

● string.cpp

```
int* z_function(char s[],int n)
{
    int* z = new int[n];
    memset(z, 0, sizeof(z));
    z[0] = n;
    int L = 0, R = 1;
    for (int i=1; i<n; ++i)
        if (R <= i || z[i-L] >= R-i){
            int x = (R <= i ? i : R);
            while (x < n && s[x] == s[x-i]) x++;
            z[i] = x-i;
            if (i < x) {L = i; R = x;}
        }
        else
            z[i] = z[i-L];
    return z;
}
/*banana*/
void IBWT(){
    vector<int> index[256];
    for(int i=0; i<N; i++) index[t[i]].push_back(i);

    for(int i=0, n=0; i<256; i++)
        for(int j=0; j<index[i].size(); j++)
            next[n++] = index[i][j];
}
/*kmp*/
for(int i=0, j=-1; i<t.size(); i++){
    while(j>=0 && p[j+1]!=t[i]) j = f[j];
    if(p[j+1]==t[i]) j++;
    if(j==p.size()-1){
        int ans = i - p.size();
        j = f[j];
    }
}
```

● 最小字典序表示法

```
void solve(){
    scanf("%s",t);
    s[0] = '\0';
    strcat(s, t);
    strcat(s+n, t);

    int j = 1, i = 0;
    while(i<n && j<n){
        if(s[j]<s[i]) i = j, j = i+1;
        else if(s[j]>s[i]) j++;
        else{
            int k = 0;
            while(k<n){
                if(s[i+k]==s[j+k]) k++;
                else if(s[i+k]<s[j+k]){
                    j = j+k + 1;
                    break;
                }
            }
            else{
                i = j;
                j = i + 1;
                break;
            }
        }
        if(k==n) break;
    }
    printf("%d\n", i);
}
```

● sa.cpp

```
int d2[maxn], d[maxn];
int ra[maxn], he[maxn], sa[maxn], c[maxn];
void build_sa(int n,int m){
    int *x = ra, *y = he;

    for(int i=0; i<m; i++) c[i] = 0;
    for(int i=0; i<n; i++) c[x[i]=d[i]]++;
    for(int i=1; i<m; i++) c[i] += c[i-1];
    for(int i=n-1; i>=0; i--) sa[--c[x[i]]] = i;

    for(int k=1; k<=n; k<=1){
        int p = 0;
        for(int i=n-k; i<n; i++) y[p++] = i;
        for(int i=0; i<n; i++) if(sa[i]>=k) y[p++] = sa[i]-k;

        for(int i=0; i<m; i++) c[i] = 0;
        for(int i=0; i<n; i++) c[x[y[i]]]++;
        for(int i=1; i<m; i++) c[i] += c[i-1];
        for(int i=n-1; i>=0; i--) sa[--c[x[y[i]]]] = y[i];

        swap(x, y);
        p = 0;
        x[sa[0]] = p++;
        for(int i=1; i<n; i++)
            x[sa[i]] = y[sa[i]]==y[sa[i-1]]&&sa[i]+k<n&&sa[i-1]+k<n&&y[sa[i]+k]==y[sa[i-1]+k]?
            p-1:p++;
        if(p>=n) break;
        m = p;
    }
}
void build_he(int n){
    for(int i=0; i<n; i++) ra[sa[i]] = i;
    // def he[i] = lcp(sa[i], sa[i-1])
    // --> he[ra[i]]>=he[ra[i-1]]-1
    he[0] = 0;
    for(int i=0,k=0; i<n; i++){
        if(ra[i]){
            if(k) k--;
            int j = sa[ra[i]-1];
            while(d[i+k]==d[j+k] && i+k < n && j+k < n) k++;
            he[ra[i]] = k;
        }
    }
}
```

