```
Contents
                                                           22
                                                                   cw 5
                                                           23
                                                              endfunc
                                                           24
                                                           25
                                                              func! Run()
  1 Setting
                                                                        "!./%<" # "!%<" if windows
     1.1 /.vimrc . . . . . . . . . . . . . . . .
                                                           26
                                                                  exec
                                                           27
                                                              endfunc
   2 Basic
                                                           <del>2</del>8
     cd ~/Desktop # C:\Users\???\Desktop
     2.2 Builtin .
     2.3 BinarySearch . . . . . . . . . . . . . . . .
     2.4 int128 . . . . . . . . . . . . . . . .
     2.5 Mergesort . . . . . . . . . . . . . . . . . .
     2.6 ThreeSearch .
                                                                   Basic
   3 Data and Structure
     3.1 Disjoint Set . . . . . . . . . . . . . . . .
     3.2 Segment Tree
     3.3 Treap . . . . . . . . . . . . . . . .
                                                              2.1 /buglist
  4 DP
     4.1 CounterLine . . . . . . . . . .
                                                           31 /*
     4.2 LCS . . . . . . . . . . . . . . . . .
     4.3 LIS . . . . . . . . . . . . .
                                                           32 cmp 不能 return true
                                                           33 變數宣告在迴圈費時,要小心使用
                                                           34 <<運算小心溢位,good way: (1LL << x)
  5 Graph
     5.1 Articulation Point . . . . . . . .
                                                           35
                                                              prime_table小心i,j溢位
                                                           46
     5.2 BellmanFord . . . . . . . . . . . . . .
     5.3 Bipartite . . . . . .
                                                           4
     5.5 Convex Hull . . . . . .
     5.6 Dinic . . .
     5.7 FloydWarshall . . . . .
                                                                     Builtin
     5.9 Longest Common Ancestor . . . . .
     5.10MST . . . . . . . . . . . . . . .
                                                              — Built-in Function: int __builtin_ffs (T x)
                                                           8 2
8 3
     5.12SumOfDistanceInTree . . . . . . . . . . .
     5.13TopologicalSort . . . . . . . . . . . . . .
                                                              Returns one plus the index of the least significant 1-
                                                                   bit of x, or if x is zero, returns zero.
   6 Number
     6.1 Catalan . . . . . . . . . . . .
                                                              返回右起第一个'1'的位置。
     6.2 Combination . . . . . .
     6.3 Extend Euclidean.cpp . . . . . .
                                                           96
                                                               — Built-in Function: int __builtin_clz (T x)
     6.4 GaussElimination . . . . .
     6.5 Matrix . . . . .
                                                           97
                                                              Returns the number of leading 0-bits in x, starting at
     6.7 Prime table . . . . . . .
                                                                   the most significant bit position. If x is 0, the
                                                                   result is undefined.
  7 String
                                                           10
                                                              返回左起第一个'1'之前0的个数。
                                                          109
     7.1 KMP .
     7.2 Trie
                                                          100
     7.3 Zvalue . . . .
                                                          <sup>1</sup>91
                                                               — Built-in Function: int __builtin_ctz (T x)
                                                           12
                                                              Returns the number of trailing \theta-bits in x, starting at
                                                           13
       Setting
                                                                    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。
                                                           14
  1.1 /.vimrc
                                                           15
                                                               — Built-in Function: int __builtin_popcount (T x)
                                                           16
                                                           17
1 syntax on
                                                           18
                                                              Returns the number of 1-bits in x.
2 color torte
                                                           19
                                                              返回'1'的个数。
3 set nu ts=4 sw=4 ai mouse=a bs=2 ci hls ru nocp
                                                           20
       showmatch ar fencs=utf-8
                                                           21
                                                               — Built-in Function: int __builtin_parity (T x)
4 set guifont=Consolas:h10
                                                           22
5 filetype plugin indent on
                                                              Returns the parity of x, i.e. the number of 1-bits in x
                                                           23
6 so $VIMRUNTIME/mswin.vim
                                                                    modulo 2.
7
  behave mswin
                                                           24
                                                              返回'1'的个数的奇偶性。
8
  autocmd CursorMoved * exe printf('match VisualNOS /\V
                                                           25
9
                                                           26 T is unsigned, unsigned long, unsigned long long
       \<%s\>/', escape(expand('<cword>'), '/\'))
10 autocmd CursorMovedi * exe printf('match VisualNOS /\V
       \<%s\>/', escape(expand('<cword>'), '/\'))
11
12 map <F5> :r ~/sample.cpp<CR>
                                                              2.3 BinarySearch
13 map <F9> :call Compile()<CR>
14 map! <F9> <ESC>:call Compile()<CR>
15 map <F10> :call Run()<CR>
                                                            1|lower_bound(a, a+n, k); //最左邊 ≥ k 的位置
16 map! <F10> <ESC>:call Run()<CR>
                                                            2 upper_bound(a, a+n, k); //最左邊 > k 的位置
17
                                                            3 upper_bound(a, a+n, k) - 1; //最右邊 ≤ k 的位置
18 func! Compile()
                                                            4 lower_bound(a, a+n, k) - 1; //最右邊 < k 的位置
19
      exec "w"
                                                            5|[lower_bound, upper_bound) //等於 k 的範圍
       exec "!g++ -Wall -Wshadow -std=gnu++0x % -o %< 2>
20
                                                            6 equal_range(a, a+n, k);
           Log.txt"
```

21

exe "cg log.txt"

2.4 int128

```
1 istream & operator >> (istream &is, __int128 &x) {
2
        char buf[30];
3
        is >> buf;
4
        bool minus = false;
5
        int len = strlen(buf);
7
       for (int i=0; i<len; i++) {
   if (i==0 && buf[i]=='-') minus = true;</pre>
8
9
            else x = x*10 + buf[i] - 48;
10
11
        if (minus) x*=-1;
12
        return is;
13|}
14 ostream & operator << (ostream &os, __int128 &x) {
15
        vector<int> v;
16
         _{int128} tmp = x;
17
        bool minus = tmp < 0;</pre>
18
        if (minus) tmp *= -1;
19
20
        while(tmp > 0) {
            v.push_back(tmp%10);
21
22
            tmp/=10;
23
        if (minus) os << "-":
24
25
26
        return os;
27 }
```

