

Dynamic Programming

- Matrix Multiplication
 - [UVa 10891 - Game of Sum](#)
 - [UVa 10003 - Cutting Sticks](#)
- Optimal Search Tree
 - [UVa 10304 - Optimal Binary Search Tree](#)
- Integer partition
 - [UVa 907 - Winterim Backpacking Trip](#)
- Longest Common Subsequence
 - [UVa 10066 - The Twin Towers](#)
 - [UVa 10192 - Vacation](#)
 - [UVa 10723 - Cyborg Genes](#)
 - [UVa 11151 - Longest Palindrome](#)
 - [UVa 12147 - DNA Sequences](#)
- Longest Common Substring
 - [UVa 1223 - Editor](#)
- Edit Distance
 - [UVa 10739 - String to Palindrome](#)
 - [UVa 1207 - AGTC](#)
- Maximum Sum Contiguous Subsequence
 - [UVa 10684 - The Jackpot](#)
 - [UVa 11059 - Maximum Product](#)
- Maximum Sum Sub-rectangle
 - [UVa 10827 - Maximum sum on a torus](#)
- Longest Increasing Subsequence
 - [UVa 473 - Raucous Rockers](#)
 - [UVa 11003 - Boxes](#)
 - [UVa 10635 - Prince and Princess](#)
 - [UVa 10154 - Weights and Measures](#)
 - [UVa 10051 - Tower of Cubes](#)
- Memoized DFS
 - [UVa 10259 - Hippiity Hopscotch](#)
- Knapsack - Counting Items
 - [UVa 1213 - Sum of Different Primes](#)
 - [UVa 1158 - CubesSquared](#)
- Knapsack - 0/1
 - [UVa 11658 - Best Coalitions](#)
 - [UVa 10930 - A-Sequence](#)
- Ad hoc

- [UVa 10444 - Multi-peg Towers of Hanoi](#)
- [UVa 11375 - Matches](#)
- [UVa 10703 - sqrt log sin](#)
- [UVa 1231 - ACORN](#)
- [UVa 1239 - Greatest K-Palindrome Substring](#)

dynamic-programming/473.cpp

```
1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #define MAX 10005
6  using namespace std;
7
8  int S[MAX], T[MAX];
9  char skip;
10 int main() {
11     int n,t,m,cases;
12     cin >> cases;
13     while(cases--) {
14         cin >> n >> t >> m;
15         memset(T, 0x3F, sizeof(T));
16
17         for(int i=1; i<=n; i++) {
18             cin >> S[i];
19             if (i<n) cin >> skip;
20         }
21
22         int k=0;
23         T[0] = 0;
24
25         for(int i=1;i<=n;i++) {
26             for(int j=k; j>=0; j--) {
27                 int add = 0;
28                 if ((T[j]%t)+S[i] > t) add = t-T[j]%t;
29                 if (T[j]+S[i]+add <= T[j+1]) {
30                     T[j+1] = T[j]+S[i]+add;
31                     k=max(k, j+1);
32                 }
33             }
34         }
35
36         int answer = 0;
37         for(int i=0; i<=k && T[i] <= m*t; i++)
38             answer = i;
39
40         cout << answer << endl;
41         if (cases) cout << endl;
42     }
43
44     return 0;
45 }
```

dynamic-programming/907.cpp

```
1  #define MAX 602
2  #include <iostream>
3  #include <cstring>
4  #include <climits>
5  using namespace std;
6
7  int T[MAX][MAX], S[MAX], n, k;
8
9  int main() {
10     while(cin >> n >> k) {
11         n++; k++;
12         memset(T, 0, sizeof(T));
13
14         S[0] = 0;
15         for(int i=1; i<=n; i++) {
16             cin >> S[i];
17         }
```

```

18
19     for(int i=1; i<=n; i++)
20         T[i][1] = T[i-1][1]+S[i];
21
22     for(int i=1; i<=k; i++)
23         T[1][i] = S[1];
24
25     for(int i=2; i<=n; i++) {
26         for(int j=2; j<=k; j++) {
27             T[i][j] = INT_MAX;
28             for(int x=1; x<i; x++)
29                 T[i][j] = min(T[i][j], max(T[x][j-1], T[i][1] - T[x][1]));
30         }
31     }
32
33     cout << T[n][k] << endl;
34 }
35 }

```

dynamic-programming/1158.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <vector>
4  using namespace std;
5
6  int K[400001];
7  vector<int> W;
8
9  int main() {
10
11     for(int i=1; i*i*i<=400000; i++)
12         W.push_back(i*i*i);
13
14     for(int a=1, i=1; a<=400000; i++, a+=i*i)
15         W.push_back(a);
16
17     memset(K, 0x3f, sizeof(K));
18     K[0] = 0;
19     for(int i=0; i<W.size(); i++)
20         for(int j=W[i]; j<=400000; j++)
21             K[j] = min(K[j], K[j-W[i]]+1);
22
23     int n;
24     while(cin >> n, n!=-1)
25         cout << K[n] << endl;
26
27     return 0;
28 }

```

dynamic-programming/1207.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #define MAX 1005
6  using namespace std;
7
8  int T[MAX][MAX];
9  string P, Q;
10
11
12 int main() {
13     int p, q;
14     while(cin >> p >> P >> q >> Q) {

```

```

15     for(int i=0; i<=p; i++) { T[i][0] = i; }
16     for(int i=0; i<=q; i++) { T[0][i] = i; }
17
18     for(int i=1; i<=p; i++) {
19         for(int j=1; j<=q; j++) {
20             if (P[i-1] == Q[j-1])
21                 T[i][j] = T[i-1][j-1];
22             else
23                 T[i][j] = min(min(T[i-1][j], T[i][j-1]), T[i-1][j-1])+1;
24         }
25     }
26
27     cout << T[p][q] << endl;
28 }
29
30 return 0;
31 }