● ACf.cpp

```
const int maxn = 100;
const int maxkind = 26;
const int maxlen = 100;
const int maxsize = maxn*maxlen + 10;
struct AC{
    int ch[maxsize][maxkind], f[maxsize],
    last[maxsize], val[maxsize];
    int root, memid;
    AC(){ clear(); }
    void newNode(){
        memset(ch[memid], 0, sizeof(ch[memid]));
        f[memid] = last[memid] = val[memid] = 0;
        return memid++;
    }
    void clear(){
        memid = 0;
        root = newNode();
    }
    void insert(const char* s,int v){
        int tmp = root;
        for(int i=0; s[i]; i++){
            int id(ID[s[i]]);
            if(!ch[tmp][id]) ch[tmp][id] = newNode();
            tmp = ch[tmp][id];
        }
    }
}
```

```

    val[tmp] = v;
}
void getfail(){
    queue<int> Q;
    f[root] = 0;
    for(int i=0; i<maxkind; i++) if(ch[root][i]){
        int u = ch[root][i];
        f[u] = last[u] = 0;
        Q.push(u);
    }
    while(!Q.empty()){
        int x = Q.front(); Q.pop();
        for(int i=0; i<maxkind; i++) if(ch[x][i]){
            int tmp = f[x], u = ch[x][i];
            while(tmp && !ch[tmp][i]) tmp = f[tmp];
            f[u] = ch[tmp][i];
            last[u] = val[f[u]]? f[u]:last[f[u]];
            Q.push(u);
        }
    }
}
void find(const char *s){
    int tmp = root;
    for(int i=0; s[i]; i++){
        int id = ID(s[i]);
        while(tmp && !ch[tmp][id]) tmp = f[tmp];
        tmp = ch[tmp][id];
        if(val[id])// find
        if(last[id]) //find
    }
}
};

```

● Hash.cpp

```

#define MAXN 1000000
#define prime_mod 1073676287
typedef long long T;
char s[MAXN+5];
T h[MAXN+5];
T h_base[MAXN+5];
inline void hash_init(int len,T prime=0xdefaced){
    h_base[0]=1;
    for(int i=1;i<=len;++i){
        h[i]=(h[i-1]*prime+s[i-1])%prime_mod;
        h_base[i]=(h_base[i-1]*prime)%prime_mod;
    }
}
inline T get_hash(int l,int r){
    return (h[r+1]-(h[l]*h_base[r-
l+1])%prime_mod+prime_mod)%prime_mod;
}

```

● Treap.cpp

```

#include <iostream>
#include <algorithm>
#include <cstdio>
using namespace std;
const int INF = 9e9;
struct Node{
    int val, pri, size, mi, tag;
    bool rev;
    Node *l, *r;
    Node(){
        Node(int
v):val(v),pri(rand()),size(1),rev(0),mi(v),tag(0){ l =
r = NULL;}
    void down();
    void up();
}*root;

int Size(Node *o){ return o? o->size:0;}
int Min(Node *o){ return o? o->mi:INF;}
int Val(Node *o){ return o? o->val:-1;}

```

```

void Node::down(){
    if(tag){
        val += tag;
        mi += tag;
        if(l) l->tag += tag;
        if(r) r->tag += tag;
        tag = 0;
    }
    if(rev){
        swap(l,r);
        if(l) l->rev ^= 1;
        if(r) r->rev ^= 1;
        rev = 0;
    }
}

void Node::up(){
    if(l) l->down();
    if(r) r->down();
    size = 1 + Size(l) + Size(r);
    mi = std::min( min(Min(l), Min(r)),val );
}

void print(Node *o){
    if(o){
        print(o->l);
        printf("%d ", o->val);
        print(o->r);
    }
}

Node* merge(Node* a,Node *b){
    if(!a || !b) return a? a:b;
    if(a->pri < b->pri){
        a->down();
        a->r = merge(a->r, b);
        a->up();
        return a;
    }else{
        b->down();
        b->l = merge(a, b->l);
        b->up();
        return b;
    }
}

void spilt(Node *o, Node *&a, Node *&b, int k){
    if(!o) a = b = NULL;
    else{
        o->down();
        if(Size(o->l)>=k){
            b = o;
            spilt(o->l, a, b->l, k);
        }
        else{
            a = o;
            spilt(o->r, a->r, b, k-Size(o->l)-1);
        }
        o->up();
    }
}

void Insert(Node *&o, int k,int v){
    if(!o) o = new Node(v);
    else{
        Node* tmp = new Node(v);
        Node *a, *b;
        spilt(o, a, b, k);
        o = merge(merge(a,tmp), b);
    }
}

void Del(Node *&o, int k){
    if(!o) return;
}