2.5

Mergesort

```
1|long long sol(int L, int R) {
     if (R - L <= 1)return 0;</pre>
3
     int M = (R + L) / 2;
     long long ans = sol(L, M) + sol(M, R);
     int i = L, j = M, k = L;
5
     while (i < M \mid \mid j < R) {
6
7
       if (i >= M)
8
          buf[k] = arr[j++];
9
       else if (j >= R)
10
          buf[k] = arr[i++];
11
       else {
          if (arr[i]<=arr[j])</pre>
12
13
            buf[k] = arr[i++];
14
15
            buf[k] = arr[j++];
16
            ans += M - i;
17
          }
18
19
       k++;
20
21
     for (int k = L; k < R; k++) arr[k] = buf[k];</pre>
22
     return ans;
23 | }
```

2.6 ThreeSearch

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 #define N 20
4 int t,n,i,j;
5 struct happy{
     double a,b,c;
7|}h[N];
8 double f2(double x, double a, double b, double c){return a 7
       *(x-b)*(x-b)+c;}
9 double f(double x){
10
     double ans=0;
     for(int i=0;i<n;i++){</pre>
11
       ans=max(ans,f2(x,h[i].a,h[i].b,h[i].c));
12
13 //
         cout<<ans<<'\n';</pre>
14
    }
```

```
15
     return ans;
16 }
17
   int main(){
     cin.tie(NULL);
18
19
     for(cin>>t;i<t;i++){</pre>
20
       for(cin>>n,j=0;j<n;j++)</pre>
21
          cin>>h[j].a>>h[j].b>>h[j].c;
22
       double L=0,R=300,M,MM;
23
       while(R-L>1e-9){
24
          M=L+(R-L)/3;
25
          MM=(M+R)/2;
            cout<<L<<' '<<M<<' '<<MM<<' '<<R<<'\n';
26
27
          if(f(M)>f(MM))L=M;
28
          else R=MM;
29
30
       cout<<fixed<<setprecision(5)<<f(L)<<'\n';</pre>
31
32 }
```

3 Data and Structure

3.1 Disjoint Set

3.2 Segment Tree

```
1 int bulit(int L,int R,int x) {
     if(L==R)return heap[x - 1]=arr[L];
 3
     int M=(L+R)>>1;
 4
     return heap[x-1]=bulit(L, M, (x << 1))+bulit(M + 1, R)
         (x << 1) + 1);
   }
 5
 6
   void modify(int L,int R,int x,int a,int b,int mo) {
 7
       if(b<L||R<a)return;</pre>
 8
     if(L==R){heap[x-1]+=mo; return;}
 9
     int M=(L+R)>>1;
     modify(L,M,(x<<1),a,b,mo);</pre>
10
     modify(M+1,R,(x<<1)+1,a,b,mo);
12
     heap[x - 1] += mo;
13
     return;
14
15
   int quest(int L,int R,int x,int a,int b) {
       if(b<L||R<a)return 0;</pre>
17
     if(a<=L&&R<=b)return heap[x - 1];</pre>
18
     int M=(L+R)>>1;
19
     return quest(L,M,(x<<1),a,b)+quest(M+1,R,(x<<1)+1,a,b
          );
20 }
```

3.3 Treap

```
1 struct Treap{
2
     Treap *1, *r;
     int val, key, pri;
3
 4
     Treap(int _val, int _key) :
5
       val(_val), key(_key), l(NULL), r(NULL), pri(rand())
    Treap(){};
  };
8
  Treap* merge(Treap* a, Treap* b){
 9
     if (!a || !b)return a ? a : b;
10
     if (a->pri > b->pri){
       a->r = merge(a->r, b);
11
12
       return a;
13
     }else{
       b->l = merge(a, b->l);
```

```
15
       return b;
16
     }
17
  }
18
  void split(Treap* t, int k, Treap *&a, Treap *&b){
19
     if (!t)a = b = NULL;
20
     else if (t->key <= k){
21
       a = t:
22
       split(t->r, k, a->r, b);
23
     }else {
24
25
       split(t->1, k, a, b->1);
26
27
     return;
28
  }
29
   Treap* insert(Treap* t, int k){
     Treap *tl, *tr;
30
31
     split(t, k, tl, tr);
32
     return merge(tl, merge(new Treap(k, ti++), tr));
33
34
   Treap* remove(Treap* t, int k){
     Treap *tl, *tr;
35
36
     split(t, k - 1, tl, t);
37
     split(t, k, t, tr);
     return merge(tl, tr);
38
39 }
```

4.2 LCS

```
1 #include <bits/stdc++.h>
   using namespace std;
 4
   int main() {
 5
     int n, m:
 6
     vector<int>a, b, dp[2];
 7
     cin >> n >> m;
     a.resize(n);
     b.resize(m);
10
     for(int i=0;i<a.size();i++){</pre>
11
       cin>>a[i];
12
13
     for(int i=0;i<b.size();i++){</pre>
14
       cin>>b[i];
15
16
     dp[0].resize(m+1);
17
     dp[1].resize(m+1);
18
     for(int i=1;i<=n;i++){</pre>
       for(int j=1;j<=m;j++){</pre>
19
          if(a[i-1]==b[j-1])dp[i&1][j]=dp[(i&1)^1][j-1]+1;
20
21
          else dp[i&1][j]=max(dp[i&1][j-1],dp[(i&1)^1][j]);
22
23
24
     cout<<dp[n&1][m]<<'\n';
25
```

4 DP

4.1 CounterLine

1 #include <bits/stdc++.h>