```

dynamic-programming/1213.cpp

```

1  #include <iostream>
2  #include <vector>
3  #include <cstring>
4  using namespace std;
5
6  long K[20][1300];
7  bool P[1300];
8  vector<int> W();
9
10 int main() {
11     int n, k;
12
13     memset(P, true, sizeof(P));
14     P[0] = P[1] = false;
15     for(int i=2; i<1300; i++) {
16         if (P[i]) {
17             W.push_back(i);
18             for(int j=i*i; j<1300; j+=i)
19                 P[j] = false;
20         }
21     }
22
23     K[0][0] = 1;
24     for(int i=0; i<W.size(); i++)
25         for(int p=19; p>0; p--)
26             for(int j = W[i]; j<1300; j++)
27                 K[p][j]+=K[p-1][j-W[i]];
28
29     while(cin >> n >> k, n|k)
30         cout << K[k][n] << endl;
31 }

```

dynamic-programming/1223.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #define MAX 5001
5  using namespace std;
6
7  int T[MAX][MAX];
8
9  int main() {
10     int t; cin >> t; t=0;
11     string s;
12     while(cin >> s) {

```

```

13     int sz = s.size();
14     int maxx = 0;
15     for(int i=1; i<=sz; i++) {
16         for(int j=1; j<=sz; j++) {
17             if (s[i-1] == s[j-1] && i!=j)
18                 maxx = max(maxx, T[i][j] = T[i-1][j-1]+1);
19             else
20                 T[i][j] = 0;
21         }
22     }
23
24     cout << maxx << endl;
25 }
26
27 return 0;
28 }

```

dynamic-programming/1231.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #define MAX 2001
5  using namespace std;
6
7  int S[MAX][MAX], T[MAX][MAX], M[MAX];
8  char skip;
9  int main() {
10     int cases; cin >> cases;
11     while(cases--) {
12         memset(S, 0, sizeof(S));
13         memset(M, 0, sizeof(M));
14         int t, h, f;
15         cin >> t >> h >> f;
16
17         for(int i=0; i<t; i++) {
18             int k, a; cin >> k;
19             while(k--) {
20                 cin >> a;
21                 S[a][i]++;
22             }
23         }
24
25         for(int i=h; i>=0; i--) {
26             for(int j=0; j<t; j++) {
27                 int move = i+f<=h ? M[i+f] : 0;
28                 int stay = i+1<=h ? T[i+1][j] : 0;
29                 T[i][j] = max(move, stay) + S[i][j];
30                 M[i] = max(M[i], T[i][j]);
31             }
32         }
33
34         cout << M[0] << endl;
35     }
36
37     return 0;
38 }

```

dynamic-programming/1239.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #define MAX 1005
6  using namespace std;

```

```

7
8  int T[MAX][MAX];
9
10 int main() {
11     int t; cin >> t; t=0;
12     string P;
13     int k;
14     while(cin >> P >> k) {
15         int p = P.size();
16
17         int maxx=0;
18         for(int i=p; i>=1; i--) {
19             for(int j=i; j<=p; j++) {
20                 T[i][j] = T[i+1][j-1] + (P[i-1] == P[j-1] ? 0 : 1);
21
22                 if (T[i][j] <= k)
23                     maxx = max(maxx, j-i+1);
24             }
25         }
26
27         cout << maxx << endl;
28     }
29
30     return 0;
31 }

```

dynamic-programming/10003.cpp

```

1  #define MAX 1001
2  #include <iostream>
3  #include <cstring>
4  #include <climits>
5  using namespace std;
6
7  int T[MAX][MAX], S[MAX], n;
8  bool V[MAX][MAX];
9
10 int TT(int a, int b) {
11     if (a+1==b) return 0;
12     if (V[a][b]) return T[a][b];
13
14     int minn = INT_MAX;
15     for(int i=a+1; i<b; i++)
16         minn = min(minn, TT(a,i) + TT(i,b) + S[b]-S[a]);
17
18     V[a][b] = true;
19     return T[a][b] = minn;
20 }
21
22 int main() {
23     int t;
24     while(cin >> t, t) {
25         cin >> n;
26
27         memset(S, 0, sizeof(S));
28         memset(V, 0, sizeof(V));
29         S[0] = 0;
30         for(int i=1; i<=n; i++) {
31             cin >> S[i];
32         }
33         S[n+1] = t;
34
35         cout << "The minimum cutting is " << TT(0, n+1) << "." << endl;
36
37     }
38 }

```

dynamic-programming/10051.cpp

```
1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #include <climits>
6  #define MAX 501
7  #define MAXC 101
8  using namespace std;
9
10 int T[MAX][MAXC], F[MAX][MAXC], P[MAX][MAXC];
11 int A[6];
12
13 string translate(int side) {
14     switch(side) {
15         case 0: return "front";
16         case 1: return "back";
17         case 2: return "left";
18         case 3: return "right";
19         case 4: return "top";
20         case 5: return "bottom";
21     }
22 }
23
24 void print(int first, int k) {
25     if (k==0) return;
26
27     print(F[k][first], k-1);
28     cout << T[k][first] << " " << translate(P[k][first]) << endl;
29 }
30
31 int main() {
32     int n, n2, t;
33     while(cin >> n, n) {
34         if (t++) cout << endl;
35         cout << "Case #" << t << endl;
36
37         memset(T, 0, sizeof(T));
38
39         for(int i=1; i<=MAXC; i++) {
40             T[0][i] = 1;
41         }
42         int k = 0;
43
44         for(int cube=1; cube<=n; cube++) {
45             for(int i=0; i<6; i++) cin >> A[i];
46             int newk = k;
47             for(int j=k; j>=0; j--) {
48                 for(int i=0; i<6; i++) {
49                     int other = (i/2*2)+(1-i%2);
50                     if (T[j][A[i]] && !T[j+1][A[other]]) {
51                         T[j+1][A[other]] = cube;
52                         F[j+1][A[other]] = A[i];
53                         P[j+1][A[other]] = i;
54                         newk = max(newk, j+1);
55                     }
56                 }
57             }
58             k=newk;
59         }
60
61         cout << k << endl;
62
63         int first=0;
64         for(int i=1; i<=100; i++)
65             if (T[k][i]) first=i;
66     }
```



