Contents

1 Setting

1.1 /.vimrc

```
syntax on
color torte
set nu ts=4 sw=4 ai mouse=a bs=2 ci hls  ru nocp
    showmatch ar fencs=utf-8
set guifont=Consolas:h10
filetype plugin indent on
so $VIMRUNTIME/mswin.vim
behave mswin

autocmd CursorMoved  * exe printf('match VisualNOS /\V
    \<%s\>/', escape(expand('<cword>'), '/\'))
autocmd CursorMovedi * exe printf('match VisualNOS /\V
    \<%s\>/', escape(expand('<cword>'), '/\'))

map <F5> :r ~/sample.cpp<CR>
map <F9> :call Compile()<CR>
map! <F9> <ESC>:call Compile()<CR>
map <F10> :call Run()<CR>
map! <F10> <ESC>:call Run()<CR>

func! Compile()
    exec "w"
    exec "!g++ -Wall -Wshadow -std=gnu++0x % -o %< 2>
        log.txt"
    exe "cg log.txt"
    cw 5
endfunc

func! Run()
    exec "!./%<"  # "!%<" if windows
endfunc

cd ~/Desktop # C:\Users\???\Desktop
```

2 Basic

2.1 /buglist

```
/*
cmp 不能 return true
變數宣告在迴圈費時，要小心使用
<<運算小心溢位，good way: (1LL << x)
prime_table小心i,j溢位
 */
```

2.2 Builtin

```
— Built-in Function: int __builtin_ffs (T x)

Returns one plus the index of the least significant 1-
    bit of x, or if x is zero, returns zero.
返回右起第一个‘1’的位置。

— Built-in Function: int __builtin_clz (T x)

Returns the number of leading 0-bits in x, starting at
    the most significant bit position. If x is 0, the
    result is undefined.
返回左起第一个‘1’之前0的个数。

— Built-in Function: int __builtin_ctz (T x)

Returns the number of trailing 0-bits in x, starting at
     the least significant bit position. If x is 0, the
     result is undefined.
返回右起第一个‘1’之后的0的个数。

— Built-in Function: int __builtin_popcount (T x)

Returns the number of 1-bits in x.
返回‘1’的个数。

— Built-in Function: int __builtin_parity (T x)

Returns the parity of x, i.e. the number of 1-bits in x
     modulo 2.
返回‘1’的个数的奇偶性。

T is unsigned, unsigned long, unsigned long long
```

2.3 BinarySearch

```
lower_bound(a, a+n, k); //最左邊 ≥ k 的位置
upper_bound(a, a+n, k); //最左邊 > k 的位置
upper_bound(a, a+n, k) - 1; //最右邊 ≤ k 的位置
lower_bound(a, a+n, k) - 1; //最右邊 < k 的位置
[lower_bound, upper_bound) //等於 k 的範圍
equal_range(a, a+n, k);
```

2.4 int128

```
istream &operator >> (istream &is, __int128 &x) {
    char buf[30];
    is >> buf;
    bool minus = false;
    int len = strlen(buf);
    x = 0;
    for (int i=0; i<len; i++) {
        if (i==0 && buf[i]=='-') minus = true;
        else x = x*10 + buf[i] - 48;
    }
    if (minus) x*=-1;
    return is;
}
ostream &operator << (ostream &os, __int128 &x) {
    vector<int> v;
    __int128 tmp = x;
    bool minus = tmp < 0;
    if (minus) tmp *= -1;

    while(tmp > 0) {
        v.push_back(tmp%10);
        tmp/=10;
    }
    if (minus) os << "-";
    for (int i=(int)v.size()-1; i>=0; i--) os << v[i];
    return os;
}
```

2.5 Mergesort

```
long long sol(int L, int R) {
  if (R - L <= 1)return 0;
  int M = (R + L) / 2;
  long long ans = sol(L, M) + sol(M, R);
  int i = L, j = M, k = L;
  while (i < M || j < R) {
    if (i >= M)
      buf[k] = arr[j++];
    else if (j >= R)
      buf[k] = arr[i++];
    else {
      if (arr[i]<=arr[j])
        buf[k] = arr[i++];
      else {
        buf[k] = arr[j++];
        ans += M - i;
      }
    }
    k++;
  }
  for (int k = L; k < R; k++) arr[k] = buf[k];
  return ans;
}
```