```
2 using namespace std;
 3 const int N=1<<15;
   int n,m,cur;
 5 long long int dp[2][N];
 7
   void update(int a,int b){
8
        if(b&(1<<m)){
9
             dp[cur][b^(1<<m)]+=dp[1-cur][a];</pre>
10
11
  }
12
   int main(){
13
14
        while(cin>>n>>m){
15
            if((n*m)&1){
                 cout<<"0 \ n";
16
17
                 continue;
18
19
             if(n==1||m==1){
                 cout<<"1\n";
20
21
                 continue;
22
23
            if(n<m)swap(n,m);</pre>
24
            memset(dp,0,sizeof(dp));
25
            cur=0:
26
            dp[0][(1<< m)-1]=1;
27
            for(int i=0;i<n;i++){</pre>
                                                                   10
28
                 for(int j=0;j<m;j++){</pre>
                                                                   11
29
                      cur^=1;
                                                                   12
30
                      memset(dp[cur],0,sizeof(dp[cur]));
                                                                   13
31
                      for(int k=0;k<(1<<m);k++){</pre>
                                                                   14
32
                           update(k,k<<1);
                                                                    15
                           if(i\&\&!(k\&(1<< m-1)))update(k,(k<<1)_{16}
33
                                ^(1<<m)^1);
34
                           if(j&&!(k&1))update(k,(k<<1)^3);</pre>
35
                                                                   17
36
                 }
37
                                                                   18
38
            cout<<dp[cur][(1<<m)-1]<<'\n';
                                                                   19
39
        }
                                                                    20
40|}
                                                                   21
```

4.3 LIS

```
1 #include <bits/stdc++.h>
   using namespace std;
 3
 4
   int main(){
 5
       int n:
 6
        while(cin>>n){
 7
            vector<int>v;
 8
            for(int i=0,x;i<n;i++){</pre>
                cin>>x;
 9
10
                if(!v.size()||x>v.back())v.push_back(x);
11
                else *lower_bound(v.begin(), v.end(),x)=x;
12
13
            cout<<v.size()<<'\n';</pre>
14
15
   }
```

4.4 TSP

```
void btb(int &x){
1
2
3
    for(int i=0,j=1;i<n;i++,j*=2)x+=b[i]*j;</pre>
4
5
6
  int main(){
7
    memset(dp,0,sizeof(dp));
8
      for(int i=1,st;i<=n;i++){//st:state</pre>
9
           for(int jj=0;jj<n;jj++)b[n-jj-1]=(jj<i);</pre>
           do{
                btb(st);
                for(int x=0;x<n;x++){</pre>
                    if(!b[x])continue;
                    if(i==1)dp[x][st]=dis[x][0];
                    for(int y=0;y<n;y++){</pre>
                         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));
      }
```

```
Zappers
                                                                                                       (March 30, 2019) 4
23
       cout<<dp[0][(1<<n)-1]<<'\n';
                                                              22
                                                                          int s = Q.front(); Q.pop();
24 }
                                                              23
                                                                          inq[s] = false;
                                                              24
                                                                          for(Edge it: G[s]){
                                                              25
                                                                              if(d[it.t] > d[s] + it.w){
                                                              26
                                                                                  d[it.t] = d[s] + it.w;
   5
        Graph
                                                              27
                                                                                   if(inq[it.t])continue;
                                                              28
                                                                                  Q.push(it.t);
                                                              29
                                                                                  inq[it.t] = true;
   5.1 Articulation Point
                                                                                  if(++cnt[it.t] > v)return true;
                                                              30
                                                              31
                                                                              }
                                                              32
                                                                          }
 1 vector<int>v[N],bcc[N];//clear
                                                              33
 2 LL dep[N],low[N],bccno[N],time_cnt,bcc_cnt;//set dep
                                                              34
                                                                      return false;
       low -1 else 0
                                                              35|}
 3 bitset<N>is_AP;//0
  struct Edge{int s,t;};
  stack<Edge>st;//clear
                                                                        Bipartite
 6 int dfs(int s,int fa){
7
       int child=0;
8
       dep[s]=low[s]=time_cnt++;
                                                               1 #include <iostream>
9
       for(auto t:v[s]){
                                                                 #include <vector>
10
           Edge e=(Edge){s,t};
                                                               3
                                                                 #include <stack>
11
           if(dep[t]==-1){
                                                               4
                                                                 #include <cstring>
12
                st.push(e);
                                                               5
13
                child++:
                                                               6
                                                                 #define S 50050
14
                dfs(t,s);
                low[s]=min(low[s],low[t]);
15
                                                               8
                                                                 using namespace std;
16
                if(dep[s]<=low[t]){</pre>
17
                    is_AP[s]=1;
                                                              10 | vector<int> map[S];
18
                    bcc_cnt++;
                                                                 int visit[S];
                                                              11
19
                    bcc[bcc_cnt].clear();
                                                              12
                                                                 bool valid;
20
                    while(1){
                                                              13
21
                        Edge x=st.top(); st.pop();
                                                              14
                                                                 void check(int start) {
22
                        if(bccno[x.s]!=bcc_cnt){
                                                              15
                                                                      stack<int> st;
23
                             bcc[bcc_cnt].push_back(x.s);
                                                              16
                                                                      st.push(start);
24
                             bccno[x.s]=bcc_cnt;
                                                              17
                                                                      visit[start] = 1;
25
                                                              18
26
                        if(bccno[x.t]!=bcc_cnt){
                                                              19
                                                                      while(valid && !st.empty()) {
27
                            bcc[bcc_cnt].push_back(x.t);
                                                                          int cur = st.top();
                                                              20
28
                             bccno[x.t]=bcc_cnt;
                                                              21
                                                                          st.pop();
29
                                                              22
30
                        if(x.s==s&&x.t==t)break;
                                                              23
                                                                          for(int i = 0; i < map[cur].size(); i++) {</pre>
31
                    }
                                                              24
                                                                              int next = map[cur][i];
32
                                                              25
33
           }else if(low[s]>dep[t]){
                                                              26
                                                                              if(visit[next] == -1) {
34
                st.push(e);
                                                              27
                                                                                  st.push(next);
                low[s]=dep[t];
35
                                                              28
36
           }
                                                              29
                                                                                   if(visit[cur] == 1) visit[next] = 2;
37
                                                              30
                                                                                   else visit[next] = 1;
38
       if(fa<0&&child==1)is_AP[s]=0;</pre>
                                                              31
39
       return low[s];
                                                              32
                                                                              else if(visit[cur] == visit[next]) valid =
40 }
                                                              33
                                                                          }
                                                              34
                                                                     }
                                                              35
                                                                 }
   5.2 BellmanFord
                                                              36
                                                              37
                                                                 int main() {
1 struct Edge{
                                                              38
                                                                     int n, m;
       int t, w;
2
                                                              39
                                                                      cin >> n >> m;
3 \ };
                                                              40
 4 int v, e;
                                                              41
                                                                      for(int i = 0; i < m; i++) {</pre>
5 int d[N], cnt[N];
                                                              42
                                                                          int a, b;
6 bitset<N> inq;
                                                              43
                                                                          cin >> a >> b;
 7 queue < int > Q;
                                                              44
8
  vector<Edge>G[N];
                                                              45
                                                                          map[a].push_back(b);
                                                              46
                                                                          map[b].push_back(a);
10
   void addEdge(int from, int to, int w){
                                                              47
11
       G[from].push_back({to,w});
                                                              48
12|}
                                                              49
                                                                     // -1 : not visit, 1 : tsudere, 2 : proud
13
                                                              50
                                                                      memset(visit, -1, sizeof(visit));
14 bool hasnegativeCycle(){
                                                              51
                                                                      valid = true;
15
       while(!Q.empty())Q.pop();
                                                              52
16
       for(int i = 1; i <= v;i++){</pre>
                                                              53
                                                                      for(int i = 1; i <= n; i++) {</pre>
```

54

55

56

57

58

if(valid && visit[i] == -1) {

check(i);

}

}

17

18

19

20

21

inq[i] = true;

Q.push(i);

while(!Q.empty()){

cnt[i] = d[i] = 0;