```

67 |         print(first, k);
68 |     }
69 |
70 |     return 0;
71 | }

```

dynamic-programming/10066.cpp

```

1 | #include <iostream>
2 | #include <string>
3 | #include <cstring>
4 | #include <cmath>
5 | #define MAX 105
6 | using namespace std;
7 |
8 | int T[MAX][MAX];
9 | int P[MAX], Q[MAX];
10 |
11 | int main() {
12 |     int p, q, tt=0;
13 |     while(cin >> p >> q, tt++, p&&q) {
14 |         memset(T, 0, sizeof(T));
15 |
16 |         for(int i=0; i<p;i++) cin >> P[i];
17 |         for(int i=0; i<q;i++) cin >> Q[i];
18 |
19 |         for(int i=0; i<=p; i++) T[i][0] = 0;
20 |         for(int i=0; i<=q; i++) T[0][i] = 0;
21 |
22 |         for(int i=1; i<=p; i++) {
23 |             for(int j=1; j<=q; j++) {
24 |                 if (P[i-1] == Q[j-1])
25 |                     T[i][j] = T[i-1][j-1] + 1;
26 |                 else
27 |                     T[i][j] = max(T[i-1][j], T[i][j-1]);
28 |             }
29 |         }
30 |         cout << "Twin Towers #" << tt << endl;
31 |         cout << "Number of Tiles : " << T[p][q] << endl;
32 |         cout << endl;
33 |     }
34 |
35 |     return 0;
36 | }

```

dynamic-programming/10154.cpp

```

1 | #include <iostream>
2 | #include <string>
3 | #include <cstring>
4 | #include <cmath>
5 | #include <climits>
6 | #include <vector>
7 | #include <algorithm>
8 | #define MAX 10005
9 | using namespace std;
10 |
11 | struct Turtle {
12 |     int w,c;
13 |     Turtle() {}
14 |     Turtle(int w, int c) : w(w), c(c) {}
15 | };
16 |
17 | bool compare(const Turtle& a, const Turtle& b) {
18 |     return a.c > b.c;
19 | }

```

```

20
21 vector<Turtle> V;
22 int T[MAX];
23 int main() {
24     int w, c, k=0;
25     T[0] = INT_MAX;
26
27     while(cin >> w >> c) {
28         V.push_back(Turtle(w, c-w));
29     }
30     sort(V.begin(), V.end(), compare);
31
32     for(int i=0; i<V.size(); i++) {
33         int w = V[i].w, c = V[i].c;
34
35         for(int j=k; j>=0; j--) {
36             int next = min(T[j]-w, c);
37             if (next >= T[j+1]) {
38                 T[j+1] = next;
39                 k=max(k, j+1);
40             }
41         }
42     }
43     cout << k << endl;
44
45     return 0;
46 }

```

dynamic-programming/10192.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #define MAX 1005
6  using namespace std;
7
8  int T[MAX][MAX];
9  string P, Q;
10
11 int main() {
12     int p, q, tt=0;
13     while(getline(cin, P), P!="#") {
14         tt++;
15         getline(cin, Q);
16         int p = P.size(), q = Q.size();
17
18         memset(T, 0, sizeof(T));
19
20         for(int i=0; i<=p; i++) T[i][0] = 0;
21         for(int i=0; i<=q; i++) T[0][i] = 0;
22
23         for(int i=1; i<=p; i++) {
24             for(int j=1; j<=q; j++) {
25                 if (P[i-1] == Q[j-1])
26                     T[i][j] = T[i-1][j-1] + 1;
27                 else
28                     T[i][j] = max(T[i-1][j], T[i][j-1]);
29             }
30         }
31         cout << "Case #" << tt << ": you can visit at most " << T[p][q] << " cities." << endl;
32     }
33
34     return 0;
35 }

```

dynamic-programming/10259.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #include <climits>
6  #define MAX 101
7  using namespace std;
8
9  int T[MAX][MAX], M[MAX][MAX], n, k;
10
11 int walk(int x, int y, int curr) {
12     if (x < 0 || x >= n || y < 0 || y >= n) return 0;
13     if (T[x][y] <= curr) return 0;
14     if (M[x][y] >= 0) return M[x][y];
15
16     int maxx = 0;
17     for(int i=1; i<=k; i++) {
18         maxx = max(maxx, walk(x-i, y, T[x][y])+T[x][y]);
19         maxx = max(maxx, walk(x+i, y, T[x][y])+T[x][y]);
20         maxx = max(maxx, walk(x, y-i, T[x][y])+T[x][y]);
21         maxx = max(maxx, walk(x, y+i, T[x][y])+T[x][y]);
22     }
23     return M[x][y] = maxx;
24 }
25
26 int main() {
27     int t;
28     cin >> t;
29     while(t-->0) {
30         cin >> n >> k;
31         memset(M, -1, sizeof(M));
32         for(int i=0; i<n; i++)
33             for(int j=0; j<n; j++)
34                 cin >> T[i][j];
35
36         cout << walk(0,0, -1) << endl;
37         if (t) cout << endl;
38     }
39
40     return 0;
41 }

```

dynamic-programming/10304.cpp

```

1  #define MAX 252
2  #include <iostream>
3  #include <cstring>
4  #include <climits>
5  using namespace std;
6
7  int T[MAX][MAX], S[MAX], n;
8  bool V[MAX][MAX];
9
10 int TT(int a, int b) {
11     if (b < a) return 0;
12     if (V[a][b]) return T[a][b];
13
14     int minn = INT_MAX;
15     for(int i=a; i<=b; i++)
16         minn = min(minn, TT(a,i-1) + TT(i+1,b) + (S[b]-S[a-1])-(S[i]-S[i-1]));
17
18     V[a][b] = true;
19     return T[a][b] = minn;
20 }
21
22 int main() {
23     while(cin >> n) {
24         memset(V, 0, sizeof(V));

```

