ", cas++);
75         solve();
76     }
77     return 0;
78 }
```

11.8 快速莫比乌斯变换 Fast_Mobius_Transform

```
1 /*
2     Coded with Leachim's ACM Template.
3     No errors. No warnings. ~~
4 */
5 #include <bits/stdc++.h>
6 #pragma GCC diagnostic ignored "-Wunused-const-variable"
7 #pragma GCC diagnostic ignored "-Wsign-conversion"
8 #pragma GCC diagnostic ignored "-Wsign-compare"
```

```
 9  #define LL long long
10  using namespace std;
11  const int inf=0x3f3f3f3f;
12  const LL INF=0x3f3f3f3f3f3f3f3f;
13  const double eps=1e-7;
14  const int dx[4]={1,-1,0,0};
15  const int dy[4]={0,0,1,-1};
16  const int MOD=998244353;
17  const int MAXN=2000005;
18
19  void FMTor(int n,int* P,int f) {
20      for (int i=1;i<n;i<<=1)
21          for(int j=0;j<n;j++)
22              if(i&j) P[j]=((P[j]+f*P[i^j])%MOD+MOD)%MOD;
23  }
24
25  void FMTand(int n,int* P,int f) {
26      for (int i=1;i<n;i<<=1)
27          for(int j=0;j<n;j++)
28              if(i&~j) P[j]=((P[j]+f*P[i|j])%MOD+MOD)%MOD;
29  }
30
31  void FWTxor(int n,int* P,int f) {
32      for (int l=1;(l<<1)<=n;l<<=1)
33          for(int i=0;i<n;i+=(l<<1))
34              for(int j=0;j<l;j++) {
35                  int t=P[i+j];
36                  P[i+j]=(1LL*f*(t+P[i+j+l])%MOD+MOD)%MOD;
37                  P[i+j+l]=(1LL*f*(t-P[i+j+l])%MOD+MOD)%MOD;
38              }
39  }
40
41  int A[MAXN],B[MAXN],C[MAXN];
42
43  void solve() {
44      int n;
45      scanf("%d", &n);
46      int lim=1<<n;
47      for(int i=0;i<lim;i++) scanf("%d", &A[i]);
48      for(int i=0;i<lim;i++) scanf("%d", &B[i]);
49
50      FMTor(lim,A,1);FMTor(lim,B,1);
51      for(int i=0;i<lim;i++) C[i]=1LL*A[i]*B[i]%MOD;
52      FMTor(lim,A,-1);FMTor(lim,B,-1);FMTor(lim,C,-1);
53      for(int i=0;i<lim;i++) printf("%d ", C[i]);
54      puts("");
55
56      FMTand(lim,A,1);FMTand(lim,B,1);
57      for(int i=0;i<lim;i++) C[i]=1LL*A[i]*B[i]%MOD;
58      FMTand(lim,A,-1);FMTand(lim,B,-1);FMTand(lim,C,-1);
59      for(int i=0;i<lim;i++) printf("%d ", C[i]);
60      puts("");
61
```

```
62      FWTxor(lim,A,1);FWTxor(lim,B,1);
63      for(int i=0;i<lim;i++) C[i]=1LL*A[i]*B[i]%MOD;
64      FWTxor(lim,A,(MOD+1)/2);FWTxor(lim,B,(MOD+1)/2);FWTxor(lim,C,(MOD+1)/2);
65      for(int i=0;i<lim;i++) printf("%d ", C[i]);
66      puts("");
67  }
68
69  int main() {
70      int T=1,cas=1;(void)(cas);
71      // scanf("%d", &T);
72      while(T--) {
73          // printf("Case #%d: ", cas++);
74          solve();
75      }
76      return 0;
77  }
```

11.9   快速子集变换 Fast_Subset_Transform

