

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int n,m;
5  map<string,int> mapa;
6  vector<int> parent,r;
7  vector<int> filhos;
8  int findSet(int i) {
9      return (parent[i] == -1) ? i : parent[i] = findSet(parent[i]);
10 }
11 bool isSameSet(int i, int j) {
12     return findSet(i) == findSet(j);
13 }
14 void unionF(int i,int j){
15     if (!isSameSet(i, j)) {
16         int x = findSet(i), y = findSet(j);
17         if (r[x] > r[y]) {
18             filhos[findSet(x)] += filhos[findSet(y)];
19             parent[y] = x;
20         } else {
21             filhos[findSet(y)] += filhos[findSet(x)];
22             parent[x] = y;
23             if (r[x] == r[y])
24                 r[y]++;
25         }
26     }
27 }
28 main(){
29     int i,j,k;
30     string a,b;
31
32     cin >> n;
33
34     for(i=0;i<n;i++){
35         cin >> m;
36         for(j=0;j<m;j++){
37             cin >> a >> b;
38             if(!mapa.count(a)){
39                 int aux = mapa.size();
40                 mapa[a] = aux;
41                 parent.push back(-1);
42                 filhos.push back(1);
43                 r.push back(0);
44             }
45             if(!mapa.count(b)){
46                 int aux = mapa.size();
47                 mapa[b] = aux;
48                 parent.push back(-1);
49                 filhos.push back(1);
50                 r.push back(0);
51             }
52             unionF(mapa[a],mapa[b]);
53             cout << filhos[findSet(mapa[a])] << endl;
54         }
55         mapa.clear();
56         parent.clear();
57         filhos.clear();
58         r.clear();
59     }
60 }
61
```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int n;
5  int A[21];
6  int M[21][21];
7
8  int dp[21][1<<21];
9
10 int solve(int current, int w,int visi){
11     visi |= (1 << current);
12     if(w<=0)
13         return 0;
14     int ans = 0;
15     if(dp[current][visi]!=-1) return dp[current][visi];
16     for(int i=0;i<n;i++){
17         if(!(visi & (1 << i)) and w-(A[i]+M[current][i]) >= 0){
18             ans = max(ans,solve(i,w-(A[i]+M[current][i]),visi)+1);
19         }
20     }
21     return dp[current][visi] = ans;
22 }
23
24
25 main(){
26     ios base::sync with stdio(0);
27     cin.tie(0);
28     while(cin >> n and n){
29         memset(dp,-1,sizeof dp);
30         for(int i=0;i<n;i++){
31             cin >> A[i];
32         }
33         for(int i=0;i<n;i++){
34             for(int j=0;j<n;j++){
35                 cin >> M[i][j];
36             }
37         }
38         int ans = 0;
39         for(int i=0;i<n;i++){
40             int visi = (1 << i);
41             if(420 - A[i] >=0)
42                 ans = max(ans,solve(i,420 - A[i],visi)+1);
43         }
44
45         cout << ans << endl;
46     }
47 }
48
49 }
50

```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  double V[16][16];
5  vector<pair<int,int> > P;
6  int n;
7
8  double dp[20][1 << 16];
9
10 inline double calc(int i,int j){
11     return sqrt((P[i].first-P[j].first)*(P[i].first-P[j].first) + (P[i].second -
12     P[j].second)*(P[i].second - P[j].second));
13 }
14 double solve(int current, int mask){
15     if(mask == ((1<<(n+1))-1)){
16         return V[current][0];
17     }
18     if(dp[current][mask]!=-1) return dp[current][mask];
19     double ans = 1e9 + 10;
20     for(int i=1;i<=n;i++){
21         if(!(mask & (1<<i)))
22             ans = min(solve(i,mask | (1<<i))+V[current][i],ans);
23     }
24     return dp[current][mask] = ans;
25 }
26
27 main(){
28     ios base::sync with stdio(0);
29     cin.tie(0);
30     int x,y,a,b;
31     while(cin >> n and n){
32         P.clear();
33         for(int i=0;i<=n;i++){
34             for(int j=0;j<=(1<<(n+1));j++){
35                 dp[i][j] = -1;
36             }
37         }
38         cin >> x >> y;
39         P.push back(make pair(x,y));
40         for(int i=1;i<=n;i++){
41             cin >> a >> b;
42             P.push back(make pair(a,b));
43         }
44         for(int i=0;i<=n;i++){
45             for(int j=0;j<=n;j++){
46                 V[i][j] = calc(i,j);
47             }
48         }
49         cout << fixed << setprecision(2) << solve(0,1) << endl;
50     }
51 }
52
53
54
55
56

```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int n;
5  int M[19][19];
6  int A[19];
7  int dp[19][(1 << 19)];
8  int dead;
9  int solve(int current,int mask){
10
11     if(mask == dead)
12         return 0;
13
14     if(dp[current][mask]!=-1) return dp[current][mask];
15
16     int ans = 1e9+10;
17
18     for(int i=0;i<n;i++){
19         if(!(mask & A[i])){
20             ans = min(ans,solve(current+1,(mask | A[i]))+M[i][current]);
21         }
22     }
23     return dp[current][mask] = ans;
24 }
25
26 main(){
27     for(int i=0;i<=18;i++)
28         A[i] = 1 << i;
29     while(scanf("%d",&n) and n){
30         for(int i=0;i<n;i++)
31             for(int j=0;j<(1 << (n+1));j++)
32                 dp[i][j] = -1;
33         dead = (1 << n)-1;
34         for(int i=0;i<n;i++){
35             for(int j=0;j<n;j++){
36                 scanf("%d",&M[i][j]);
37             }
38         }
39         cout << solve(0,0) << endl;
40     }
41 }
42