```

25     S[0] = 0;
26     for(int i=1; i<=n; i++) {
27         cin >> S[i];
28         S[i] += S[i-1];
29     }
30
31     cout << TT(1, n) << endl;
32
33 }
34 }

```

dynamic-programming/10444.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #include <climits>
6  #define MAX 205
7  using namespace std;
8
9  int T[MAX][MAX];
10 int main() {
11     int n=201, p=21, t=0;
12
13     for(int i=0; i<=n; i++) {
14         if (i<31)
15             T[i][3] = (1<<i)-1;
16         else
17             T[i][3] = INT_MAX; //avoid overflow
18     }
19
20     for(int i=1; i<=n; i++) {
21         for(int j=4; j<=p; j++) {
22             if (i<j) {
23                 T[i][j] = 2*i-1;
24             } else {
25                 int minn = INT_MAX;
26                 for(int k=1; k<=i; k++) {
27                     int value = 2*T[k][j]+T[i-k][j-1];
28                     if (value >= 0) //avoid overflow
29                         minn = min(minn, value);
30                 }
31                 T[i][j] = minn;
32             }
33         }
34     }
35
36     while(cin >> n >> p, n | p) {
37         cout << "Case " << ++t << ": " << T[n][p] << endl;
38     }
39
40     return 0;
41 }

```

dynamic-programming/10635.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <cmath>
5  #include <algorithm>
6  #define MAX 255*255
7  using namespace std;
8

```

```

9 | int P[MAX], Q[MAX], M[MAX];
10
11 | int main() {
12 |     int n, p, q, tt=0, temp;
13 |     cin >> n;
14 |     while(cin >> n >> p >> q) {
15 |         memset(P, 0, sizeof(P));
16 |         q++; p++;
17 |         for(int i=1;i<=p;i++) {
18 |             cin >> temp;
19 |             P[temp] = i;
20 |         }
21
22 |         for (int i=1;i<=q;i++) {
23 |             cin >> temp;
24 |             Q[i] = P[temp];
25 |         }
26
27 |         int k=0; M[0]=0;
28 |         for(int i=1;i<=q;i++) {
29 |             if (Q[i] > M[k]) {
30 |                 k++; M[k] = Q[i];
31 |             } else {
32 |                 int j = (int)(lower_bound(M, M+k+1, Q[i])-M);
33 |                 if (Q[i] > M[j]) j++;
34 |                 M[j] = Q[i];
35 |             }
36 |         }
37
38 |         cout << "Case " << ++tt << ": " << k << endl;
39 |     }
40
41 |     return 0;
42 | }

```

dynamic-programming/10684.cpp

```

1 | #include <iostream>
2 | #include <cmath>
3 | #define MAX 1005
4 | using namespace std;
5
6 | int main() {
7 |     int n, a;
8 |     while(cin >> n, n) {
9 |         int t=0, s=0;
10 |         for(int i=0;i<n;i++) {
11 |             cin >> a;
12 |             if (s+a>=0)
13 |                 t = max(t, s+=a);
14 |             else
15 |                 s = 0;
16 |         }
17 |         if (s>0) {
18 |             cout << "The maximum winning streak is " << t << "." << endl;
19 |         } else {
20 |             cout << "Losing streak." << endl;
21 |         }
22 |     }
23
24 |     return 0;
25 | }

```

dynamic-programming/10723.cpp

```

1 | #include <iostream>

```

```

2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #define MAX 1005
6  using namespace std;
7
8  int T[MAX][MAX], D[MAX][MAX];
9  string P, Q;
10
11 int main() {
12     int p, q, t, tt=0;
13     cin >> t;
14     getline(cin, P);
15     while(tt++ < t) {
16         getline(cin, P);
17         getline(cin, Q);
18         int p = P.size(), q = Q.size();
19
20         for(int i=0; i<=p; i++) { T[i][0] = 0; D[i][0] = 1; }
21         for(int i=0; i<=q; i++) { T[0][i] = 0; D[0][i] = 1; }
22
23         for(int i=1; i<=p; i++) {
24             for(int j=1; j<=q; j++) {
25                 D[i][j] = 0;
26                 if (P[i-1] == Q[j-1]) {
27                     T[i][j] = T[i-1][j-1] + 1;
28                     D[i][j] = D[i-1][j-1];
29                 }
30                 else {
31                     T[i][j] = max(T[i-1][j], T[i][j-1]);
32                     if (T[i-1][j] == T[i][j]) D[i][j] += D[i-1][j];
33                     if (T[i][j-1] == T[i][j]) D[i][j] += D[i][j-1];
34                 }
35             }
36         }
37
38         cout << "Case #" << tt << ": " << p+q-T[p][q] << " " << D[p][q] << endl;
39     }
40
41     return 0;
42 }

```

dynamic-programming/10739.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #define MAX 1005
6  using namespace std;
7
8  int T[MAX][MAX];
9  string P, Q;
10
11 int main() {
12     int p, q, t, tt=0;
13     cin >> t;
14     getline(cin, P);
15     while(tt++ < t) {
16         getline(cin, P);
17         Q = string(P.rbegin(), P.rend());
18         int p = P.size(), q = Q.size();
19
20         for(int i=0; i<=p; i++) { T[i][0] = i; }
21         for(int i=0; i<=q; i++) { T[0][i] = i; }
22
23         for(int i=1; i<=p; i++) {
24             for(int j=1; j<=q; j++) {

```

```

25         if (P[i-1] == Q[j-1])
26             T[i][j] = T[i-1][j-1];
27         else
28             T[i][j] = min(min(T[i-1][j], T[i][j-1]), T[i-1][j-1])+1;
29     }
30 }
31
32     cout << "Case " << tt << ": " << T[p][q]/2 << endl;
33 }
34
35     return 0;
36 }