```
1   /*
2       Coded with Leachim's ACM Template.
3       No errors. No warnings. ~~
4   */
5   #include <bits/stdc++.h>
6   #pragma GCC diagnostic ignored "-Wunused-const-variable"
7   #pragma GCC diagnostic ignored "-Wsign-conversion"
8   #pragma GCC diagnostic ignored "-Wsign-compare"
9   #define LL long long
10  using namespace std;
11  const int inf=0x3f3f3f3f;
12  const LL INF=0x3f3f3f3f3f3f3f3f;
13  const double eps=1e-7;
14  const int dx[4]={1,-1,0,0};
15  const int dy[4]={0,0,1,-1};
16  const int MOD=1e9+9;
17  const int MAXN=2000005;
18
19  void FWTor(int n,int* P,int f) {
20      for (int l=1;(l<<1)<=n;l<<=1)
21          for(int i=0;i<n;i+=(l<<1))
22              for(int j=0;j<l;j++)
23                  P[i+j+l]=((P[i+j+l]+P[i+j]*f)%MOD+MOD)%MOD;
24  }
25
26  int A[21][MAXN],B[21][MAXN],C[21][MAXN];
27
28  void solve() {
29      int n;
30      scanf("%d", &n);
31      int lim=1<<n;
32      for(int i=0;i<=n;i++) {
33          memset(A[i],0,lim*sizeof(A[i][0]));
```

```
34          memset(B[i],0,lim*sizeof(B[i][0]));
35          memset(C[i],0,lim*sizeof(C[i][0]));
36      }
37      for(int i=0;i<lim;i++)
38          scanf("%d", &A[__builtin_popcount(i)][i]);
39      for(int i=0;i<lim;i++)
40          scanf("%d", &B[__builtin_popcount(i)][i]);
41
42      for(int i=0;i<=n;i++) {
43          FWTor(lim,A[i],1);
44          FWTor(lim,B[i],1);
45      }
46
47      for(int i=0;i<=n;i++) {
48          for(int j=0;j<=i;j++) {
49              for(int s=0;s<lim;s++) {
50                  C[i][s]=(C[i][s]+1LL*A[j][s]*B[i-j][s]%MOD)%MOD;
51              }
52          }
53          FWTor(lim,C[i],-1);
54      }
55
56      for(int i=0;i<lim;i++) printf("%d ", C[__builtin_popcount(i)][i]);
57      puts("");
58  }
59
60  int main() {
61      int T=1,cas=1;(void)(cas);
62      // scanf("%d", &T);
63      while(T--) {
64          // printf("Case #%d: ", cas++);
65          solve();
66      }
67      return 0;
68  }
```

11.10 24 点 24_Point

```
1   /*
2       Coded with Leachim's ACM Template.
3       No errors. No warnings. ~~
4   */
5   #include <bits/stdc++.h>
6   #pragma GCC diagnostic ignored "-Wunused-const-variable"
7   #pragma GCC diagnostic ignored "-Wsign-conversion"
8   #pragma GCC diagnostic ignored "-Wsign-compare"
9   #define LL long long
10  using namespace std;
11  const int inf=0x3f3f3f3f;
12  const LL INF=0x3f3f3f3f3f3f3f3f;
13  const double eps=1e-7;
14  const int dx[4]={1,-1,0,0};
```

```
15  const int dy[4]={0,0,1,-1};
16  const int MOD=998244353;
17  const int MAXN=2000005;
18
19  int a[5],p[5];
20
21  set<pair<double,string>> dfs(int l, int r) {
22      set<pair<double,string>> s;
23      if(l==r) {
24          s.emplace(a[p[l]],to_string(a[p[l]]));
25      }
26      for(int i=l;i<r;i++) {
27          auto s1=dfs(l,i);
28          auto s2=dfs(i+1,r);
29          for(auto p1:s1) {
30              for(auto p2:s2) {
31                  s.emplace(p1.first+p2.first,"("+p1.second+"+"+p2.second+")");
32                  s.emplace(p1.first-p2.first,"("+p1.second+"-"+p2.second+")");
33                  s.emplace(p1.first*p2.first,"("+p1.second+"*"+p2.second+")");
34                  if(abs(p2.first)>eps) s.emplace(p1.first/p2.first,"("+p1.second+"/"+p2.
                        second+")");
35              }
36          }
37      }
38      return s;
39  }
40
41  void solve() {
42      for(int i=1;i<=4;i++) {
43          scanf("%d", &a[i]);
44          p[i]=i;
45      }
46      do {
47          auto s=dfs(1,4);
48          for(auto p:s) {
49              if(abs(p.first-24)<eps) {
50                  cout<<p.second.substr(1,p.second.size()-2)<<endl;
51                  return;
52              }
53          }
54      }while(next_permutation(p+1,p+5));
55
56      printf("-1\n");
57  }
58
59  int main() {
60      int T=1,cas=1;(void)(cas);
61      // scanf("%d", &T);
62      while(T--) {
63          // printf("Case #%d: ", cas++);
64          solve();
65      }
66      return 0;
```

```
67  }
```