```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef long long int ll;
5
6  main(){
7      ios base::sync with stdio(0);
8      cin.tie(0);
9      ll z,n;
10     cin >> z;
11     for(int k=0;k<z;k++){
12         cin >> n;
13         ll nr = n << 1;
14         ll M[nr][nr];
15         for(int i=0;i<n;i++){
16             for(int j=0;j<n;j++){
17                 cin >> M[i][j];
18                 M[i+n][j] = M[i][j];
19                 M[i][j+n] = M[i][j];
20                 M[i+n][j+n] = M[i][j];
21                 if(i>0) M[i][j] += M[i-1][j];
22                 if(j>0) M[i][j] += M[i][j-1];
23                 if(i>0 && j>0) M[i][j] -= M[i-1][j-1];
24             }
25         }
26         for(int i=0;i<nr;i++){
27             for(int j=0;j<nr;j++){
28                 if((i>=n or j>=n)){
29                     if(i>0){
30                         M[i][j] += M[i-1][j];
31                     }
32                     if(j>0){
33                         M[i][j] += M[i][j-1];
34                     }
35                     if(i>0 && j>0){
36                         M[i][j] -= M[i-1][j-1];
37                     }
38                 }
39             }
40         }
41         ll ans = -1000*100*100;
42         int xi,yi,xf,yf;
43         for(int i=0;i<n;i++){
44             for(int j=0;j<n;j++){
45                 for(int x=i;x<i+n;x++){
46                     for(int y=j;y<j+n;y++){
47                         ll at = M[x][y];
48                         if(i>0) at -= M[i-1][y];
49                         if(j>0) at -= M[x][j-1];
50                         if(i>0 and j>0) at += M[i-1][j-1];
51                         ans = max(ans,at);
52                         if(ans == at){
53                             xi = i;
54                             yi = j;
55                             xf = x;
56                             yf = y;
57                         }
58                     }
59                 }
60             }
61         }
62         if(xi==0 and yi==0 and ((xf==nr-1 and yf==n-1) or (xf==n-1 and yf==nr-1)))
63             ans -= M[n-1][n-1];
64         cout << ans << endl;
65         //cout << "COORD I:  " << xi << " " << yi << endl;
66         // << "COORD F:  " << xf << " " << yf << endl;
67     }
68 }
69
```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  #define F first
4  #define S second
5  #define mp make pair
6  #define pb push back
7
8  typedef long long int ll;
9
10 ll sum[20005];
11 ll n,s;
12
13 main(){
14     ios base::sync with stdio(0);
15     cin.tie(0);
16     cin >> n;
17
18     for(int k=0;k<n;k++){
19         cin >> s;
20         ll ans = 0,ansx=1,ansy=1;
21         ll at = 0;
22         int from=1,to=1;
23         for(int i=0;i<s-1;i++){
24             cin >> sum[i];
25             at += sum[i];
26             if(at >= ans){
27                 if(at > ans or (ansy-ansx < i+2-from) or (ansy-ansx == i+2-from
28                     and from <= ansx)){
29                     ansx = from;
30                     ansy = i+2;
31                 }
32                 ans = at;
33             }
34             if(at<0){
35                 at = 0;
36                 from = i+2;
37             }
38         }
39         if(ans > 0)
40             cout << "The nicest part of route " << k+1 << " is between stops " <<
41                 ansx << " and " << ansy << endl;
42         else
43             cout << "Route " << k+1 << " has no nice parts" << endl;
44     }
```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  #define mp make pair
5  #define F first
6  #define S second
7  typedef long long int ll;
8  typedef pair<ll,ll> ii;
9  typedef vector<ll> vi;
10 typedef vector<ii> vii;
11
12 vi A;
13 vii st;
14 int n,m;
15
16 int getMax(vector<ll> e){
17     ll s = -1;
18     int p=-1;
19     for(int i=0;i<e.size();i++){
20         if(e[i]!=-1 and s < A[e[i]]){
21             s = A[e[i]];
22             p = i;
23         }
24     }
25     return p;
26 }
27
28 void build(int P,int L,int R){
29     if(L==R){
30         st[P] = mp(L,-1);
31         return;
32     }
33     if(L>R || R<L)
34         return;
35     int nxt = P << 1;
36     int mid = (L+R) >> 1;
37     build(nxt,L,mid);
38     build(nxt+1,mid+1,R);
39
40     vi e;
41     ll s1,s2;
42     int p;
43     e.push_back(st[nxt].F);
44     e.push_back(st[nxt].S);
45     e.push_back(st[nxt+1].F);
46     e.push_back(st[nxt+1].S);
47     p = getMax(e);
48     s1 = p==-1 ? -1:e[p];
49     e[p]=-1;
50     p = getMax(e);
51     s2 = p==-1 ? -1:e[p];
52
53     st[P] = mp(s1,s2);
54 }
55
56
57 void update (int p, int L, int R, int i, int value) {
58
59     // no overlap
60     if(L > i or R < i) return;
61
62     // total overlap
63     if(L == R and L == i) {
64         A[i] = value;
65         st[p] = mp(i,-1);
66         return;
67     }
68
69     int nxt = p << 1;
70     int mid = (L + R) >> 1;

```