```

dynamic-programming/10827.cpp

```

1  #include <iostream>
2  #include <climits>
3  #define MAX 160
4  using namespace std;
5
6  int T[MAX][MAX];
7
8  int main() {
9      int n, a, cases;
10     cin >> cases;
11     while(cin >> n) {
12         for(int i=1; i<=n; i++) {
13             for(int j=1; j<=n; j++) {
14                 cin >> T[i][j];
15                 T[i+n][j] = T[i][j];
16             }
17         }
18
19         for(int i=1; i<=2*n; i++)
20             for(int j=1; j<=n; j++)
21                 T[i][j] += T[i-1][j];
22
23         int t = 0;
24         for(int i=1; i<=2*n; i++) {
25             for(int j=i; j<=min(i+n-1, 2*n); j++) {
26                 int smax=0, smin=0, ssum=0, tmax=0, tmin=0;
27                 for(int k=1; k<=n; k++)
28                     ssum += T[j][k] - T[i-1][k];
29
30                 for(int k=1; k<=n; k++) {
31                     int a = T[j][k] - T[i-1][k];
32                     smax += a;
33                     smin += a;
34
35                     tmax = max(tmax, smax);
36                     tmin = min(tmin, smin);
37
38                     if (smax < 0) smax = 0;
39                     if (smin > 0) smin = 0;
40                 }
41                 t = max(t, max(tmax, ssum-tmin));
42             }
43         }
44
45         cout << t << endl;
46     }
47
48     return 0;
49 }

```

dynamic-programming/10891.cpp

```

1  #define MAX 101
2  #include <iostream>
3  #include <cstring>
4  #include <climits>
5  using namespace std;
6
7  int T[MAX][MAX], S[MAX], n;
8  bool V[MAX][MAX];
9
10 int TT(int a, int b) {
11     if (b<a) return 0;
12     if (V[a][b]) return T[a][b];
13
14     int maxx = INT_MIN;
15     for(int i=a; i<=b; i++)
16         maxx = max(maxx, S[b]-S[a-1] - TT(i+1,b));
17
18     for(int i=b; i>=a; i--)
19         maxx = max(maxx, S[b]-S[a-1] - TT(a,i-1));
20
21     V[a][b] = true;
22     return T[a][b] = maxx;
23 }
24
25 int main() {
26     while(cin >> n, n) {
27         memset(S, 0, sizeof(S));
28         memset(V, 0, sizeof(V));
29         S[0] = 0;
30         for(int i=1; i<=n; i++) {
31             cin >> S[i];
32             S[i] += S[i-1];
33         }
34
35         cout << 2*TT(1, n)-S[n]-S[0] << endl;
36
37     }
38 }

```

dynamic-programming/10930.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <iomanip>
4  using namespace std;
5
6  int K[30001];
7
8  int main() {
9     int n, t=0, w;
10     while(t++, cin >> n) {
11         cout << "Case #" << t << ":";
12         memset(K, 0, sizeof(K));
13
14         bool ok=true;
15
16         K[0] = 1; int last = 0;
17         for(int i=1; i<=n; i++) {
18             cin >> w; cout << " " << w;
19             ok &= !K[w] && w > last;
20             for(int j=10000; j>=w; j--)
21                 if (K[j-w])
22                     K[j] = 1;
23             last = w;
24         }
25         cout << endl;
26         cout << "This is" << (ok?"":" not") << " an A-sequence." << endl;
27     }
}

```



```

28 |
29 |     return 0;
30 | }

```

dynamic-programming/11003.cpp

```

1 | #include <iostream>
2 | #include <string>
3 | #include <cstring>
4 | #include <cmath>
5 | #include <climits>
6 | #define MAX 10005
7 | using namespace std;
8 |
9 | int T[MAX];
10 | int main() {
11 |     int n, w, c;
12 |     while(cin >> n, n) {
13 |         memset(T, 0, sizeof(T));
14 |
15 |         int k = 0;
16 |         T[0] = INT_MAX;
17 |         for(int i=1; i<=n; i++) {
18 |             cin >> w >> c;
19 |             for(int j=k; j>=0; j--) {
20 |                 int next = min(T[j]-w, c);
21 |                 if (next >= T[j+1]) {
22 |                     T[j+1] = next;
23 |                     k=max(k, j+1);
24 |                 }
25 |             }
26 |         }
27 |
28 |         cout << k << endl;
29 |     }
30 |
31 |     return 0;
32 | }

```

dynamic-programming/11059.cpp

```

1 | #include <iostream>
2 | #include <climits>
3 | #include <cmath>
4 | #define MAX 1005
5 | using namespace std;
6 |
7 | int main() {
8 |     long long n, a, t=0;
9 |     while(cin >> n) {
10 |         long long maxx=0, newneg=0, newpos=0, spos=1, sneg=1;
11 |         bool valid = false;
12 |         for(int i=0; i<n; i++) {
13 |             cin >> a;
14 |             if (spos*a>0) {
15 |                 valid = true;
16 |                 spos*=a;
17 |             } else {
18 |                 newneg = spos*a;
19 |                 spos = 1;
20 |             }
21 |
22 |             if (sneg*a<0) {
23 |                 sneg*=a;
24 |             } else {
25 |

```

```

26         if (sneg*a>0) valid = true;
27         newpos = sneg*a;
28         sneg = 1;
29     }
30
31     maxx = max(maxx, spos = max(spos, newpos));
32     sneg = min(sneg, newneg);
33     newpos = newneg = 0;
34 }
35 if (!valid) maxx = 0;
36 cout << "Case #" << ++t << ": The maximum product is " << maxx << "." << endl;
37 cout << endl;
38 }
39
40 return 0;
41 }

```

dynamic-programming/11151.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #define MAX 1005
6  using namespace std;
7
8  int T[MAX][MAX];
9  string P, Q;
10
11 int main() {
12     int p, q, t;
13     cin >> t;
14     getline(cin, P);
15     while(t--) {
16         getline(cin, P);
17         Q = string(P.rbegin(), P.rend());
18         int p = P.size(), q = Q.size();
19
20         for(int i=0; i<=p; i++) { T[i][0] = 0; }
21         for(int i=0; i<=q; i++) { T[0][i] = 0; }
22
23         for(int i=1; i<=p; i++) {
24             for(int j=1; j<=q; j++) {
25                 if (P[i-1] == Q[j-1]) {
26                     T[i][j] = T[i-1][j-1] + 1;
27                 }
28                 else {
29                     T[i][j] = max(T[i-1][j], T[i][j-1]);
30                 }
31             }
32         }
33
34         cout << T[p][q] << endl;
35     }
36
37     return 0;
38 }