```
if(valid) cout << "yes" << endl;</pre>
59
                                                             20
                                                                     while (m > 1 \&\& (p1[m - 1] - p1[m - 2]).cross(p[i])
       else cout << "no" << endl;</pre>
60
                                                                         - p1[m - 2]) <= 0)m--;
61
                                                              21
                                                                     p1[m++] = p[i];
62
       return 0;
                                                             22
                                                                   int k = m;
63 }
                                                             23
                                                                   for (int i = p.size() - 2; i >= 0; i--) {
                                                             24
                                                             25
                                                                     while (m > k \&\& (p1[m - 1] - p1[m - 2]).cross(p[i])
                                                                          - p1[m - 2]) <= 0)m--;
          dijkstra
                                                                     p1[m++] = p[i];
                                                             26
                                                             27
                                                             28
                                                                   if (n > 1)m--;
 1 struct Edge{
                                                             29
                                                                   p1.resize(m);
 2
       int from, to, w;
                                                             30 }
 3 };
 4 vector<Edge>E;
 5 vector<int>v[N];
                                                                 5.6 Dinic
 6 bitset<N> vis;
 7
  void init(){
       E.clear();
 8
                                                              1 struct dinic{
 9
       for(int i=0;i<N;i++){</pre>
                                                                   static const int M = 10000;
10
           v[i].clear();
                                                                   static const int INF = 1e9;
11
                                                                   struct Edge{
12|}
                                                               5
                                                                     int v;
13
                                                               6
                                                                     int f; //residual flow
14
   void addEdge(int from,int to,int w){
                                                               7
                                                                     int re:
       v[from].push_back(E.size());
15
                                                               8
16
       E.push_back(Edge{from,to,w});
                                                              9
                                                                   int n, s, t, level[M], now[M];
17|}
                                                              10
                                                                   vector<Edge> e[M];
18
                                                             11
                                                                   void init(int _n, int _s, int _t){
  void dijkstra(int s,int d[],int p[]){// set d[] INF &&
19
                                                                     n = _n; s = _s; t = _t;
       set p[] -1
                                                             13
                                                                     for (int i = 0; i <= n; i++)e[i].clear();</pre>
20
       d[s]=0;
                                                             14
21
       priority_queue<PII, vector<PII>, greater<PII>>pq;
                                                             15
                                                                   void add_edge(int u, int v, int f){
22
       vis.reset();
                                                                     e[u].push_back({ v, f, (int)e[v].size() });
                                                             16
       pq.push(MP(d[s],s));
23
                                                             17
                                                                     e[v].push_back({ u, f, (int)e[u].size() - 1 });
24
       while(!pq.empty()){
                                                             18
25
           PII k=pq.top(); pq.pop();
                                                             19
                                                                   bool bfs(){
26
           if(vis[k.second])continue;
                                                              20
                                                                     fill(level, level + n + 1, -1);
27
           vis[k.second]=true;
                                                             21
                                                                     queue<int> q;
           for(auto it:v[k.second]){
28
                                                                     q.push(s); level[s] = 0;
                                                              22
29
                Edge e=E[it];
                                                             23
                                                                     while (!q.empty()){
30
                if(d[e.to]>d[e.from]+e.w){
                                                             24
                                                                       int u = q.front(); q.pop();
31
                    d[e.to]=d[e.from]+e.w;
                                                             25
                                                                       for (auto it : e[u]){
32
                    p[e.to]=e.from;
                                                                         if (it.f > 0 && level[it.v] == -1){
                                                             26
33
                    pq.push(MP(d[e.to],e.to));
                                                              27
                                                                           level[it.v] = level[u] + 1;
34
               }
                                                              28
                                                                           q.push(it.v);
35
           }
                                                              29
36
       }
                                                             30
                                                                       }
37 }
                                                             31
                                                             32
                                                                     return level[t] != -1;
                                                             33
                                                              34
                                                                   int dfs(int u, int nf){
         Convex Hull
   5.5
                                                             35
                                                                     if (u == t)return nf;
                                                                     int res = 0;
                                                             36
 1 struct loc {
                                                              37
                                                                     while (now[u] < e[u].size()){</pre>
 2
     int x, y;
                                                             38
                                                                       Edge &it = e[u][now[u]];
 3
     loc() {};
                                                             39
                                                                       if (it.f>0 && level[it.v] == level[u] + 1){
                                                                         int tf = dfs(it.v, min(nf, it.f));
 4
     loc(int x, int y): x(x), y(y) {}
                                                             40
     bool operator <(const loc& b)const {return x != b.x ?41</pre>
                                                                         res += tf; nf -= tf; it.f -= tf;
          x < b.x : y < b.y;
                                                                         e[it.v][it.re].f += tf;
     bool operator ==(const loc& b)const {return x == b.x 43
                                                                         if (nf == 0)return res;
 6
         && y == b.y;
 7
     loc operator -(const loc& b)const {return loc(x - b.x45
                                                                       else now[u]++;
          , y - b.y);}
 8
     int cross(const loc& b)const {return x * b.y - y * b.47
                                                                     if (!res)level[u] = -1;
                                                                     return res;
 9
     int dis(loc a, loc b) {return (x - b.x) * (x - b.x) +49
          (y - b.y) * (y - b.y);
                                                                   int flow(int res = 0){
                                                             50
10|};
                                                                     while (bfs()){
11 vector<loc>p, p1;
                                                             52
                                                                       int temp;
12 int n;
                                                             53
                                                                       memset(now, 0, sizeof(now));
13 void convexhull() {
                                                              54
                                                                       while (temp = (dfs(s, INF))){
     sort(p.begin(), p.end());
                                                             55
                                                                         res += temp;
14
15
     p.erase(unique(p.begin(), p.end()), p.end());
                                                              56
16
     p1.clear();
                                                             57
17
     p1.resize(p.size());
                                                             58
                                                                     return res;
18
     int m = 0;
                                                             59
```

60 }d;

19

for (int i = 0; i < p.size(); i++) {</pre>