```

71     update (nxt, L, mid, i, value);
72     update (nxt + 1, mid + 1, R, i, value);
73
74     vi e;
75     ll s1,s2;
76     int V;
77     e.push_back(st[nxt].F);
78     e.push_back(st[nxt].S);
79     e.push_back(st[nxt+1].F);
80     e.push_back(st[nxt+1].S);
81     V = getMax(e);
82     s1 = V==-1? -1:e[V];
83     e[V]=-1;
84     V = getMax(e);
85     s2 = V==-1? -1:e[V];
86     st[p].F = s1;
87     st[p].S = s2;
88 }
89 ii query(int p, int L, int R, int i, int j){
90     // no overlap
91     if(i>R || j<L) return mp(-1,-1);
92
93     // total overlap
94     if(L>=i && R<=j) return st[p];
95
96     // partial overlap
97     int nxt = p << 1;
98     int mid = (L + R) >> 1;
99     ii p1 = query(nxt,L,mid,i,j);
100    ii p2 = query(nxt + 1,mid +1,R,i,j);
101
102    if(p1.F== -1 and p1.S== -1) return p2;
103    if(p2.F== -1 and p2.S== -1) return p1;
104
105    vi e;
106    ll s1,s2;
107    int V;
108    e.push_back(p1.F);
109    e.push_back(p1.S);
110    e.push_back(p2.F);
111    e.push_back(p2.S);
112    V = getMax(e);
113    s1 = V==-1? -1:e[V];
114    e[V] = -1;
115    V = getMax(e);
116    s2 = e[V];
117
118    return mp(s1,s2);
119 }
120
121
122 main(){
123     int i,j,k,a,b;
124     char o;
125     cin >> n;
126     st.resize(4*n);
127     A.resize(2*n);
128     st.assign(4*n,mp(-1,-1));
129     A.assign(2*n,-1);
130     for(i=0;i<n;i++){
131         cin >> A[i];
132     }
133     build(1,0,n-1);
134     cin >> m;
135     for(i=0;i<m;i++){
136         cin >> o >> a >> b;
137         if(o=='Q'){
138             ii aux = query(1,0,n-1,a-1,b-1);
139             cout << A[aux.F]+A[aux.S] << endl;
140         }

```



```
141         else{
142             update(1,0,n-1,a-1,b);
143         }
144     }
145 }
146 }
147
148
149
150
151
```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef long long int ll;
5  typedef pair<ll,ll> ii;
6  typedef vector<ii> vii;
7  typedef vector<ll> vi;
8
9  vi st,lazy;
10 int n;
11
12 ll query(int p, int L, int R, int i,int j){
13
14     if(lazy[p]!=0){
15         st[p] += (R-L+1)*lazy[p];
16         if(R!=L){
17             lazy[p<<1] += lazy[p];
18             lazy[(p<<1)+1] += lazy[p];
19         }
20         lazy[p] = 0;
21     }
22     // no overlap
23     if(i>R || j<L) return 0;
24
25     // total overlap
26     if(L>=i && R<=j) return st[p];
27
28     // partial overlap
29     int nxt = p << 1;
30     int mid = (L + R) >> 1;
31
32     return query(nxt,L,mid,i,j) + query(nxt + 1,mid +1,R,i,j);
33 }
34 void update(int P,int L,int R, int i,int j, ll value){
35
36     if(lazy[P]!=0){
37         st[P] += (R-L+1)*lazy[P];
38         if(L!=R){
39             lazy[P << 1] += lazy[P];
40             lazy[(P << 1)+1] += lazy[P];
41         }
42         lazy[P] = 0;
43     }
44
45     // no overlap
46     if( L > j or R < i) return;
47
48     // total overlap
49     if(L >= i and R <= j){
50         st[P] += (R-L+1)*value;
51         if(L!=R){
52             lazy[P<<1] += value;
53             lazy[(P<<1)+1] += value;
54         }
55
56         return;
57     }
58
59     // partial overlap
60     int nxt = P << 1;
61     int mid = (L+R) >> 1;
62
63     update(nxt, L, mid, i, j, value);
64     update(nxt+1,mid+1,R,i,j,value);
65
66     st[P] = st[nxt]+st[nxt+1];
67
68 }
69
70 main(){