```

dynamic-programming/11375.cpp

```

1  #include <iostream>
2  #include <vector>
3  #include <cstring>
4  using namespace std;
5
6  int K[] = {6, 2, 5, 5, 4, 5, 6, 3, 7, 6};

```

```

7 | vector<int> T[2001][10];
8 |
9 | void add(vector<int> &a, const vector<int> &b) {
10 |     int carry = 0;
11 |     for(int i=0; i<max(a.size(), b.size()); i++) {
12 |         int aa = i<a.size()?a[i]:0;
13 |         int bb = i<b.size()?b[i]:0;
14 |         int cc = aa+bb+carry;
15 |         if (i >= a.size()) a.push_back(0);
16 |         a[i] = cc%10;
17 |         carry = cc/10;
18 |     }
19 |     if (carry)
20 |         a.push_back(carry);
21 | }
22 |
23 | int main() {
24 |     vector<int> one; one.push_back(1);
25 |
26 |     for(int i=2; i<2001; i++) {
27 |         for(int j=0; j<10; j++)
28 |             if (i>=K[j]) {
29 |                 add(T[i][j], one);
30 |                 for(int k=0; k<10; k++)
31 |                     add(T[i][j], T[i-K[j]][k]);
32 |             }
33 |     }
34 |
35 |     int n;
36 |     while(cin >> n) {
37 |         vector<int> ans = n>=6?one:vector<int>();
38 |         for(int i=1; i<10; i++)
39 |             add(ans, T[n][i]);
40 |
41 |         for(int i=ans.size()-1; i>=0; i--) {
42 |             cout << ans[i];
43 |         }
44 |         if (ans.size()==0) cout << 0;
45 |         cout << endl;
46 |     }
47 |
48 |     return 0;
49 | }

```

dynamic-programming/11658.cpp

```

1 | #include <iostream>
2 | #include <cstring>
3 | #include <iomanip>
4 | using namespace std;
5 |
6 | int K[10001], W[102];
7 |
8 | int main() {
9 |     int n, x, a, b;
10 |
11 |     while(cin >> n >> x, n|x) {
12 |         memset(K, 0, sizeof(K));
13 |
14 |         for(int i=1; i<=n; i++) {
15 |             cin >> a; cin.ignore(); cin >> b;
16 |             W[i] = a*100+b;
17 |         }
18 |
19 |         K[W[x]] = 1;
20 |         for(int i=1; i<=n; i++) {
21 |             if (i==x) continue;
22 |             for(int j=10000; j>=W[i]; j--)

```

```

23         if (K[j-W[i]])
24             K[j] = 1;
25     }
26
27     int maxx = 0;
28     for(int i=5001; i<=10000; i++) {
29         if (K[i]) {
30             maxx = i;
31             break;
32         }
33     }
34
35     cout << fixed << setprecision(2) << (W[x]/((double)maxx)*100.0) << endl;
36 }
37
38 return 0;
39 }

```

dynamic-programming/11703.cpp

```

1  #include <iostream>
2  #include <cmath>
3  #include <cstring>
4  #include <cassert>
5  using namespace std;
6
7  int K[1000001];
8
9  int main() {
10     K[0] = 1;
11     for(int i=1; i<1000001; i++) {
12         int a = (int)(i-sqrt(i));
13         int b = (int)log(i);
14         int c = (int)(i*pow(sin(i), 2));
15         K[i] = (K[a] + K[b] + K[c])%1000000;
16     }
17
18     int n;
19     while(cin >> n, n>-1)
20         cout << K[n] << endl;
21
22     return 0;
23 }

```

dynamic-programming/12147.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <cstring>
4  #include <cmath>
5  #define MAX 1005
6  using namespace std;
7
8  int T[MAX][MAX];
9  int S[MAX][MAX];
10 string P, Q;
11
12 int main() {
13     int k;
14     while(cin >> k, k) {
15         cin >> P >> Q;
16         int p = P.size(), q = Q.size();
17
18         for(int i=0; i<=p; i++) T[i][0] = S[i][0] = 0;
19         for(int i=0; i<=q; i++) T[0][i] = S[0][i] = 0;
20

```