2.6 ThreeSearch

```
#include <bits/stdc++.h>
using namespace std;
#define N 20
int t,n,i,j;
struct happy{
  double a,b,c;
}h[N];
double f2(double x,double a,double b,double c){return a
    *(x-b)*(x-b)+c;}
double f(double x){
  double ans=0;
  for(int i=0;i<n;i++){
    ans=max(ans,f2(x,h[i].a,h[i].b,h[i].c));
//    cout<<ans<<'\n';
  }
  return ans;
}
int main(){
  cin.tie(NULL);
  for(cin>>t;i<t;i++){
    for(cin>>n,j=0;j<n;j++)
      cin>>h[j].a>>h[j].b>>h[j].c;
    double L=0,R=300,M,MM;
    while(R-L>1e-9){
      M=L+(R-L)/3;
      MM=(M+R)/2;
//      cout<<L<<' '<<M<<' '<<MM<<' '<<R<<'\n';
      if(f(M)>f(MM))L=M;
      else R=MM;
    }
    cout<<fixed<<setprecision(5)<<f(L)<<'\n';
  }
}
```

3 Data and Structure

3.1 Disjoint Set

```
void init(){for (int i = 0; i < N; i++)p[i] = i;}
int find(int x){return x == p[x] ? x : p[x]=find(p[x])
    ;}
void Union(int a, int b){p[find(a)] = find(b);}
```

3.2 Segment Tree

```
int bulit(int L,int R,int x) {
  if(L==R)return heap[x - 1]=arr[L];
  int M=(L+R)>>1;
  return heap[x-1]=bulit(L, M, (x << 1))+bulit(M + 1, R
      , (x << 1) + 1);
}
void modify(int L,int R,int x,int a,int b,int mo) {
    if(b<L||R<a)return;
  if(L==R){heap[x-1]+=mo; return;}
  int M=(L+R)>>1;
  modify(L,M,(x<<1),a,b,mo);
  modify(M+1,R,(x<<1)+1,a,b,mo);
  heap[x - 1] += mo;
  return;
}
int quest(int L,int R,int x,int a,int b) {
    if(b<L||R<a)return 0;
  if(a<=L&&R<=b)return heap[x - 1];
  int M=(L+R)>>1;
  return quest(L,M,(x<<1),a,b)+quest(M+1,R,(x<<1)+1,a,b
      );
}
```

3.3 Treap

```
struct Treap{
  Treap *l, *r;
  int val, key, pri;
  Treap(int _val, int _key) :
    val(_val), key(_key), l(NULL), r(NULL), pri(rand())
        {}
  Treap(){};
};
Treap* merge(Treap* a, Treap* b){
  if (!a || !b)return a ? a : b;
  if (a->pri > b->pri){
    a->r = merge(a->r, b);
    return a;
  }else{
    b->l = merge(a, b->l);
    return b;
  }
}
void split(Treap* t, int k, Treap *&a, Treap *&b){
  if (!t)a = b = NULL;
  else if (t->key <= k){
    a = t;
    split(t->r, k, a->r, b);
  }else {
    b = t;
    split(t->l, k, a, b->l);
  }
  return;
}
Treap* insert(Treap* t, int k){
  Treap *tl, *tr;
  split(t, k, tl, tr);
  return merge(tl, merge(new Treap(k, ti++), tr));
}
Treap* remove(Treap* t, int k){
  Treap *tl, *tr;
  split(t, k - 1, tl, t);
  split(t, k, t, tr);
  return merge(tl, tr);
}
```

4 DP

4.1 CounterLine

```
#include <bits/stdc++.h>
using namespace std;
const int N=1<<15;
int n,m,cur;
long long int dp[2][N];

void update(int a,int b){
    if(b&(1<<m)){
        dp[cur][b^(1<<m)]+=dp[1-cur][a];
    }
}

int main(){
    while(cin>>n>>m){
        if((n*m)&1){
            cout<<"0\n";
            continue;
        }
        if(n==1||m==1){
            cout<<"1\n";
            continue;
        }
        if(n<m)swap(n,m);
        memset(dp,0,sizeof(dp));
        cur=0;
        dp[0][(1<<m)-1]=1;
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                cur^=1;
                memset(dp[cur],0,sizeof(dp[cur]));
                for(int k=0;k<(1<<m);k++){
                    update(k,k<<1);
                    if(i&&!(k&(1<<m-1)))update(k,(k<<1)
                        ^(1<<m)^1);
                    if(j&&!(k&1))update(k,(k<<1)^3);
                }
            }
        }
        cout<<dp[cur][(1<<m)-1]<<'\n';
    }
}
```