```

```
71     int i,j,q,z,a,b,o;
72     ll v;
73
74     cin >> z;
75
76     for(i=0;i<z;i++){
77         cin >> n >> q;
78         st.resize(n << 2);
79         st.assign(n << 2,0);
80         lazy.resize(n << 2);
81         lazy.assign(n << 2,0);
82         for(j=0;j<q;j++){
83             cin >> o;
84             if(o==1){
85                 cin >> a >> b;
86                 cout << query(1,0,n-1,a-1,b-1) << endl;
87             }
88             else{
89                 cin >> a >> b >> v;
90                 update(1,0,n-1,a-1,b-1,v);
91             }
92         }
93     }
94 }
95
```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  vector<vector<string > > Grafo(105);
5  map<string,int> V;
6  map<int,string> S;
7  map<string,int> grau;
8  int n,m;
9  vector<int> saida;
10 void kahn(){
11     int i;
12     priority queue<int> F;
13     for(i=0;i<n;i++){
14         if(grau[S[i]]==0)
15             F.push(-i);
16     }
17     while(!F.empty()){
18         int aux = -F.top();
19         saida.push back(aux);
20         F.pop();
21         for(i=0;i<Grafo[aux].size();i++){
22             if(--grau[Grafo[aux][i]]==0)
23                 F.push(-V[Grafo[aux][i]]);
24         }
25     }
26 }
27
28 main(){
29     int i,j,k,cont=1;
30     string e,from,to;
31     while(cin >> n){
32         cin.ignore();
33         for(i=0;i<n;i++){
34             cin >> e;
35             cin.ignore();
36             V[e]=i;
37             S[i]=e;
38             Grafo[i].clear();
39         }
40         cin >> m;
41         for(i=0;i<m;i++){
42             cin >> from >> to;
43             Grafo[V[from]].push back(to);
44             grau[to]++;
45         }
46         kahn();
47         cout << "Case #" << cont << ": Dilbert should drink beverages in this
order: ";
48         for(i=0;i<saida.size();i++){
49             cout << S[saida[i]];
50             if(i!=saida.size()-1)
51                 cout << " ";
52         }
53         cout << "." << endl;
54         V.clear();
55         S.clear();
56         saida.clear();
57         cont++;
58         cout << endl;
59     }
60 }
61

```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef vector<vector<int> > vvi;
4  typedef vector<int> vi;
5  typedef vector<pair<int,int> > vii;
6  typedef pair<int,int> ii;
7  vvi Grafo(100001);
8  bool visitados[100001];
9  vector<bool> visi;
10 stack<int> ordem;
11 int n,m;
12 void dfs(int n){
13     visitados[n] = true;
14     for(int i=0;i<Grafo[n].size();i++){
15         if(!visitados[Grafo[n][i]])
16             dfs(Grafo[n][i]);
17     }
18 }
19 void dfsOrd(int n){
20     visitados[n] = true;
21     for(int i=0;i<Grafo[n].size();i++){
22         if(!visitados[Grafo[n][i]])
23             dfs(Grafo[n][i]);
24     }
25     ordem.push(n);
26 }
27 void reset(){
28     for(int i=0;i<n;i++)
29         visitados[i]=false;
30 }
31
32 main(){
33     ios base::sync with stdio(0);
34     cin.tie(0);
35     int i,j,z,from,to;
36     cin >> z;
37
38     for(i=0;i<z;i++){
39         cin >> n >> m;
40         for(j=0;j<n;j++)
41             Grafo[j].clear();
42         for(j=0;j<m;j++){
43             cin >> from >> to;
44             Grafo[from-1].push back(to-1);
45         }
46         reset();
47
48         for(j=0;j<n;j++){
49             if(!visitados[j])
50                 dfsOrd(j);
51         }
52         reset();
53         int cont = 0;
54         while(!ordem.empty()){
55             int x = ordem.top();
56             ordem.pop();
57             if(!visitados[x]){
58                 dfs(x);
59                 cont++;
60             }
61         }
62         cout << cont << endl;
63     }
64 }
65
66

```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef long long int ll;
5
6  int n,m;
7  int produtos[1010][2],P[110];
8  ll dp[1010][30];
9  inline ll solve(ll current, ll w){
10
11     if(current<0 or w <= 0) return 0LL;
12
13     if(dp[current][w]!=-1) return dp[current][w];
14
15     ll ans;
16     if(produtos[current][1]<=w)
17         ans = solve(current-1,w-produtos[current][1])+produtos[current][0];
18
19     ans = max(solve(current-1,w),ans);
20
21     return dp[current][w] = ans;
22 }
23
24 main(){
25     int t;
26
27     scanf("%d",&t);
28
29     for(int k=0;k<t;k++){
30
31         scanf("%d",&n);
32
33         for(int i=0;i<n;i++){
34             scanf("%d %d",&produtos[i][0], &produtos[i][1]);
35         }
36
37         scanf("%d",&m);
38         memset(dp,-1,sizeof dp);
39         ll ans = 0;
40         for(int i=0;i<m;i++){
41             scanf("%d",&P[i]);
42             ans += solve(n-1,P[i]);
43         }
44         printf("%lld\n",ans);
45     }
46 }
47
```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef long long ll;
5
6  ll w,T;
7  ll wt[40];
8  ll A[40][40];
9  int n;
10
11 inline ll knapsack(){
12
13     ll K[n+1][T+1];
14
15     for(int i=0;i<=n;i++){
16         for(int j=0;j<=T;j++){
17             if(i==0 or j==0)
18                 K[i][j] = 0;
19             else if(A[i-1][1] <= j){
20                 K[i][j] = max(A[i-1][0]+K[i-1][j-A[i-1][1]],K[i-1][j]);
21             }
22             else
23                 K[i][j] = K[i-1][j];
24         }
25     }
26     ll total B = 0;
27     ll total w = 0;
28     vector<pair<ll,ll> > V;
29     for(int i=n,j=T;i>0;i--){
30         if(K[i][j]!=K[i-1][j]){
31             V.push back(make pair(wt[i-1],A[i-1][0]));
32             ++total B;
33             j-=A[i-1][1];
34         }
35     }
36     cout << K[n][T] << endl
37         << total B << endl;
38     for(int i=V.size()-1;i>=0;i--){
39         cout << V[i].first << " " << V[i].second << endl;
40     }
41     main(){
42         bool f = true;
43         while(cin >> T >> w){
44             if(!f)
45                 cout << endl;
46             cin >> n;
47             for(int i=0;i<n;i++){
48                 cin >> A[i][1] >> A[i][0];
49                 wt[i] = A[i][1];
50                 A[i][1] = (2*w*A[i][1]) + (w*A[i][1]);
51             }
52             knapsack();
53             f = false;
54         }
55
56
57     }
58