```

21     for(int i=1; i<=p; i++) {
22         for(int j=1; j<=q; j++) {
23             if (P[i-1] == Q[j-1])
24                 S[i][j] = S[i-1][j-1] + 1;
25             else
26                 S[i][j] = 0;
27         }
28     }
29
30     for(int i=1; i<=p; i++) {
31         for(int j=1; j<=q; j++) {
32             T[i][j] = max(T[i-1][j], T[i][j-1]);
33
34             for(int s=k; s<=S[i][j]; s++)
35                 T[i][j] = max(T[i][j], T[i-s][j-s]+s);
36         }
37     }
38     cout << T[p][q] << endl;
39 }
40
41 return 0;
42 }

```

Graphs

- DFS
 - [UVa 10243 - Fire! Fire! Fire!](#)
 - [UVa 1220 - Party at Hali-Bula](#)
 - [UVa 12186 - Another Crisis](#)
 - [UVa 273 - Jack Straws](#)
 - [UVa 1216 - The Bug Sensor Problem](#)
 - [UVa 1197 - The Suspects](#)
- Flood Fill
 - [UVa 11110 - Equidivisions](#)
 - [UVa 11518 - Dominos 2](#)
- Shortest Path (BFS)
 - [UVa 314 - Robot](#)
 - [UVa 321 - The New Villa](#)
 - [UVa 627 - The Net](#)
 - [UVa 12101 - Prime Path](#)
 - [UVa 10044 - Erdos Number](#)
 - [UVa 12260 - Unlock the Lock](#)
 - [UVa 12135 - Switch Bulbs](#)
 - [UVa 298 - Race Tracks](#)
- Maximum Flow (Ford-Fulkerson with DFS)
 - [UVa 820 - Internet Bandwidth](#)
 - [UVa 10480 - Sabotage](#)
 - [UVa 10092 - The Problem with the Problem Setter](#)
 - [UVa 10511 - Counselling](#)
- Bipartite Matching/Konig Theorem
 - [UVa 11419 - SAM I AM](#)
 - [UVa 12168 - Cat vs. Dog](#)
 - [UVa 12159 - Gun Fight](#)
- Minimum Spanning Tree (Prim)
 - [UVa 1235 - Anti Brute Force Lock](#)
 - [UVa 1208 - Oreon](#)
 - [UVa 908 - Re-connecting Computer Sites](#)
- Minimum Spanning Tree (Prim with Priority Queue)
 - [UVa 11631 - Dark roads](#)
 - [UVa 11733 - Airports](#)
 - [UVa 11747 - Heavy Cycle Edges](#)
 - [UVa 10397 - Connect the Campus](#)
 - [UVa 1234 - RACING](#)
 - [UVa 1174 - IP-TV](#)
- Strongly Connected Components
 - [UVa 11838 - Come and Go](#)
 - [UVa 11709 - Trust Groups](#)
 - [UVa 1223 - Sub-Dictionary](#)
- Finding Articulation Points/Bridges
 - [UVa 315 - Network](#)

- [UVa 769 - Critical Links](#)
- Topological Sorting
 - [UVa 11686 - Pick up Sticks](#)
 - [UVa 1263 - Mines](#)
 - [UVa 11770 - Lighting Away](#)
- Minimum Path (Floyd-Warshall)
 - [UVa 10278 - Fire Station](#)
 - [UVa 12173 - Randomly-priced Tickets](#)
 - [UVa 1056 - Degrees of Separation](#)
- Minimum Path (Dijkstra)
 - [UVa 10986 - Sending email](#)
 - [UVa 10389 - Subway](#)
 - [UVa 11833 - Route Change](#)
 - [UVa 12144 - Almost Shortest Path](#)
 - [UVa 1247 - Interstar Transport](#)

graphs/273.cpp

```
1  #include <iostream>
2  #include <cstring>
3  #define MAX 100002
4  using namespace std;
5
6  static bool segment(int xi, int yi, int xj, int yj,
7                     int xk, int yk) {
8      return (xi <= xk || xj <= xk) && (xk <= xi || xk <= xj) &&
9             (yi <= yk || yj <= yk) && (yk <= yi || yk <= yj);
10 }
11
12 static char direction(int xi, int yi, int xj, int yj,
13                      int xk, int yk) {
14     int a = (xk - xi) * (yj - yi);
15     int b = (xj - xi) * (yk - yi);
16     return a < b ? -1 : a > b ? 1 : 0;
17 }
18
19 bool intersect(int x1, int y1, int x2, int y2, int x3, int y3, int x4, int y4) {
20     char d1 = direction(x3, y3, x4, y4, x1, y1);
21     char d2 = direction(x3, y3, x4, y4, x2, y2);
22     char d3 = direction(x1, y1, x2, y2, x3, y3);
23     char d4 = direction(x1, y1, x2, y2, x4, y4);
24     return (((d1 > 0 && d2 < 0) || (d1 < 0 && d2 > 0)) &&
25            ((d3 > 0 && d4 < 0) || (d3 < 0 && d4 > 0))) ||
26            (d1 == 0 && segment(x3, y3, x4, y4, x1, y1)) ||
27            (d2 == 0 && segment(x3, y3, x4, y4, x2, y2)) ||
28            (d3 == 0 && segment(x1, y1, x2, y2, x3, y3)) ||
29            (d4 == 0 && segment(x1, y1, x2, y2, x4, y4));
30 }
31
32 int G[20][20], V[20], A[20], B[20], C[20], D[20], n;
33
34 int dfs(int v, int comp) {
35     V[v] = comp;
36     for(int i=1; i<=n; i++)
37         if (!V[i] && G[v][i])
38             dfs(i, comp);
39 }
40
41 int main() {
42     int t; cin >> t; t=0;
43     while(cin >> n) {
44         memset(G, 0, sizeof(G));
45         memset(V, 0, sizeof(V));
46         for(int i=1; i<=n; i++) {
47             cin >> A[i] >> B[i] >> C[i] >> D[i];
48             for(int j=1; j<i; j++)
49                 G[i][j] = G[j][i] = intersect(A[i], B[i], C[i], D[i], A[j], B[j], C[j], D[j]);
50         }
51
52         int compn = 0;
53         for(int i=1; i<=n; i++)
54             if (!V[i])
55                 dfs(i, ++compn);
56
57         if (t++) cout << endl;
58         int a, b;
59         while(cin >> a >> b, a|b) {
60             cout << (V[a] == V[b]?"CONNECTED":"NOT CONNECTED") << endl;
61         }
62     }
63     return 0;
64 }
```


graphs/298.cpp