```
5.7
          FloydWarshall
                                                               30
                                                                    }
                                                                  }
                                                               31
                                                               32
 1 #include <iostream>
                                                               33
                                                                  void KM() {
2
                                                               34
                                                                    for (int i = 1; i <= n; i++) {
3
  #define INF 1e9
                                                               35
                                                                       Left[i] = Lx[i] = Ly[i] = 0;
4
  #define LL long long
                                                               36
                                                                       for (int j = 1; j <= n; j++) {</pre>
                                                               37
                                                                         Lx[i] = max(Lx[i], w[i][j]);
  using namespace std;
                                                               38
                                                               39
8
  int main() {
                                                               40
                                                                     for (int i = 1; i <= n; i++) {</pre>
9
       int n;
                                                                       while (true) {
                                                               41
10
                                                               42
                                                                         vx.reset(); vy.reset();
       while(cin >> n) {
11
                                                               43
                                                                         if (match(i))break;
12
           LL dis[n][n];
                                                               44
                                                                         update();
           LL ans = INF;
13
                                                               45
14
                                                               46
                                                                     }
            for(int i = 0; i < n; i++)</pre>
15
                                                               47
                                                                  }
16
                for(int j = 0; j < n; j++) {</pre>
                    cin >> dis[i][j];
17
18
                    if(dis[i][j] == 0) dis[i][j] = INF;
                }
                                                                          Longest Common Ancestor
19
20
21
           for(int i = 0; i < n; i++) {</pre>
22
                for(int j = 0; j < n; j++) {</pre>
                                                                1
                                                                  void preprocess() {
                                                                     for (int i = 1; i <= 25; i++) {
23
                    if(i == j) continue;
                                                                3
                                                                       for (int j = 1; j <= n; j++) {</pre>
24
                    ans = min(ans, dis[i][j] + dis[j][i]);
                                                                4
                                                                         if (par[j][i - 1] == -1 || par[par[j][i - 1]][i -
                    for(int k = 0; k < n; k++) {</pre>
25
                                                                              1] == -1)continue;
26
                         dis[i][j] = min(dis[i][j], dis[i][k]
                                                                         par[j][i] = par[par[j][i - 1]][i - 1];
                             ] + dis[k][j]);
                                                                6
27
28
                         ans = min(ans, dis[i][j] + dis[k][i
                             ] + dis[j][k]);
29
                    }
30
                }
31
           }
                                                                  5.10 MST
32
33
           if(ans == INF) cout << -1 << endl;</pre>
34
            else cout << ans << endl;</pre>
                                                                1 #include <iostream>
35
                                                                2 #include <vector>
36
                                                                3 #include <stack>
37
       return 0;
                                                                4 #include <cstring>
38 }
                                                                5
                                                                  #include <algorithm>
                                                                6
                                                                7
                                                                  #define LL long long
                                                                ۶l
                                                                  #define MAX 1e11
   5.8
          KM
                                                                  #define S 50050
                                                               10
                                                                  using namespace std;
1 int n;
                                                               11
 2 int Left[N];
                                                               12
                                                                  int n, m;
 3 double w[N][N], Lx[N], Ly[N];
                                                               13
                                                                  int sum;
 4 bitset<N> vx, vy;
                                                               14
                                                               15
                                                                  typedef struct {
6 bool match(int i) {
                                                                      int a, b, 1;
                                                               16
     vx[i] = true;
7
                                                               17
                                                                  } edge;
8
     for (int j = 1; j <= n; j++) {</pre>
                                                               18
                                                                  bool cmp(edge 1, edge r) { return 1.1 < r.1; }</pre>
9
       if ((fabs(Lx[i] + Ly[j] - w[i][j]) < 1e-9) && !vy[j19</pre>
                                                               20
                                                                  vector<edge> v;
10
         vy[j] = true;
                                                               21
11
         if (!Left[j] || match(Left[j])) {
                                                               22
                                                                  typedef struct {
12
           Left[j] = i;
                                                               23
                                                                       int d;
13
            return true;
                                                               24
                                                                       LL 1;
14
         }
                                                               25
                                                                  } node;
15
       }
                                                               26
16
                                                               27
                                                                  vector<node> map[S];
     return false;
17
                                                               28
18
                                                               29
                                                                  int disjoint[S];
19
                                                               30
20
   void update() {
                                                                   int root(int x) {
21
     double a = 1e30;
                                                               32
                                                                       if(disjoint[x] < 0) return x;</pre>
22
     for (int i = 1; i <= n; i++) {</pre>
                                                               33
                                                                       else {
       if (vx[i])for (int j = 1; j <= n; j++) {</pre>
23
                                                                           disjoint[x] = root(disjoint[x]);
24
           if (!vy[j])a = min(a, Lx[i] + Ly[j] - w[i][j]);35
                                                                           return disjoint[x];
25
                                                               36
                                                                       }
                                                               37
26
                                                                  }
27
     for (int i = 1; i <= n; i++) {
                                                               38
28
       if (vx[i])Lx[i] -= a;
                                                               39
                                                                  bool same(int a, int b) {
29
       if (vy[i])Ly[i] += a;
                                                               40
                                                                      return root(a) == root(b);
```