```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  int w;
4  vector<int> car;
5  int dp[10001][10001];
6  int solve(int current,int s1,int s2){
7      if(current>=car.size())return dp[s1][s2] = 0LL;
8
9      if(dp[s1][s2]!=-1) return dp[s1][s2];
10
11     int ans = 0;
12     if(car[current]+s1 <= w)
13         ans = max(solve(current+1,s1+car[current],s2)+1,ans);
14     if(car[current]+s2 <= w)
15         ans = max(solve(current+1,s1,s2+car[current])+1,ans);
16
17     return dp[s1][s2] = ans;
18 }
19 void print(int current, int s1, int s2){
20
21     if(current>=car.size()) return;
22     if(s1+car[current] <=w and dp[s1][s2] - 1 == dp[s1+car[current]][s2]){
23         printf("port\n");
24         print(current+1,s1+car[current],s2);
25     }
26     else if(s2+car[current] <=w and dp[s1][s2]-1 == dp[s1][s2+car[current]]){
27         printf("starboard\n");
28         print(current+1,s1,s2+car[current]);
29     }
30 }
31
32 }
33 main(){
34     int n,aux;
35     scanf("%d",&n);
36     for(int k=0;k<n;k++){
37         scanf("%d",&w);
38         w*=100;
39         while(scanf("%d",&aux) and aux){
40             car.push back(aux);
41         }
42         memset(dp,-1,sizeof dp);
43         printf("%d\n",solve(0,0,0));
44
45         print(0,0,0);
46         car.clear();
47         if(k<n-1)
48             puts("");
49     }
50 }
51

```



```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef vector<vector<int> > vvi;
5  typedef vector<int> vi;
6  typedef vector<pair<int,int> > vii;
7  vvi Grafo(1005);
8  vii pontes;
9  int dfs low[1005];
10 int dfs num[1005];
11 int dfs parent[1005];
12 bool articulation vertex[1005];
13 int dfsNumberCounter,Children,dfsRoot,n,arti;
14 void print dfs(){
15     cout << pontes.size() << " critical links\n";
16     for(int i=0;i<pontes.size();i++){
17         cout << pontes[i].first << " - " << pontes[i].second << "\n";
18     }
19     cout << "\n";
20 }
21 void dfs(int u){
22     dfs low[u] = dfs num[u] = dfsNumberCounter++;
23     for(int i=0;i<Grafo[u].size();i++){
24         int v = Grafo[u][i];
25         if(dfs num[v]==-1){
26             dfs parent[v] = u;
27             if(u==dfsRoot)
28                 Children++;
29             dfs(v);
30
31             if(dfs low[v]>=dfs num[u]){
32                 articulation vertex[u] = true;
33             }
34             // if(dfs low[v]>dfs num[u])
35             //     pontes.push back(make pair(u,v));
36             dfs low[u] = min(dfs low[u],dfs low[v]);
37         }
38         else if(v!=dfs parent[u])
39             dfs low[u] = min(dfs low[u],dfs num[v]);
40     }
41 }
42 void reset(){
43     for(int i=0;i<n;i++){
44         Grafo[i].clear();
45         dfs num[i] = -1;
46         dfs low[i] = 0;
47         dfs parent[i] = 0;
48         articulation vertex[i] = false;
49     }
50     //pontes.clear();
51     dfsNumberCounter = 0;
52     arti = 0;
53 }
54 void solve(){
55     for(int i=0;i<n;i++){
56         if(dfs num[i]==-1){
57             dfsRoot = i;
58             Children = 0;
59             dfs(i);
60             articulation vertex[i] = (Children>1);
61         }
62     }
63     for(int i=0;i<n;i++){
64         if(articulation vertex[i])
65             arti++;
66     }
67     cout << arti << "\n";
68 }
69 main(){
70     int i,j,k,from,to,m;

```

```
71
72     while(cin >> n and n){
73         reset();
74         for(i=0;i<n+1;i++){
75             cin >> from;
76             if(from==0)
77                 break;
78             while(cin >> to and to){
79                 Grafo[from-1].push back(to-1);
80                 Grafo[to-1].push back(from-1);
81                 if(getchar()=='\n')
82                     break;
83             }
84         }
85         solve();
86     }
87 }
88
```