4.2 LCS

```
#include <bits/stdc++.h>
using namespace std;

int main() {
  int n, m;
  vector<int>a, b, dp[2];
  cin >> n >> m;
  a.resize(n);
  b.resize(m);
  for(int i=0;i<a.size();i++){
    cin>>a[i];
  }
  for(int i=0;i<b.size();i++){
    cin>>b[i];
  }
  dp[0].resize(m+1);
  dp[1].resize(m+1);
  for(int i=1;i<=n;i++){
    for(int j=1;j<=m;j++){
      if(a[i-1]==b[j-1])dp[i&1][j]=dp[(i&1)^1][j-1]+1;
      else dp[i&1][j]=max(dp[i&1][j-1],dp[(i&1)^1][j]);
    }
  }
  cout<<dp[n&1][m]<<'\n';
}
```

4.3 LIS

```
#include <bits/stdc++.h>
using namespace std;

int main(){
    int n;
    while(cin>>n){
        vector<int>v;
        for(int i=0,x;i<n;i++){
            cin>>x;
            if(!v.size()||x>v.back())v.push_back(x);
            else *lower_bound(v.begin(), v.end(),x)=x;
        }
        cout<<v.size()<<'\n';
    }
}
```

4.4 TSP

```
void btb(int &x){
  x=0;
  for(int i=0,j=1;i<n;i++,j*=2)x+=b[i]*j;
  return;
}
int main(){
  memset(dp,0,sizeof(dp));
    for(int i=1,st;i<=n;i++){//st:state
        for(int jj=0;jj<n;jj++)b[n-jj-1]=(jj<i);
        do{
            btb(st);
            for(int x=0;x<n;x++){
                if(!b[x])continue;
                if(i==1)dp[x][st]=dis[x][0];
                for(int y=0;y<n;y++){
                    if(x!=y&&b[y]&&(dp[x][st]==0||dp[x
                        ][st]>dp[y][st-(1<<x)]+dis[y][x
                        ])){
                        dp[x][st]=dp[y][st-(1<<x)]+dis[
                            y][x];
                    }
                }
            }
        }while(next_permutation(b,b+n));
    }
    cout<<dp[0][(1<<n)-1]<<'\n';
}
```

5 Graph

5.1 Articulation Point

```
vector<int>v[N],bcc[N];//clear
LL dep[N],low[N],bccno[N],time_cnt,bcc_cnt;//set dep
    low -1 else 0
bitset<N>is_AP;//0
struct Edge{int s,t;};
stack<Edge>st;//clear
int dfs(int s,int fa){
    int child=0;
    dep[s]=low[s]=time_cnt++;
    for(auto t:v[s]){
        Edge e=(Edge){s,t};
        if(dep[t]==-1){
            st.push(e);
            child++;
            dfs(t,s);
            low[s]=min(low[s],low[t]);
            if(dep[s]<=low[t]){
                is_AP[s]=1;
                bcc_cnt++;
                bcc[bcc_cnt].clear();
                while(1){
                    Edge x=st.top(); st.pop();
                    if(bccno[x.s]!=bcc_cnt){
                        bcc[bcc_cnt].push_back(x.s);
                        bccno[x.s]=bcc_cnt;
                    }
                    if(bccno[x.t]!=bcc_cnt){
                        bcc[bcc_cnt].push_back(x.t);
                        bccno[x.t]=bcc_cnt;
                    }
                    if(x.s==s&&x.t==t)break;
                }
            }
        }else if(low[s]>dep[t]){
            st.push(e);
            low[s]=dep[t];
        }
    }
    if(fa<0&&child==1)is_AP[s]=0;
    return low[s];
}
```

5.2 BellmanFord

```
struct Edge{
    int t, w;
};
int v, e;
int d[N], cnt[N];
bitset<N> inq;
queue<int>Q;
vector<Edge>G[N];

void addEdge(int from, int to, int w){
    G[from].push_back({to,w});
}

bool hasnegativeCycle(){
    while(!Q.empty())Q.pop();
    for(int i = 1; i <= v;i++){
        inq[i] = true;
        cnt[i] = d[i] = 0;
        Q.push(i);
    }
    while(!Q.empty()){
        int s = Q.front(); Q.pop();
        inq[s] = false;
        for(Edge it: G[s]){
            if(d[it.t] > d[s] + it.w){
                d[it.t] = d[s] + it.w;
                if(inq[it.t])continue;
                Q.push(it.t);
                inq[it.t] = true;
                if(++cnt[it.t] > v)return true;
            }
        }
    }
    return false;
}
```