```

```

else{
    Node *a, *b, *c;
    spilt(o, a, b, k);
    spilt(a, a, c, k-1);
    o = merge(a, b);
}
}

int Min(Node *&o, int x, int y){
    if(!o) return 0;
    else{
        Node *a, *b, *c;
        spilt(o, a, b, y);
        spilt(a, a, c, x-1);
        if(c==0) return 0;
        c->up();
        int ans = c->mi;
        o = merge(merge(a,c), b);
        return ans;
    }
}

void Add(Node *&o, int x, int y, int v){
    if(!o) return;
    Node *a, *b, *c;
    spilt(o, a, b, y);
    spilt(a, a, c, x-1);
    if(c) c->tag += v;
    o = merge(merge(a,c), b);
}

void Reverse(Node *&o, int x, int y){
    if(!o) return;
    Node *a, *b, *c;
    spilt(o, a, b, y); // a b c
    spilt(a, a, c, x-1);
    if(c) c->rev ^= 1;
    o = merge(merge(a,c), b);
}

void Rotate(Node *&o, int x, int y, int t){
    if(!o) return;
    Node *a, *b, *c;
    spilt(o, a, b, y);
    spilt(a, a, c, x-1);

    Node *d, *e;
    t %= (y-x+1);
    if(t<0) t = y-x+1+t;
    spilt(c,d,e, Size(c)-t);
    c = merge(e, d);
    o = merge(merge(a,c), b);
}

/*
ADD x y D: Add D to each number in sub-sequence
{Ax ... Ay}. For example, performing "ADD 2 4 1" on
{1, 2, 3, 4, 5} results in {1, 3, 4, 5, 5}
REVERSE x y: reverse the sub-sequence {Ax ... Ay}. For
example, performing "REVERSE 2 4" on {1, 2, 3, 4, 5}
results in {1, 4, 3, 2, 5}
REVOLVE x y T: rotate sub-sequence {Ax ... Ay} T
times. For example, performing "REVOLVE 2 4 2" on {1,
2, 3, 4, 5} results in {1, 3, 4, 2, 5}
INSERT x P: insert P after Ax. For example, performing
"INSERT 2 4" on {1, 2, 3, 4, 5} results in {1, 2, 4,
3, 4, 5}
DELETE x: delete Ax. For example, performing "DELETE
2" on {1, 2, 3, 4, 5} results in {1, 3, 4, 5}
MIN x y: query the participant what is the minimum
number in sub-sequence {Ax ... Ay}. For example, the
correct answer to "MIN 2 4" on {1, 2, 3, 4, 5} is 2
*/
int main()
{

```

```

int n;
while(scanf("%d",&n)==1){
    root = NULL;
    for(int i=0; i<n; i++){
        scanf("%d",&a);
        root = merge(root, new Node(a));
    }
    int m, x, y, c;
    char s[20];
    scanf("%d", &m);
    for(int i=0; i<m; i++){
        scanf("%s", s);
        if(s[0]=='A'){
            scanf("%d%d",&x,&y,&c);
            Add(root, x, y, c);
        }
        else if(s[0]=='R' && s[3]=='E'){
            scanf("%d",&x,&y);
            Reverse(root, x, y);
        }
        else if(s[0]=='R'){
            scanf("%d%d",&x,&y,&c);
            Rotate(root, x, y, c);
        }
        else if(s[0]=='I'){
            scanf("%d",&x,&y);
            n++;
            Insert(root, x, y);
        }
        else if(s[0]=='D'){
            scanf("%d",&x);
            Del(root, x);
            n--;
        }
        else{
            scanf("%d",&x,&y);
            printf("%d\n", Min(root, x, y));
        }
    }
}
return 0;
}

```

● LCA

```

// adj[u] : adjacency list of u
// par[u][i] : (2^i)-th parent pf u
int LOG = 20;
int time = 0;
void dfs(int u, int p) {
    par[u][0] = p;
    timer_in[u] = ++timer;
    for (int v : adj[u]) if (v!=p) dfs(v, u);
    time_out[u] = ++timer;
}
bool anc(int x, int y) {
    return time_in[x] <= time_in[y]
        && time_out[y] <= time_out[x];
}
int lca(int x, int y) {
    if (anc(y, x)) return y;
    for (int j = LOG; j >= 0; j--) {
        if (!anc(par[y][j], x)) y = par[y][j];
    }
    return par[y][0];
}
int main() {
    int root = 1; // set root node
    dfs(root, root);

    for (int j = 1; j <= LOG; j++)
        for (int i = 1; i <= n; i++)
            pair[i][j] = par[par[i][j-1]][j-1];
    return 0;
}