```

1  #include <bits/stdc++.h>
2  #define S second
3  #define F first
4  using namespace std;
5  typedef pair<int,int> ii;
6  typedef pair<int,ii> iii;
7  typedef vector<int> vi;
8  typedef vector<iii> viii;
9  int dy[] = {1,-1,0,0};
10 int dx[] = {0,0,1,-1};
11 bool Grafo[1001][1001];
12 int n,m;
13
14 bool valid(int i,int j){
15     if(Grafo[i][j]) return false;
16     if(i<0 or i>=n) return false;
17     if(j<0 or j>=m) return false;
18     return true;
19 }
20
21 int bfs(ii ini,ii dest){
22     queue<iii> q;
23     q.push(make pair(0,make pair(ini.F,ini.S)));
24     Grafo[ini.F][ini.S]=true;
25     while(!q.empty()){
26         iii x = q.front();
27         ii p = x.second;
28         q.pop();
29         if(x.second == dest)
30             return x.first;
31         for(int i=0;i<4;i++){
32             if(valid(p.first+dy[i],p.second+dx[i])){
33                 Grafo[p.first+dy[i]][p.second+dx[i]] = true;
34                 q.push(make pair(x.first+1,make pair(p.first+dy[i],p.second+dx[i])));
35             }
36         }
37     }
38 }
39
40 }
41
42 main(){
43     ios base::sync with stdio(0);
44     cin.tie(0);
45     int l,c,i,j,k,z,xi,yi,xd,yd,b;
46     while(cin >> n >> m and n and m){
47         cin >> z;
48         for(i=0;i<n;i++){
49             for(j=0;j<m;j++){
50                 Grafo[i][j] = false;
51             }
52         }
53         for(i=0;i<z;i++){
54             cin >> l >> b;
55             for(j=0;j<b;j++){
56                 cin >> c;
57                 Grafo[l][c] = true;
58             }
59         }
60         cin >> xi >> yi;
61         cin >> xd >> yd;
62         cout << bfs(make pair(xi,yi),make pair(xd,yd)) <<"\n";
63     }
64 }
65
66

```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef pair<int,int> ii;
4  typedef vector<ii> vii;
5  typedef vector<vii> vvii;
6  typedef vector<pair<int,ii> > viii;
7  typedef long long int ll;
8  viii Grafo,MST;
9  int parent[200001];
10 ll total;
11 int findset(int x){
12     if(x!=parent[x])
13         parent[x] = findset(parent[x]);
14     return parent[x];
15 }
16 void UNION(int x,int y){
17     parent[x] = parent[y];
18 }
19 void kruskal(){
20     int pu,pv;
21     sort(Grafo.begin(),Grafo.end());
22     for(int i=0;i<Grafo.size();i++){
23         pu = findset(Grafo[i].second.first);
24         pv = findset(Grafo[i].second.second);
25         if(pu!=pv){
26             total+=Grafo[i].first;
27             UNION(pu,pv);
28         }
29     }
30 }
31
32 main(){
33     int n,m,from,to,w;
34
35     while(cin >> n >> m and n and m){
36         Grafo.clear();
37         ll t = 0;
38         for(int i=0;i<m;i++){
39             cin >> from >> to >> w;
40             Grafo.push back(make pair(w,make pair(from,to)));
41             t+=w;
42         }
43         for(int i=0;i<n;i++){
44             parent[i] = i;
45         }
46         total = 0;
47         kruskal();
48         cout << t-total << endl;
49     }
50 }
51

```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef vector<vector<int> > vvi;
4  typedef vector<int> vi;
5  vvi Grafo(2005);
6  bool visitados[2005];
7  int n,m;
8  void dfs(int n){
9      visitados[n] = true;
10     for(int i=0;i<Grafo[n].size();i++){
11         if(!visitados[Grafo[n][i]])
12             dfs(Grafo[n][i]);
13     }
14 }
15
16
17 main(){
18     ios base::sync with stdio(0);
19     cin.tie(0);
20     int i,j,z,from,to,way;
21
22     while(cin >> n >> m and n and m){
23         for(j=0;j<n;j++){Grafo[j].clear();}
24         for(j=0;j<m;j++){
25             cin >> from >> to >> way;
26             Grafo[from-1].push back(to-1);
27             if(way==2)
28                 Grafo[to-1].push back(from-1);
29         }
30         bool falha = false;
31         for(j=0;j<n;j++){
32             memset(visitados,false,sizeof(visitados));
33             dfs(j);
34             for(int k=0;k<n;k++){if(visitados[k]==false){falha = true;break;}}
35             if(falha)
36                 break;
37         }
38         if(falha)
39             cout << 0 << endl;
40         else
41             cout << 1 << endl;
42     }
43 }
44
45
```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  typedef pair<int,int> ii;
5  typedef vector<ii> vii;
6  typedef vector<vii> vvii;
7  typedef vector<int> vi;
8  typedef vector<ll> vll;
9  vvii Grafo(1001);
10 int n,m;
11
12 bool ford(int ini){
13     vll dist(n+1,LLONG_MAX);
14     dist[ini] = 0;
15     for(int i=0;i<n;i++){
16         for(int k=0;k<n;k++){
17             for(int j=0;j<Grafo[k].size();j++){
18                 ii v = Grafo[k][j];
19                 if(dist[k]!=LLONG_MAX)
20                     dist[v.first] = min(dist[v.first],dist[k]+v.second);
21             }
22         }
23     }
24     // bool negative = false;
25     for(int i=0;i<n;i++){
26         for(int j = 0;j<Grafo[i].size();j++){
27             ii v = Grafo[i][j];
28             if(dist[v.first]!=LLONG_MAX and dist[v.first] > dist[i]+v.second){
29                 return true;
30             }
31         }
32     }
33     return false;
34 }
35 void reset(){
36     for(int i=0;i<n;i++){
37         Grafo[i].clear();
38     }
39 }
40
41 main(){
42     int k,from,to,w,i,j;
43
44     cin >> k;
45
46     for(i=0;i<k;i++){
47         cin >> n >> m;
48         reset();
49         for(j=0;j<m;j++){
50             cin >> from >> to >> w;
51             Grafo[from].push back(make pair(to,w));
52         }
53         if(ford(0))
54             cout << "possible\n";
55         else
56             cout << "not possible\n";
57     }
58 }
59