5.3 Bipartite

```
#include <iostream>
#include <vector>
#include <stack>
#include <cstring>

#define S 50050

using namespace std;

vector<int> map[S];
int visit[S];
bool valid;

void check(int start) {
    stack<int> st;
    st.push(start);
    visit[start] = 1;

    while(valid && !st.empty()) {
        int cur = st.top();
        st.pop();

        for(int i = 0; i < map[cur].size(); i++) {
            int next = map[cur][i];

            if(visit[next] == -1) {
                st.push(next);

                if(visit[cur] == 1) visit[next] = 2;
                else visit[next] = 1;
            }
            else if(visit[cur] == visit[next]) valid =
                false;
        }
    }
}

int main() {
    int n, m;
    cin >> n >> m;

    for(int i = 0; i < m; i++) {
        int a, b;
        cin >> a >> b;

        map[a].push_back(b);
        map[b].push_back(a);
    }

    // -1 : not visit, 1 : tsudere, 2 : proud
    memset(visit, -1, sizeof(visit));
    valid = true;

    for(int i = 1; i <= n; i++) {
        if(valid && visit[i] == -1) {
            check(i);
        }
    }
```

```cpp
    if(valid) cout << "yes" << endl;
    else cout << "no" << endl;

    return 0;
}
```

5.4 dijkstra

```cpp
struct Edge{
    int from,to,w;
};
vector<Edge>E;
vector<int>v[N];
bitset<N> vis;
void init(){
    E.clear();
    for(int i=0;i<N;i++){
        v[i].clear();
    }
}

void addEdge(int from,int to,int w){
    v[from].push_back(E.size());
    E.push_back(Edge{from,to,w});
}

void dijkstra(int s,int d[],int p[]){// set d[] INF &&
    set p[] -1
    d[s]=0;
    priority_queue<PII,vector<PII>,greater<PII>>pq;
    vis.reset();
    pq.push(MP(d[s],s));
    while(!pq.empty()){
        PII k=pq.top(); pq.pop();
        if(vis[k.second])continue;
        vis[k.second]=true;
        for(auto it:v[k.second]){
            Edge e=E[it];
            if(d[e.to]>d[e.from]+e.w){
                d[e.to]=d[e.from]+e.w;
                p[e.to]=e.from;
                pq.push(MP(d[e.to],e.to));
            }
        }
    }
}
```

5.5 Convex Hull

```cpp
struct loc {
  int x, y;
  loc() {};
  loc(int x, int y): x(x), y(y) {}
  bool operator <(const loc& b)const {return x != b.x ?
       x < b.x : y < b.y;}
  bool operator ==(const loc& b)const {return x == b.x
      && y == b.y;}
  loc operator -(const loc& b)const {return loc(x - b.x
      , y - b.y);}
  int cross(const loc& b)const {return x * b.y - y * b.
      x;}
  int dis(loc a, loc b) {return (x - b.x) * (x - b.x) +
       (y - b.y) * (y - b.y);}
};
vector<loc>p, p1;
int n;
void convexhull() {
  sort(p.begin(), p.end());
  p.erase(unique(p.begin(), p.end()), p.end());
  p1.clear();
  p1.resize(p.size());
  int m = 0;
  for (int i = 0; i < p.size(); i++) {
    while (m > 1 && (p1[m - 1] - p1[m - 2]).cross(p[i]
        - p1[m - 2]) <= 0)m--;
    p1[m++] = p[i];
  }
  int k = m;
  for (int i = p.size() - 2; i >= 0; i--) {
    while (m > k && (p1[m - 1] - p1[m - 2]).cross(p[i]
        - p1[m - 2]) <= 0)m--;
    p1[m++] = p[i];
  }
  if (n > 1)m--;
  p1.resize(m);
}
```