```
41 }
                                                               118
                                                                        sort(v.begin(), v.end(), cmp);
 42
                                                               119
 43
    void connect(int a, int b) {
                                                               120
                                                                        kruskal();
        // cout << "CONNECT" << a << " " << b << endl;
 44
                                                               121
                                                                        dfs(1);
 45
        int ra = root(a);
                                                               122
                                                               123
                                                                        if(!check()) cout << -1 << endl;</pre>
 46
        int rb = root(b);
 47
                                                               124
                                                                        else cout << sum << endl;</pre>
 48
        disjoint[ra] += disjoint[rb];
                                                               125
 49
        disjoint[rb] = ra;
                                                               126
                                                                        return 0;
 50|}
                                                               127 }
 51
 52
    void kruskal() {
 53
        int remain = n - 1;
                                                                    5.11 SPFA
 54
        for(auto i : v) {
 55
            if(remain == 0) break;
                                                                 1 #include <iostream>
 56
                                                                 2
                                                                   #include <vector>
 57
            if(!same(i.a, i.b)) {
                                                                 3
                                                                    #include <stack>
 58
                 connect(i.a, i.b);
 59
                                                                   #include <queue>
                                                                   #include <cstring>
 60
                 map[i.a].push_back((node){i.b, i.l});
 61
                 map[i.b].push_back((node){i.a, i.l});
                                                                 7
                                                                    #define S 50050
 62
                                                                 8
                                                                    #define MAX 1e11
 63
                 sum += i.l;
                                                                 9
                                                                   #define LL long long
                 remain--;
 64
                                                                10
 65
            }
                                                                11
                                                                   using namespace std;
 66
        }
                                                                12
 67
   }
                                                                13
                                                                    typedef struct {
 68
                                                                        int d;
 69
   bool book[S];
                                                                14
                                                                15
                                                                        LL 1;
 70
                                                                16
                                                                   } XXX;
 71
   void dfs(int start) {
                                                                    vector<XXX> map[S];
                                                                17
 72
        stack<int> st;
                                                                18
 73
        st.push(start);
                                                                19
 74
                                                                20 LL lon[S];
 75
 76
        memset(book, false, sizeof(book));
                                                                21 int cnt[S];
                                                                22
                                                                   int n, m;
 77
                                                                23
                                                                    bool cycle;
 78
        while(!st.empty()) {
                                                                24
                                                                   bool inqueue[S];
 79
            int cur = st.top();
                                                                25
            // cout << cur << endl;
 80
                                                                26
                                                                    void dfs(int start) {
 81
            st.pop();
                                                                27
                                                                        stack<int> st;
 82
 83
                                                                28
                                                                        st.push(start);
            book[cur] = true;
                                                                29
 84
                                                                        bool book[S];
                                                                30
 85
             for(int i = 0; i < map[cur].size(); i++) {</pre>
                                                                31
                                                                        memset(book, false, sizeof(book));
 86
                 int next = map[cur][i].d;
                                                                32
 87
                 if(!book[next]) {
                                                                33
                                                                        while(!st.empty()) {
 88
                     st.push(next);
                                                                34
                                                                            int cur = st.top();
 89
                                                                35
                                                                             // cout << cur << endl;</pre>
 90
            }
                                                                36
                                                                            st.pop();
 91
                                                                37
                                                                            lon[cur] = -MAX;
 92|}
                                                                38
                                                                            book[cur] = true;
 93
                                                                39
 94
    void init() {
                                                                40
                                                                            for(int i = 0; i < map[cur].size(); i++) {</pre>
 95
        memset(disjoint, -1, sizeof(disjoint));
                                                                41
                                                                                 int next = map[cur][i].d;
 96
        sum = 0;
                                                                42
                                                                                 if(!book[next]) st.push(next);
 97
   }
                                                                43
                                                                            }
 98
                                                                44
                                                                        }
 99
    bool check() {
                                                                45
100
        for(int i = 1; i <= n; i++)</pre>
                                                                46
101
            if(!book[i]) return false;
                                                                47
                                                                    void spfa(int start) {
102
                                                                        memset(inqueue, false, sizeof(inqueue));
                                                                48
103
        return true;
                                                                49
                                                                        for(int i = 0; i < S; i++) lon[i] = MAX;</pre>
104 }
                                                                50
                                                                        cycle = false;
105
                                                                51
106
   int main() {
107
                                                                52
                                                                        queue<int> q;
        init();
                                                                53
108
                                                                        q.push(start);
                                                                54
                                                                        lon[start] = 0;
109
        cin >> n >> m;
                                                                55
                                                                        inqueue[start] = true;
110
                                                                56
111
        for(int i = 0; i < m; i++) {</pre>
                                                                57
                                                                        while(!q.empty()) {
112
            edge tmp;
                                                                            int cur = q.front();
                                                                58
113
            cin >> tmp.a >> tmp.b >> tmp.l;
                                                                59
                                                                             q.pop();
114
                                                                60
                                                                             inqueue[cur] = false;
115
            v.push_back(tmp);
                                                                             // cout << "AT: " << cur << " " << cnt[cur] <<
                                                                61
116
        }
                                                                                 endl:
117
                                                                62
                                                                             cnt[cur]++;
```