```

● Dinic flow

```
const int maxn = 1000+10;
const int INF = 2147483647;
template<class T>
struct Dinic{
    struct Edge{
        int f,to;
        T c;
        Edge(int _f,int _to,T _c):f(_f),to(_to),c(_c){}
    };

    vector<int> G[maxn];
    vector<Edge> es;
    int level[maxn],st, end, n;

    void init(int _n){
        n = _n;
        es.clear();
        for(int i=0; i<=n; i++) G[i].clear();
    }

    void addEdge(int f,int t,T c, bool
directed=false){
        es.push_back(Edge(f,t,c));
        G[f].push_back(es.size()-1);
        es.push_back(Edge(t,f,directed?0:c));
        G[t].push_back(es.size()-1);
    }

    bool BFS(int s,int t){
        queue<int> Q;
        for(int i=0; i<=n; i++) level[i] = 0;
        level[s] = 1;
        Q.push(s);
        while(!Q.empty()){
            int x = Q.front(); Q.pop();
            for(int i=0; i<G[x].size(); i++){
                Edge e = es[ G[x][i] ];
                if(e.c==0 || level[e.to]) continue;
                level[e.to] = level[x] + 1;
                Q.push(e.to);
            }
        }
        return level[t]!=0;
    }

    T DFS(int s,int cur_flow){ // can't exceed c
        if(s==end) return cur_flow;
        T ans = 0, temp, total = 0;
        for(int i=0; i<G[s].size(); i++){
            Edge &e = es[ G[s][i] ];
            if(e.c==0 || level[e.to]!=level[s]+1)
continue;
            temp = DFS(e.to, min(e.c, cur_flow));
            if(temp!=0){
                e.c -= temp;
                es[G[s][i]^1].c += temp;
                cur_flow -= temp;
                total += temp;
                if(cur_flow==0) break;
            }
        }
        return total;
    }

    T maxFlow(int s,int t){
        T ans = 0;
        st = s, end = t;
        while(BFS(s,t)){
            while(true){
                T temp = DFS(s,INF);
                if(temp==0) break;
                ans += temp;
            }
        }
        return ans;
    }
};
```

● 最小費用流

```
struct Min_cost_flow {
    // 0-base
    struct Edge {
        int fr,to,flow,cap,cost;
    };
    int V,E;
    vector<Edge> edge;
    vector<int> G[MAXV+5];

    void init(int _V) {
        V=_V;
        E=0;
        for (int i=0;i<V;i++) G[i].clear();
    }

    void add_edge(int fr, int to, int cap, int cost)
    {
        edge.pb({fr,to,0,cap,cost});
        edge.pb({to,fr,0,0,-cost});
        G[fr].pb(E);
        G[to].pb(E^1);
        E+=2;
    }

    bool SPFA(int src, int dest, int &ans_flow, int
&ans_cost) {
        queue<int> que;
        int dist[MAXV+5],pre[MAXV+5],flow[MAXV+5];
        bool inque[MAXV+5];
        for (int i=0;i<V;i++) {
            dist[i]=INF;
            pre[i]=-1;
            inque[i]=false;
            flow[i]=-1;
        }
        dist[src]=0;
        flow[src]=INF;
        inque[src]=true;
        que.push(src);
        while (!que.empty()) {
            int v=que.front(); que.pop();
            inque[v]=false;
            for (auto idx : G[v]) {
                Edge &e=edge[idx];
                if (e.flow<e.cap &&
dist[e.fr]+e.cost<dist[e.to]) {

                    flow[e.to]=min(flow[e.fr],e.cap-e.flow);
                    dist[e.to]=dist[e.fr]+e.cost;
                    pre[e.to]=idx;
                    if (!inque[e.to])
que.push(e.to);

                    inque[e.to]=true;
                }
            }
        }
        if (dist[dest]==INF) return false;
        int v=dest;
        ans_flow+=flow[dest];
        ans_cost+=(dist[dest]*flow[dest]);
        while (v!=src) {
            static int num;
            edge[pre[v]].flow+=flow[dest];
            edge[pre[v]^1].flow-=flow[dest];
            v=edge[pre[v]].fr;
        }
        return true;
    }

    PII min_cost_flow(int src, int dest) {
        int ans_flow=0, ans_cost=0;
    }
};
```

```

        while (SPFA(src,dest,ans_flow,ans_cost));
        return make_pair(ans_flow,ans_cost);
    }
};