5.6 Dinic

```cpp
struct dinic{
  static const int M = 10000;
  static const int INF = 1e9;
  struct Edge{
    int v;
    int f; //residual flow
    int re;
  };
  int n, s, t, level[M], now[M];
  vector<Edge> e[M];
  void init(int _n, int _s, int _t){
    n = _n; s = _s; t = _t;
    for (int i = 0; i <= n; i++)e[i].clear();
  }
  void add_edge(int u, int v, int f){
    e[u].push_back({ v, f, (int)e[v].size() });
    e[v].push_back({ u, f, (int)e[u].size() - 1 });
  }
  bool bfs(){
    fill(level, level + n + 1, -1);
    queue<int> q;
    q.push(s); level[s] = 0;
    while (!q.empty()){
      int u = q.front(); q.pop();
      for (auto it : e[u]){
        if (it.f > 0 && level[it.v] == -1){
          level[it.v] = level[u] + 1;
          q.push(it.v);
        }
      }
    }
    return level[t] != -1;
  }
  int dfs(int u, int nf){
    if (u == t)return nf;
    int res = 0;
    while (now[u] < e[u].size()){
      Edge &it = e[u][now[u]];
      if (it.f>0 && level[it.v] == level[u] + 1){
        int tf = dfs(it.v, min(nf, it.f));
        res += tf; nf -= tf; it.f -= tf;
        e[it.v][it.re].f += tf;
        if (nf == 0)return res;
      }
      else now[u]++;
    }
    if (!res)level[u] = -1;
    return res;
  }
  int flow(int res = 0){
    while (bfs()){
      int temp;
      memset(now, 0, sizeof(now));
      while (temp = (dfs(s, INF))){
        res += temp;
      }
    }
    return res;
  }
}d;
```

5.7 FloydWarshall

```cpp
#include <iostream>

#define INF 1e9
#define LL long long

using namespace std;

int main() {
    int n;

    while(cin >> n) {
        LL dis[n][n];
        LL ans = INF;

        for(int i = 0; i < n; i++)
            for(int j = 0; j < n; j++) {
                cin >> dis[i][j];
                if(dis[i][j] == 0) dis[i][j] = INF;
            }

        for(int i = 0; i < n; i++) {
            for(int j = 0; j < n; j++) {
                if(i == j) continue;
                ans = min(ans, dis[i][j] + dis[j][i]);
                for(int k = 0; k < n; k++) {
                    dis[i][j] = min(dis[i][j], dis[i][k
                        ] + dis[k][j]);

                    ans = min(ans, dis[i][j] + dis[k][i
                        ] + dis[j][k]);
                }
            }
        }

        if(ans == INF) cout << -1 << endl;
        else cout << ans << endl;
    }

    return 0;
}
```

5.8 KM

```cpp
int n;
int Left[N];
double w[N][N], Lx[N], Ly[N];
bitset<N> vx, vy;

bool match(int i) {
  vx[i] = true;
  for (int j = 1; j <= n; j++) {
    if ((fabs(Lx[i] + Ly[j] - w[i][j]) < 1e-9) && !vy[j
        ]) {
      vy[j] = true;
      if (!Left[j] || match(Left[j])) {
        Left[j] = i;
        return true;
      }
    }
  }
  return false;
}

void update() {
  double a = 1e30;
  for (int i = 1; i <= n; i++) {
    if (vx[i])for (int j = 1; j <= n; j++) {
        if (!vy[j])a = min(a, Lx[i] + Ly[j] - w[i][j]);
      }
  }
  for (int i = 1; i <= n; i++) {
    if (vx[i])Lx[i] -= a;
    if (vy[i])Ly[i] += a;
  }
}

void KM() {
  for (int i = 1; i <= n; i++) {
    Left[i] = Lx[i] = Ly[i] = 0;
    for (int j = 1; j <= n; j++) {
      Lx[i] = max(Lx[i], w[i][j]);
    }
  }
  for (int i = 1; i <= n; i++) {
    while (true) {
      vx.reset(); vy.reset();
      if (match(i))break;
      update();
    }
  }
}
```

5.9 Longest Common Ancestor

```cpp
void preprocess() {
  for (int i = 1; i <= 25; i++) {
    for (int j = 1; j <= n; j++) {
      if (par[j][i - 1] == -1 || par[par[j][i - 1]][i -
           1] == -1)continue;
      par[j][i] = par[par[j][i - 1]][i - 1];
    }
  }
}
```