```
63
            if(cnt[cur] > n) {
                                                                33
                                                                            {
64
                                                                34
                                                                                dp[p[s][i]] = dp[s]+n-num[p[s][i]]*2;
                 dfs(cur);
 65
                 return ;
                                                                35
                                                                                solve(p[s][i], n);
                                                                36
66
            }
                                                                            }
67
                                                                37
                                                                        }
68
            for(int i = 0; i < map[cur].size(); i++) {</pre>
                                                                38
69
                 int next = map[cur][i].d;
                                                                39
70
                                                                40
                                                                   int main()
71
                 if(lon[next] > lon[cur] + map[cur][i].1) {
                                                                41
 72
                     lon[next] = lon[cur] + map[cur][i].1;
                                                                42
                                                                        int n;
                                                                        scanf("%d", &n);
73
                     if(!inqueue[next] && cnt[cur] <= n) {</pre>
                                                                43
74
                          q.push(next);
                                                                44
                                                                        for(int i=1; i<n; i++)</pre>
75
                          inqueue[next] = true;
                                                                45
76
                                                                46
                     }
                                                                            int a, b;
77
                 }
                                                                47
                                                                            scanf("%d%d", &a, &b);
78
            }
                                                                48
                                                                            p[a].push_back(b);
 79
        }
                                                                49
                                                                            p[b].push_back(a);
80|}
                                                                50
81
                                                                51
                                                                       dfs(1, 0);
                                                                       memset(f, 0, sizeof(f));
82
                                                                52
                                                                53
83
   int main() {
                                                                        solve(1, n);
        cin >> n >> m;
                                                                54
                                                                        for(int i=1; i<=n; i++)</pre>
85
                                                                55
                                                                            printf("%d \setminus n", dp[i]);
86
        for(int i = 0; i < m; i++) {</pre>
                                                                56
                                                                        return 0;
                                                                57 }
87
            int a, b;
            LL c;
88
 89
            cin >> a >> b >> c;
90
                                                                   5.13 TopologicalSort
91
            map[a].push_back((XXX) {b, c});
92
                                                                 1 #include <iostream>
93
                                                                 2
                                                                   #include <stack>
94
        spfa(1);
                                                                 3
                                                                   #include <vector>
95
                                                                 4
                                                                   #include <cstring>
96
        if(lon[n] >= MAX || lon[n] <= -MAX) cout << "QAQ"</pre>
             << endl;
                                                                 6
                                                                   #define S 50050
 97
        else cout << lon[n] << endl;</pre>
                                                                 7
98
                                                                 8
                                                                   using namespace std;
99
        return 0;
100 }
                                                                10 | vector<int> map[S];
                                                                11 stack<int> ans;
                                                                12
                                                                   int state[S];
   5.12 SumOfDistanceInTree
                                                                13
                                                                   bool head[S];
                                                                   bool valid;
                                                                14
                                                                15
                                                                   int n, m;
 1 #include <bits/stdc++.h>
                                                                16
 2 #pragma comment(linker, "/STACK:10240000,10240000")//递17
                                                                   void dfs(int cur) {
        归太深,导致爆栈,所以使用扩栈语句
                                                                18
                                                                        state[cur] = 1;
 3 using namespace std;
                                                                19
                                                                20
                                                                        for(auto next : map[cur])
 5 | const int N = 100009;
                                                                21
                                                                            if(!state[next]) dfs(next);
   int dp[N] = {}, num[N];
                                                                22
                                                                            else if(state[next] == 1) {
 7 vector<int> p[N];
                                                                23
                                                                                valid = false;
 8 | bool f[N] = {};
                                                                24
                                                                                return ;
 9
                                                                25
                                                                            }
 10 void dfs(int s, int depth)
                                                                26
11 | {
                                                                27
                                                                        state[cur] = 2;
        int len = p[s].size();
                                                                28
12
                                                                        ans.push(cur);
13
        f[s] = 1;
                                                                29
                                                                30
14
        num[s] = 1;
15
        dp[1] += depth;
                                                                31
16
        for(int i=0; i<len; i++)</pre>
                                                                32
                                                                   void topology_sort() {
                                                                33
                                                                       for(int i = 1; i <= n; i++)</pre>
17
18
            if(!f[p[s][i]])
                                                                34
                                                                            if(valid && head[i]) dfs(i);
19
                                                                35
                 dfs(p[s][i], depth+1);
                                                                        if(!valid) {
 20
                                                                36
 21
                 num[s] += num[p[s][i]];
                                                                37
                                                                            cout << -1 << endl;
22
                                                                38
            }
                                                                            return ;
 23
        }
                                                                39
24 }
                                                                40
25
                                                                41
                                                                       while(!ans.empty()) {
 26 void solve(int s, int n)
                                                                42
                                                                            cout << ans.top() << endl;</pre>
27 | {
                                                                43
                                                                            ans.pop();
 28
        int len = p[s].size();
                                                                44
                                                                        }
                                                                45
 29
        f[s] = 1;
                                                                   }
```

46 47

48

int main() {

cin >> n >> m;

30

31

32

for(int i=0; i<len; i++)</pre>

if(!f[p[s][i]])

```
49
50
       memset(head, true, sizeof(head));
51
52
       for(int i = 0; i < m; i++) {</pre>
53
            int a, b;
54
            cin >> a >> b;
55
56
            head[b] = false;
57
58
            map[a].push_back(b);
59
60
61
       memset(state, 0, sizeof(state));
62
       valid = true;
63
64
       topology_sort();
65
66
       return 0;
67 }
```

6 Number

6.1 Catalan

```
C_0 = 1 \quad \text{and} \quad C_{n+1} = \tfrac{2(2n+1)}{n+2} C_n,
```

6.2 Combination

```
1 #include <bits/stdc++.h>
 2 using namespace std;
3 typedef long long LL;
4 const int M=1000005;
5 int n,k;
6 LL m,phi;
 7 vector <int> facs;
8 LL dp[M],dp2[M][32];
10 LL pw(LL x,LL y){
11  // cout<<x<<' '<<y<<'\n';
12
       LL ret=1,tmp=x%m;
13
       while(y){
            if(y&1)ret=ret*tmp%m;
14
15
            tmp=tmp*tmp%m;
16
            y>>=1;
17
18
       return ret;
19|}
20
21
   void init(){
22
       facs.clear();
23
       LL x=m, sq=(LL)sqrt(m);
24
       phi=1;
       for(LL i=2;i<=sq;i++){
25
26
            if(x%i)continue;
27
            phi*=i-1; x/=i;
            facs.push_back(i);
28
29
            while(x\%i==0){
                phi*=i;
30
31
                x/=i;
            }
32
33
34
       if(x>1){
            phi*=x-1;
35
36
            facs.push_back((int)x);
37
38
       k=facs.size();
39
       dp[0]=1;
40
       memset(dp2,0,sizeof(dp2));
41
       for(int i=1;i<M;i++){</pre>
42
            LL tmp=i;
43
            for(int j=0;j<k;j++){</pre>
44
                dp2[i][j]=dp2[i-1][j];
45
                while(tmp%facs[j]==0){
```