```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef long long int ll;
4  ll Grafo[101][101];
5  bool visitados[101];
6  int m;
7
8  main(){
9      int n,z,from,to,x,y;
10     scanf("%d",&z);
11     for(int v =0;v<z;v++){
12         scanf("%d", &m);
13         scanf("%d", &n);
14         for(int i=0;i<m;i++){
15             for(int j=0;j<m;j++){
16                 if(i==j)
17                     Grafo[i][j] = 0;
18                 else
19                     Grafo[i][j] = 100000000;
20             }
21         }
22         for(int q=0;q<n;q++){
23             scanf("%d %d",&from,&to);
24             Grafo[from][to] = 1;
25             Grafo[to][from] = 1;
26         }
27         scanf("%d %d",&x,&y);
28         for (int k = 0; k < m; k++){
29             for (int i = 0; i < m; i++){
30                 for (int j = 0; j < m; j++){
31                     Grafo[i][j] = min(Grafo[i][j], Grafo[i][k] + Grafo[k][j]);
32                 }
33             }
34         }
35         ll saida = 0;
36         for(int i=0;i<m;i++)
37             saida = max(Grafo[x][i]+Grafo[i][y],saida);
38
39         cout << "Case " << v+1<< ": " << saida << endl;
40     }
41 }
42
```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef long long int ll;
5  map<string,int> cidades;
6  map<int,string> vertices;
7  vector<vector<pair<int,int> > > Grafo(30);
8  int pai[30];
9  int n,m;
10
11 void dijkstra(string o,string des){
12
13     int dist[n];
14     for(int i=0;i<n;i++) dist[i] = 10101010;
15     priority queue<pair<int,int>, vector<pair<int,int> >, greater<pair<int,int> > >
16     pq;
17
18     pq.push(make pair(0,cidades[o]));
19     dist[cidades[o]] = 0;
20
21     while(!pq.empty()){
22
23         int d = pq.top().first;
24         int v = pq.top().second;
25         pq.pop();
26         for(int i=0;i<Grafo[v].size();i++){
27             if(dist[Grafo[v][i].first] > Grafo[v][i].second+d){
28                 pq.push(make pair(Grafo[v][i].second+d,Grafo[v][i].first));
29                 pai[Grafo[v][i].first] = v;
30                 dist[Grafo[v][i].first] = Grafo[v][i].second+d;
31             }
32         }
33     }
34
35 }
36
37 main(){
38
39     memset(pai,-1,sizeof pai);
40     string o,des,e,aux;
41     int d;
42     cin >> n;
43     cin.ignore();
44     for(int i=0;i<n;i++){
45         cin >> e;
46         cidades[e] = i;
47         vertices[i] = e;
48     }
49     cin >> m;
50     for(int i=0;i<m;i++){
51         cin.ignore();
52         cin >> e >> aux >> d;
53         Grafo[cidades[e]].push back(make pair(cidades[aux],d));
54         Grafo[cidades[aux]].push back(make pair(cidades[e],d));
55     }
56     cin >> o >> des;
57
58     dijkstra(o,des);
59     vector<string> ans;
60     while(pai[cidades[des]]!=-1){
61         ans.push back(des);
62         des = vertices[pai[cidades[des]]];
63     }
64     ans.push back(o);
65     for(int i=ans.size()-1;i>=0;i--){
66         if(i!=ans.size()-1)
67             cout << "- ";
68         cout << ans[i];
69     }

```



```
70     cout << endl;  
71 }  
72
```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef long long int ll;
5  ll n;
6  int moedas[] = {1,5,10,25,50};
7  ll dp[10][1000010];
8
9  inline ll solve(ll current,ll sum){
10
11     if(sum==0) return 1LL;
12     if(current < 0 or sum<0) return 0LL;
13
14     if(dp[current][sum]!=-1) return dp[current][sum];
15
16     ll ans = solve(current,sum-moedas[current])+solve(current-1,sum);
17
18     return dp[current][sum] = ans;
19 }
20
21
22 main(){
23     ios base::sync with stdio(0);
24     cin.tie(0);
25     memset(dp,-1,sizeof dp);
26     ll ans;
27     while(scanf("%lld",&n)!=EOF){
28         ans = solve(4,n);
29         cout << "There " << (ans==1? "is only ":"are ") << ans << (ans==1? "
30             way":" ways") << " to produce " << n << " cents change." << endl;
31     }
32 }
33
```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  #define EPS 1e-2
4
5  typedef long long int ll;
6  int M[] = {10000,5000,2000,1000,500,200,100,50,20,10,5};
7
8  ll dp[11][40100];
9  ll n;
10
11 ll solve(ll current, ll sum){
12     if(sum==0) return dp[current][sum] = 1LL;
13     if(current < 0 or sum < 0) return 0LL;
14
15     if(dp[current][sum]!=-1) return dp[current][sum];
16
17     ll ans = solve(current,sum-M[current]) + solve(current-1,sum);
18
19     return dp[current][sum] = ans;
20 }
21
22 }
23
24 main(){
25     float aux;
26     memset(dp,-1,sizeof dp);
27     while(cin >> aux and aux!= 0.00){
28         n = (ll)(aux*100);
29         if(fabs(aux*100 - n) > EPS)
30             n++;
31         printf("%6.2f %16lld\n", aux, solve(10,n));
32     }
33 }
34
35 }
```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef long long int ll;
5
6  ll BIT[100100];
7  int n;
8
9  inline void update(ll index){
10
11     for(;index<=n;index += index &(-index))
12         BIT[index] += 1;
13 }
14 inline ll query(ll index){
15     ll ans =0;
16     for(;index>0;index -= index & (-index))
17         ans += BIT[index];
18     return ans;
19 }
20
21
22 main(){
23     vector<ll> num;
24     scanf("%d",&n);
25     ll ans = 0,aux;
26     for(int i=1;i<=n;i++){
27         scanf("%lld",&aux);
28         num.push back(aux);
29     }
30     for(int i=num.size()-1;i>=0;i--){
31         update(num[i]);
32         ans += query(num[i]-1);
33     }
34     printf("%lld\n",ans);
35 }
36
37
```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  vector<vector<int> > Grafo(205);
5  int cores[205];
6  bool visitados[205];
7  int n,m;
8
9  bool bfs bi(){
10     memset(visitados,false,sizeof(visitados));
11     memset(cores,-1,sizeof(cores));
12     queue<int> F;
13     F.push(0);
14     cores[0] = false;
15     while(!F.empty()){
16         int aux = F.front();
17         F.pop();
18         if(!visitados[aux]){
19             visitados[aux] = true;
20             for(int i=0;i<Grafo[aux].size();i++){
21                 if(cores[Grafo[aux][i]]==-1)
22                     cores[Grafo[aux][i]] = 1-cores[aux];
23                 else if(cores[Grafo[aux][i]]==cores[aux])
24                     return false;
25                 cores[Grafo[aux][i]] = 1-cores[aux];
26                 F.push(Grafo[aux][i]);
27             }
28         }
29     }
30     return true;
31 }
32
33 main(){
34     int i,j,aux,from,to;
35
36     while(scanf("%d",&n) and n){
37         cin >> m;
38         for(i=0;i<n;i++){Grafo[i].clear();}
39         for(i=0;i<m;i++){
40             scanf("%d %d",&from,&to);
41             Grafo[from].push back(to);
42         }
43         if(bfs bi())
44             printf("BICOLORABLE.\n");
45         else
46             printf("NOT BICOLORABLE.\n");
47     }
48
49 }
50
```