5.10 MST

```cpp
#include <iostream>
#include <vector>
#include <stack>
#include <cstring>
#include <algorithm>

#define LL long long
#define MAX 1e11
#define S 50050
using namespace std;

int n, m;
int sum;

typedef struct {
    int a, b, l;
} edge;
bool cmp(edge l, edge r) { return l.l < r.l; }

vector<edge> v;

typedef struct {
    int d;
    LL l;
} node;

vector<node> map[S];

int disjoint[S];

int root(int x) {
    if(disjoint[x] < 0) return x;
    else {
        disjoint[x] = root(disjoint[x]);
        return disjoint[x];
    }
}

bool same(int a, int b) {
    return root(a) == root(b);
}

void connect(int a, int b) {
    // cout << "CONNECT " << a << " " << b << endl;
    int ra = root(a);
    int rb = root(b);

    disjoint[ra] += disjoint[rb];
    disjoint[rb] = ra;
}

void kruskal() {
    int remain = n - 1;
    for(auto i : v) {
        if(remain == 0) break;

        if(!same(i.a, i.b)) {
            connect(i.a, i.b);

            map[i.a].push_back((node){i.b, i.l});
            map[i.b].push_back((node){i.a, i.l});

            sum += i.l;
            remain--;
        }
    }
}

bool book[S];

void dfs(int start) {
    stack<int> st;
    st.push(start);


    memset(book, false, sizeof(book));

    while(!st.empty()) {
        int cur = st.top();
        // cout << cur << endl;
        st.pop();

        book[cur] = true;

        for(int i = 0; i < map[cur].size(); i++) {
            int next = map[cur][i].d;
            if(!book[next]) {
                st.push(next);
            }
        }
    }
}

void init() {
    memset(disjoint, -1, sizeof(disjoint));
    sum = 0;
}

bool check() {
    for(int i = 1; i <= n; i++)
        if(!book[i]) return false;

    return true;
}

int main() {
    init();

    cin >> n >> m;

    for(int i = 0; i < m; i++) {
        edge tmp;
        cin >> tmp.a >> tmp.b >> tmp.l;

        v.push_back(tmp);
    }

    sort(v.begin(), v.end(), cmp);

    kruskal();
    dfs(1);

    if(!check()) cout << -1 << endl;
    else cout << sum << endl;

    return 0;
}
```

5.11 SPFA

```cpp
#include <iostream>
#include <vector>
#include <stack>
#include <queue>
#include <cstring>

#define S 50050
#define MAX 1e11
#define LL long long

using namespace std;

typedef struct {
    int d;
    LL l;
} XXX;
vector<XXX> map[S];


LL lon[S];
int cnt[S];
int n, m;
bool cycle;
bool inqueue[S];

void dfs(int start) {
    stack<int> st;
    st.push(start);

    bool book[S];
    memset(book, false, sizeof(book));

    while(!st.empty()) {
        int cur = st.top();
        // cout << cur << endl;
        st.pop();
        lon[cur] = -MAX;
        book[cur] = true;

        for(int i = 0; i < map[cur].size(); i++) {
            int next = map[cur][i].d;
            if(!book[next]) st.push(next);
        }
    }
}

void spfa(int start) {
    memset(inqueue, false, sizeof(inqueue));
    for(int i = 0; i < S; i++) lon[i] = MAX;
    cycle = false;

    queue<int> q;
    q.push(start);
    lon[start] = 0;
    inqueue[start] = true;

    while(!q.empty()) {
        int cur = q.front();
        q.pop();
        inqueue[cur] = false;
        // cout << "AT: " << cur << " " << cnt[cur] <<
            endl;
        cnt[cur]++;
        if(cnt[cur] > n) {
            dfs(cur);
            return ;
        }

        for(int i = 0; i < map[cur].size(); i++) {
            int next = map[cur][i].d;

            if(lon[next] > lon[cur] + map[cur][i].l) {
                lon[next] = lon[cur] + map[cur][i].l;
                if(!inqueue[next] && cnt[cur] <= n) {
                    q.push(next);
                    inqueue[next] = true;
                }
            }
        }
    }
}


int main() {
    cin >> n >> m;

    for(int i = 0; i < m; i++) {
        int a, b;
        LL c;
        cin >> a >> b >> c;

        map[a].push_back((XXX) {b, c});
    }

    spfa(1);

    if(lon[n] >= MAX || lon[n] <= -MAX) cout << "QAQ"
        << endl;
    else cout << lon[n] << endl;

    return 0;
}
```