```
46
                     tmp/=facs[j];
47
                     dp2[i][j]++;
48
                }
49
            dp[i]=dp[i-1]*tmp%m;
50
51
        }
52
       return;
53
   }
54
55
   int main(){
56
        while(cin>>n>>m){
57
            init();
58
            while(n--){
59
                LL ans=1;
60
                int x,y;
61
                cin>>x>>y;
62
                 for(int i=0;i<k;i++){</pre>
                     ans=ans*pw(facs[i],dp2[x][i]-dp2[x-y][i
63
        ]-dp2[y][i])%m;
64
                ans=ans*dp[x]%m;
65
                ans=ans*pw(dp[y],phi-1)%m;
66
                ans=ans*pw(dp[x-y],phi-1)%m;
67
68
                cout << ans << '\n';</pre>
69
            }
70
        }
71 }
```

6.3 Extend Euclidean.cpp

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

6.4 GaussElimination

```
1 \mid const int MAXN = 300;
   const double EPS = 1e-8;
   int n;
   double A[MAXN][MAXN];
 5
   void Gauss() {
     for(int i = 0; i < n; i++) {</pre>
 7
       bool ok = 0;
        for(int j = i; j < n; j++) {</pre>
 8
 9
          if(fabs(A[j][i]) > EPS) {
10
            swap(A[j], A[i]);
11
            ok = 1;
12
            break;
13
          }
14
15
        if(!ok) continue;
16
        double fs = A[i][i];
17
        for(int j = i+1; j < n; j++) {</pre>
18
          double r = A[j][i] / fs;
19
          for(int k = i; k < n; k++) {</pre>
20
            A[j][k] -= A[i][k] * r;
21
22
       }
23
24 }
```

6.5 Matrix

```
1 template < typename T, int N=2>
2 struct Mat {//Matrix
3 unsigned long long v[N][N];
4 Mat operator*(Mat b)const {
5 Mat val;
```

```
6
       for (int i = 0; i < N; i++) {
                                                                 24 }
7
          for (int j = 0; j < N; j++) {
                                                                 25
                                                                    int main(){
8
            val.v[i][j] = 0;
                                                                 26
                                                                       int t, i;
9
            for (int k = 0; k < N; k++) {
                                                                 27
                                                                       string s;
              val.v[i][j] += v[i][k] * b.v[k][j];
10
                                                                 28
                                                                       for (i = 0, cin >> t; i < t; i++){}
11
                                                                 29
                                                                         cin >> s;
12
         }
                                                                 30
                                                                         int fail[N];
13
       }
                                                                 31
                                                                         bulid_fail_funtion(s, fail);
14
       return val;
                                                                 32
                                                                         int p = s.length() - 1;
15
                                                                         if (fail[p] != -1 && (p + 1) % (p - fail[p]) == 0)
                                                                 33
                                                                         printf("%d\n", p - fail[p]);
else printf("%d\n", p + 1);
16 };
                                                                 35
                                                                 36 }
```

6.6 Phi

```
1 void phi_table(int n){
2
       phi[1] = 1;
 3
       for(int i = 2; i <= n; i++){</pre>
4
            if(phi[i])continue;
5
            for(int j = i; j < n; j += i){</pre>
6
                if(!phi[j])phi[j] = j;
                phi[j] = phi[j] / i * (i - 1);
7
8
9
       }
10|}
```

6.7 Prime table

```
1 void PrimeTable(){
2
       is_notp.reset();
3
     is_notp[0] = is_notp[1] = 1;
4
     for (int i = 2; i < N; i++){</pre>
5
       if (is_notp[i])continue;
6
       p.push_back(i);
7
       for (int j=0;i*p[j]<N&&j<p.size();j++){</pre>
         is_notp[i*p[j]] = 1;
8
9
          if(i%p[j]==0)break;
10
11
     }
12|}
```

String

7.1 KMP

23

}

```
2
1 void bulid_fail_funtion(string B, int *fail){
 2
     int len = B.length(), current_pos;
3
     current_pos = fail[0] = -1;
                                                               5
4
     for (int i = 1; i<len; i++){</pre>
5
       while (current_pos != -1 && B[current_pos + 1] != B
            [i]){
6
         current_pos = fail[current_pos];
                                                               9
7
                                                              10
       if (B[current_pos + 1] == B[i])current_pos++;
8
                                                              11
9
       fail[i] = current_pos;
10
     }
                                                              12
11 }
                                                              13
  void match(string A, string B, int *fail){
12
                                                              14
13
     int lenA = A.length(), lenB = B.length();
                                                              15
14
     int current_pos = -1;
                                                              16
     for (int i = 0; i<lenA; i++){</pre>
15
       while (current_pos != -1 && B[current_pos + 1] != A<sup>17</sup>|}
16
            [i]){
17
         current_pos = fail[current_pos];
18
19
       if (B[current_pos + 1] == A[i])current_pos++;
20
       if (current_pos == lenB - 1){//match! A[i-lenB+1,i
            ]=B
21
         current pos = fail[current pos];
22
```

7.2 Trie

```
1 //init sz=1 trie[0]=0
 2
   void insert(string s){
 3
       int u=0,v;
       for(int i=0;i<r.size();i++){</pre>
 5
            v=r[i]-'a';
 6
            if(!trie[u][v]){
 7
                memset(trie[sz],0,sizeof(trie[sz]));
 8
                val[sz]=0;
 9
                trie[u][v]=sz++;
10
11
            u=trie[u][v];
12
       val[u]=1;
13
14
       return;
15
   void search(string s,int i){
16
       int u=0,v;
17
18
       dp[i]=0;
19
       for(int j=i;j<s.size();j++){</pre>
20
            v=s[j]-'a';
21
            if(!trie[u][v])return;
22
            u=trie[u][v];
23
            if(val[u])dp[i]=(dp[i]+dp[j+1])%MOD;
24
       }
25
       return;
26 }
```

7.3 Zvalue

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