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  #define EPS 1e-6
4
5  typedef long long int ll;
6
7
8  main(){
9      int p,q,r,s,t,u;
10
11
12      while(cin >> p >> q >> r >> s >> t >> u){
13          double ini= 0.000000,fim = 1.000000, mid = 0.500000;
14          bool achou = false;
15
16          while(ini<=fim){
17              double ans = p*exp(-mid)+q*sin(mid)+r*cos(mid)+s*tan(mid)+t*(mid*mid)+u;
18              if(fabs(ans) <= EPS){
19                  cout << fixed << setprecision(4) << mid << endl;
20                  achou = true;
21                  break;
22              }
23              else if(ans < EPS)
24                  fim = mid - 0.000000001;
25              else
26                  ini = mid + 0.000000001;
27              mid = (ini+fim)/2;
28          }
29          if(!achou)
30              cout << "No solution\n";
31      }
32
33
34 }
35
```

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef long long int ll;
5
6
7  ll D[10100];
8  ll n,num,X,mid;
9
10 ll binary_search(ll lo, ll hi){
11     mid = (lo+hi)/2;
12     if(lo > hi or mid == X) return -1;
13     if(num==D[mid]) return mid;
14     else if(num < D[mid]) return binary_search(lo,mid-1);
15     else if(num > D[mid]) return binary_search(mid+1,hi);
16 }
17
18 main(){
19     ios base::sync with stdio(0);
20     cin.tie(0);
21     ll y,ans,ans1,aux;
22
23     while(cin >> n){
24         for(int i=0;i<n;i++){
25             cin >> D[i];
26         }
27         cin >> y;
28         sort(D,D+n);
29         ans = -1;
30         ans1 = -1;
31         for(X=0;X<n;X++){
32             num = y - D[X];
33             aux = -1;
34             if(num < D[X])
35                 aux = binary_search(0,X-1);
36             if(aux != -1){
37                 if(ans == -1){
38                     ans = aux;
39                     ans1 = X;
40                 }
41                 else{
42                     if(abs(D[X]-D[aux]) < abs(D[ans]-D[ans1])){
43                         ans = aux;
44                         ans1 = X;
45                     }
46                 }
47             }
48         }
49         if(ans <= ans1)
50             cout << "Peter should buy books whose prices are " << D[ans] << " and " << D[ans1] << "." << endl;
51         else
52             cout << "Peter should buy books whose prices are " << D[ans1] << " and " << D[ans] << "." << endl;
53         cout << endl;
54     }
55 }
56

```

```
1  import java.math.BigInteger;
2  import java.util.Scanner;
3
4  public class Main{
5
6      public Main(){}
7
8      public static String reverse(String a){
9          String aux = "";
10
11          for(int i=a.length()-1;i>=0;i--){
12              aux = aux.concat(a.substring(i,i+1));
13          }
14          return aux;
15      }
16
17      public static void main(String[] args){
18          long n;
19          Scanner ler = new Scanner(System.in);
20          BigInteger a,b;
21          String s1,s2,aux,aux2;
22
23          n = ler.nextLong();
24
25          for(long i = 0;i< n;i++){
26              a = ler.nextBigInteger();
27              b = ler.nextBigInteger();
28
29              s1 = reverse(a.toString());
30              s2 = reverse(b.toString());
31              a = new BigInteger(s1);
32              b = new BigInteger(s2);
33
34              s1 = reverse(a.add(b).toString());
35              a = new BigInteger(s1);
36              System.out.println(a);
37          }
38      }
39  }
40
41  }
42
```



```
1
2 import java.math.BigInteger;
3 import java.util.Scanner;
4
5 public class Main{
6
7     public Main(){
8     }
9
10    public static void main(String[] args){
11
12        BigInteger b,ans;
13        int n=-1,a=-1,cont=1;
14        Scanner ler = new Scanner(System.in);
15
16        while(n!=0 && a!=0){
17            n = ler.nextInt();
18            a = ler.nextInt();
19            if(n==0 && a==0)
20                break;
21            ans = BigInteger.ZERO;
22            for(int i = 0;i<n;i++){
23                b = ler.nextBigInteger();
24                ans = ans.add(b);
25            }
26            b = ans.divide(BigInteger.valueOf(a));
27            System.out.println("Bill #"+cont+" costs "+ans.toString()+" : each
28            friend should pay "+b.toString()+"\n");
29            cont++;
30        }
31    }
32 }
33
34 }
35
```