5.12 SumOfDistanceInTree

```
#include <bits/stdc++.h>
#pragma comment(linker, "/STACK:10240000,10240000")//递
    归太深，导致爆栈，所以使用扩栈语句
using namespace std;

const int N = 100009;
int dp[N] = {}, num[N];
vector<int> p[N];
bool f[N] = {};

void dfs(int s, int depth)
{
    int len = p[s].size();
    f[s] = 1;
    num[s] = 1;
    dp[1] += depth;
    for(int i=0; i<len; i++)
    {
        if(!f[p[s][i]])
        {
            dfs(p[s][i], depth+1);
            num[s] += num[p[s][i]];
        }
    }
}

void solve(int s, int n)
{
    int len = p[s].size();
    f[s] = 1;
    for(int i=0; i<len; i++)
    {
        if(!f[p[s][i]])
        {
            dp[p[s][i]] = dp[s]+n-num[p[s][i]]*2;
            solve(p[s][i], n);
        }
    }
}

int main()
{
    int n;
    scanf("%d", &n);
    for(int i=1; i<n; i++)
    {
        int a, b;
        scanf("%d%d", &a, &b);
        p[a].push_back(b);
        p[b].push_back(a);
    }
    dfs(1, 0);
    memset(f, 0, sizeof(f));
    solve(1, n);
    for(int i=1; i<=n; i++)
        printf("%d\n", dp[i]);
    return 0;
}
```

5.13 TopologicalSort

```
#include <iostream>
#include <stack>
#include <vector>
#include <cstring>

#define S 50050

using namespace std;

vector<int> map[S];
stack<int> ans;
int state[S];
bool head[S];
bool valid;
int n, m;

void dfs(int cur) {
    state[cur] = 1;

    for(auto next : map[cur])
        if(!state[next]) dfs(next);
        else if(state[next] == 1) {
            valid = false;
            return ;
        }

    state[cur] = 2;

    ans.push(cur);
}

void topology_sort() {
    for(int i = 1; i <= n; i++)
        if(valid && head[i]) dfs(i);

    if(!valid) {
        cout << -1 << endl;
        return ;
    }

    while(!ans.empty()) {
        cout << ans.top() << endl;
        ans.pop();
    }
}

int main() {
    cin >> n >> m;

    memset(head, true, sizeof(head));

    for(int i = 0; i < m; i++) {
        int a, b;
        cin >> a >> b;

        head[b] = false;

        map[a].push_back(b);
    }

    memset(state, 0, sizeof(state));
    valid = true;

    topology_sort();

    return 0;
}
```

6 Number

6.1 Catalan

$C_0 = 1$ and $C_{n+1} = \frac{2(2n+1)}{n+2} C_n,$

6.2 Combination

```
#include <bits/stdc++.h>
using namespace std;
typedef long long LL;
const int M=1000005;
int n,k;
LL m,phi;
vector <int> facs;
LL dp[M],dp2[M][32];

LL pw(LL x,LL y){
    // cout<<x<<' '<<y<<'\n';
    LL ret=1,tmp=x%m;
    while(y){
        if(y&1)ret=ret*tmp%m;
        tmp=tmp*tmp%m;
        y>>=1;
    }
    return ret;
}

void init(){
    facs.clear();
    LL x=m,sq=(LL)sqrt(m);
    phi=1;
    for(LL i=2;i<=sq;i++){
        if(x%i)continue;
        phi*=i-1; x/=i;
        facs.push_back(i);
        while(x%i==0){
            phi*=i;
            x/=i;
        }
    }
    if(x>1){
        phi*=x-1;
        facs.push_back((int)x);
    }
    k=facs.size();
    dp[0]=1;
    memset(dp2,0,sizeof(dp2));
    for(int i=1;i<M;i++){
        LL tmp=i;
        for(int j=0;j<k;j++){
            dp2[i][j]=dp2[i-1][j];
            while(tmp%facs[j]==0){
                tmp/=facs[j];
                dp2[i][j]++;
            }
        }
        dp[i]=dp[i-1]*tmp%m;
    }
    return;
}

int main(){
    while(cin>>n>>m){
        init();
        while(n--){
            LL ans=1;
            int x,y;
            cin>>x>>y;
            for(int i=0;i<k;i++){
                ans=ans*pw(facs[i],dp2[x][i]-dp2[x-y][i
    ]-dp2[y][i])%m;
            }
            ans=ans*dp[x]%m;
            ans=ans*pw(dp[y],phi-1)%m;
            ans=ans*pw(dp[x-y],phi-1)%m;
            cout<<ans<<'\n';
        }
    }
}
```

6.3 Extend Euclidean.cpp

```
int extgcd(int a,int b,int &x,int &y){
    int d=a;
    if(b){d=extgcd(b,a%b,y,x),y-=(a/b)*x;}
    else x=1,y=0;
    return d;
}//ax+by=1 ax同餘 1 mod b
```