```

● 穩定婚姻

```

const int maxn = 1100;
int manWant[maxn][maxn], nextW[maxn];
int women[maxn][maxn], order[maxn][maxn];
int wife[maxn], husband[maxn];
queue<int> singleDog;

void engage(int m, int w){
    if(husband[w]!=0){
        wife[ husband[w] ] = 0;
        singleDog.push( husband[w] );
        husband[w] = 0;
    }
    husband[w] = m;
    wife[m] = w;
    // cout << m << " --> " << w << endl;
}

int main()
{
    int Time, n, cas = 0;
    scanf("%d",&Time);

    while(Time-- && scanf("%d",&n)==1){
        for(int i=1; i<=n; i++){
            for(int j=1; j<=n; j++){
                scanf("%d",&manWant[i][j]);
                nextW[i] = 1;
                wife[i] = 0;
                singleDog.push(i);
            }
            for(int i=1; i<=n; i++){
                for(int j=1; j<=n; j++){
                    scanf("%d",&women[i][j]);
                    order[i][ women[i][j] ] = j;
                }
                husband[i] = 0;
            }

            while(!singleDog.empty()){
                int x = singleDog.front(); singleDog.pop();
                // cout << x << endl;
                int to = manWant[x][nextW[x]++];

                if(husband[to]==0) engage(x, to);
                else if(order[to][husband[to]] >
                    order[to][x]) engage(x, to);
                else singleDog.push(x);
            }
            if(cas++) printf("\n");
            for(int i=1; i<=n; i++) printf("%d\n",
                wife[i]);
        }
        return 0;
    }
}

```

● EXT_GCD

```

typedef long long LL;
typedef pair < LL, LL> ii;

ii exd_gcd( LL a, LL b) {
    if (a % b == 0) return ii(0, 1);
    ii T = exd_gcd(b, a % b);
    return ii( T.second, T.first - a / b * T.second);
}

```

● LUCAS

```

const int N = 100000;
ll n, m, p = 24851, fac[N];
void init() {
    int i;
    fac[0] = 1;
    for(i = 1; i <= p; i++)
        fac[i] = fac[i-1]*i % p;
}
ll q_pow(ll a, ll b) {
    ll ans = 1;
    while(b) {
        if(b & 1) ans = ans * a % p;
        b >>= 1;
        a = a*a % p;
    }
    return ans;
}
ll C(ll n, ll m) {
    if(m > n) return 0;
    return fac[n]*q_pow(fac[m]*fac[n-m], p-2) % p;
}

ll Lucas(ll n, ll m) {
    if(m == 0) return 1;
    else return (C(n/p, m/p)*Lucas(n/p, m/p))%p;
}

```

● Miller Rabin

```

inline long long mod_mul(long long a, long long b, long
long m){
    a%=m,b%=m;
    long long y=(long long)((double)a*b/m+0.5);/*
    fast for m < 2^58 */
    long long r=(a*b-y*m)%m;
    return r<0?r+m:r;
}

template<typename T>
inline T pow(T a,T b,T mod){/*a^b%mod
    T ans=1;
    for(;b;a=mod_mul(a,a,mod),b>>=1)
        if(b&1)ans=mod_mul(ans,a,mod);
    return ans;
}

int sprp[3]={2,7,61};/*int%d^3<=i.N
int
llsprp[7]={2,325,9375,28178,450775,9780504,1795265022}
;/*!Üœunsigned long long%d^3<=
template<typename T>
inline bool isprime(T n,int *sprp,int num){
    if(n==2)return 1;
    if(n<2||n%2==0)return 0;
    int t=0;
    T u=n-1;
    for(;u%2==0;++t)u>>=1;
    for(int i=0;i<num;++i){
        T a=sprp[i]%n;
        if(a==0||a==1||a==n-1)continue;
        T x=pow(a,u,n);
        if(x==1||x==n-1)continue;
        for(int j=0;j<t;++j){
            x=mod_mul(x,x,n);
            if(x==1)return 0;
            if(x==n-1)break;
        }
        if(x==n-1)continue;
        return 0;
    }
    return 1;
}

```

● Formulas

尤拉函數: $\phi(n) = n \times \prod_{p|n} (1 - 1/p)$, 對 n 之所有質因數 p

費馬小定理: $a * a^{(p-2)} = 1 \pmod{p}$, a, p 互質

次方同餘定理: $a^k \pmod{p} = (a \pmod{p})^{(k \pmod{p-1})}$ p 是質數

枚舉擴展歐幾里得之解:

若 x_0, y_0 為 $a*x + b*y = k$ 之一組解, 則

$x = x_0 + t*b/\gcd(a, b)$, $y = y_0 + t*a/\gcd(a, b)$ 亦為解, t

為整數

最大獨立集: 點的集合, 其內點不相鄰

最小點覆蓋: 點的集合, 所有邊都被覆蓋

最大匹配: 邊的集合, 其內邊不共用點

最小邊覆蓋: 邊的集合, 所有點都被覆蓋

最大獨立集+最小點覆蓋= V (數值)

最大匹配+最小邊覆蓋= V (數值)

最大匹配=最大流(二分圖)

最大匹配=最小點覆蓋(二分圖)

最小點覆蓋+最小邊覆蓋= V (數值, 二分圖)