6.4 GaussElimination

```
const int MAXN = 300;
const double EPS = 1e-8;
int n;
double A[MAXN][MAXN];
void Gauss() {
  for(int i = 0; i < n; i++) {
    bool ok = 0;
    for(int j = i; j < n; j++) {
      if(fabs(A[j][i]) > EPS) {
        swap(A[j], A[i]);
        ok = 1;
        break;
      }
    }
    if(!ok) continue;
    double fs = A[i][i];
    for(int j = i+1; j < n; j++) {
      double r = A[j][i] / fs;
      for(int k = i; k < n; k++) {
        A[j][k] -= A[i][k] * r;
      }
    }
  }
}
```

6.5 Matrix

```
template<typename T,int N=2>
struct Mat {//Matrix
  unsigned long long v[N][N];
  Mat operator*(Mat b)const {
    Mat val;
  for (int i = 0; i < N; i++) {
   for (int j = 0; j < N; j++) {
    val.v[i][j] = 0;
    for (int k = 0; k < N; k++) {
     val.v[i][j] += v[i][k] * b.v[k][j];
    }
   }
  }
  return val;
 }
};
```

6.6 Phi

```
void phi_table(int n){
  phi[1] = 1;
  for(int i = 2; i <= n; i++){
    if(phi[i])continue;
    for(int j = i; j < n; j += i){
      if(!phi[j])phi[j] = j;
      phi[j] = phi[j] / i * (i - 1);
    }
  }
}
```

6.7 Prime table

```
void PrimeTable(){
  is_notp.reset();
 is_notp[0] = is_notp[1] = 1;
 for (int i = 2; i < N; i++){
  if (is_notp[i])continue;
  p.push_back(i);
  for (int j=0;i*p[j]<N&&j<p.size();j++){
   is_notp[i*p[j]] = 1;
   if(i%p[j]==0)break;
  }
 }
}
```

7 String

7.1 KMP

```
void bulid_fail_funtion(string B, int *fail){
 int len = B.length(), current_pos;
 current_pos = fail[0] = -1;
 for (int i = 1; i<len; i++){
  while (current_pos != -1 && B[current_pos + 1] != B[i]){
   current_pos = fail[current_pos];
  }
  if (B[current_pos + 1] == B[i])current_pos++;
  fail[i] = current_pos;
 }
}
void match(string A, string B, int *fail){
 int lenA = A.length(), lenB = B.length();
 int current_pos = -1;
 for (int i = 0; i<lenA; i++){
  while (current_pos != -1 && B[current_pos + 1] != A[i]){
   current_pos = fail[current_pos];
  }
  if (B[current_pos + 1] == A[i])current_pos++;
  if (current_pos == lenB - 1){//match! A[i-lenB+1,i]=B
   current_pos = fail[current_pos];
  }
 }
}
int main(){
 int t, i;
 string s;
 for (i = 0, cin >> t; i<t; i++){
  cin >> s;
  int fail[N];
  bulid_fail_funtion(s, fail);
  int p = s.length() - 1;
  if (fail[p] != -1 && (p + 1) % (p - fail[p]) == 0) printf("%d\n", p - fail[p]);
  else printf("%d\n", p + 1);
 }
}
```

7.2 Trie

```
//init sz=1 trie[0]=0
void insert(string s){
  int u=0,v;
  for(int i=0;i<r.size();i++){
    v=r[i]-'a';
    if(!trie[u][v]){
      memset(trie[sz],0,sizeof(trie[sz]));
      val[sz]=0;
      trie[u][v]=sz++;
    }
    u=trie[u][v];
  }
  val[u]=1;
  return;
}
void search(string s,int i){
  int u=0,v;
  dp[i]=0;
  for(int j=i;j<s.size();j++){
    v=s[j]-'a';
    if(!trie[u][v])return;
    u=trie[u][v];
    if(val[u])dp[i]=(dp[i]+dp[j+1])%MOD;
  }
  return;
}
```

7.3 Zvalue

```
void z_value(){
 int lens = s.size(), l = 0, r = 0;
 z[0] = 0;
 for (int i = 1; i < lens; i++){
  if (i>r)z[i] = 0;
  else{
   int ip = i - l;
   if (ip + z[ip] < z[l])z[i] = z[ip];
   else z[i] = r - l + 1;
  }
  while (i + z[i] < lens&&s[i + z[i]] == s[z[i]])z[i]++;
  if (i + z[i] - 1 > r){
   l = i;
   r = l + z[i] - 1;
  }
 }
}
```