```
1  #include <iostream>
2  #include <cstring>
3  #include <queue>
4  #define MAX 30
5  using namespace std;
6
7  bool V[MAX][MAX][7][7];
8  int X, Y;
9
10 struct Step {
11     int x, y, a, b, v;
12     Step() {}
13     Step(int x, int y, int a, int b, int v) : x(x), y(y), a(a), b(b), v(v) {}
14
15     bool valid() {
16         return x>=0 && x<X && y>=0 && y<Y && a >= -3 && a <= 3 && b >= -3 && b <= 3 && !V[x][y][a+
17     }
18
19     void mark() {
20         V[x][y][a+3][b+3] = true;
21     }
22
23     Step go(int mx, int my) {
24         return Step(x+a+mx, y+b+my, a+mx, b+my, v+1);
25     }
26 };
27
28 int main() {
29     int t; cin >> t; t=0;
30     while(cin >> X >> Y) {
31         memset(V, 0, sizeof(V));
32         int x1, y1, x2, y2;
33         cin >> x1 >> y1 >> x2 >> y2;
34
35         int p, px1, px2, py1, py2;
36         cin >> p;
37         while(p-->0) {
38             cin >> px1 >> px2 >> py1 >> py2;
39             for(int i=px1; i<=px2; i++)
40                 for(int j=py1; j<=py2; j++)
41                     for(int ai=0; ai<=6; ai++)
42                         for(int bi=0; bi<=6; bi++)
43                             V[i][j][ai][bi] = true;
44         }
45
46         bool found = false;
47         queue<Step> Q;
48         Q.push(Step(x1, y1, 0, 0, 0));
49
50         while(!Q.empty()) {
51             Step it = Q.front(); Q.pop();
52             if (!it.valid()) continue;
53             it.mark();
54
55             if (it.x == x2 && it.y == y2) {
56                 cout << "Optimal solution takes " << it.v << " hops." << endl;
57                 found = true;
58                 break;
59             }
60
61             for(int ai=-1; ai<=1; ai++)
62                 for(int bi=-1; bi<=1; bi++)
63                     Q.push(it.go(ai, bi));
64         }
65         if (!found) cout << "No solution." << endl;
66     }
```

graphs/314.cpp

```

1  #include <iostream>
2  #include <iomanip>
3  #include <cstring>
4  #include <string>
5  #include <cmath>
6  #include <climits>
7  #include <vector>
8  #define MAX 70
9  using namespace std;
10
11 int G[MAX][MAX], n, m, sx, sy, tx, ty;
12 bool V[MAX][MAX][4];
13 string dir;
14
15 struct Step {
16     int x, y, d, v, p;
17     Step() {}
18     Step(int x, int y, int d, int v, int p) : x(x), y(y), d(d), v(v), p(p) {}
19     Step left(int pp) {
20         return Step(x, y, (d+3)%4, v+1, pp);
21     }
22     Step right(int pp) {
23         return Step(x, y, (d+1)%4, v+1, pp);
24     }
25     bool canGo(int i) {
26         return (d==0 && x-i>=1 && !G[x-i][y]) ||
27                (d==1 && y+i<m-1 && !G[x][y+i]) ||
28                (d==2 && x+i<n-1 && !G[x+i][y]) ||
29                (d==3 && y-i>=1 && !G[x][y-i]);
30     }
31     Step go(int pp, int i) {
32         if (d==0) return Step(x-i, y, d, v+1, pp);
33         if (d==1) return Step(x, y+i, d, v+1, pp);
34         if (d==2) return Step(x+i, y, d, v+1, pp);
35         if (d==3) return Step(x, y-i, d, v+1, pp);
36     }
37 };
38
39
40 int main() {
41     while(cin >> n >> m, n|m) {
42         vector<Step> Q;
43
44         memset(G, 0, sizeof(G));
45         memset(V, 0, sizeof(V));
46
47         for(int i=0;i<n;i++)
48             for (int j=0;j<m;j++)
49                 cin >> G[i][j];
50
51         n++; m++;
52         for(int i=n-1;i>=0;i--)
53             for (int j=m-1;j>=0;j--)
54                 if (G[i][j])
55                     G[i+1][j] = G[i][j+1] = G[i+1][j+1] = 1;
56
57         cin >> sx >> sy >> tx >> ty >> dir;
58         if (dir=="north") Q.push_back(Step(sx, sy, 0, 0, -1));
59         if (dir=="east") Q.push_back(Step(sx, sy, 1, 0, -1));
60         if (dir=="south") Q.push_back(Step(sx, sy, 2, 0, -1));
61         if (dir=="west") Q.push_back(Step(sx, sy, 3, 0, -1));
62
63         int ptr = 0;
64         while(ptr < Q.size()) {

```

```

65         Step it = Q[ptr];
66         if (it.x == tx && it.y == ty) {
67             cout << it.v << endl;
68             break;
69         }
70
71         if (V[it.x][it.y][it.d]) { ptr++; continue; }
72         V[it.x][it.y][it.d] = true;
73
74         Q.push_back(it.left(ptr));
75         Q.push_back(it.right(ptr));
76         for (int i=1; i<=3 && it.canGo(i); i++)
77             Q.push_back(it.go(ptr, i));
78
79         ptr++;
80     }
81     if (ptr == Q.size()) cout << -1 << endl;
82 }
83 }

```

graphs/315.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <string>
4  #include <sstream>
5  #define MAX 101
6  using namespace std;
7  int G[MAX][MAX], V[MAX], L[MAX], P[MAX], n, gpe;
8
9  void dfs(int u, int v) {
10     V[v] = L[v] = ++gpe;
11     for(int i = 1; i <= n; i++) {
12         if(G[v][i]) {
13             if(!V[i]){
14                 dfs(v, i);
15                 L[v] = min(L[v], L[i]);
16                 if(L[i] >= V[v]) P[v]++;
17             } else if(i != u) {
18                 L[v] = min(L[v], V[i]);
19             }
20         }
21     }
22 }
23
24 int main() {
25     while(cin >> n, n) {
26         memset(G, 0, sizeof(G));
27         memset(V, 0, sizeof(V));
28         memset(L, 0, sizeof(L));
29         memset(P, 0, sizeof(P));
30         gpe = 0;
31
32         int a, b; string s;
33         while(getline(cin, s), s != "0") {
34             stringstream sin(s);
35             sin >> a;
36             while(sin >> b) {
37                 G[a][b] = G[b][a] = 1;
38             }
39         }
40         dfs(1, 1); P[1]--;
41         int cnt = 0;
42         for(int i=1; i<=n; i++)
43             if (P[i]) cnt++;
44
45         cout << cnt << endl;
46     }

```

graphs/321.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #define MAX 15
6  using namespace std;
7
8
9
10 int G[MAX][MAX], C[MAX][MAX], n, m1, m2;
11 bool V[MAX][1200];
12 string dir;
13 struct Step {
14     int x, s, v, p;
15     int type, room;
16     Step() {}
17     Step(int x, int s, int v, int p) : x(x), s(s), v(v), p(p) {}
18     Step(int x, int s, int v, int p, int type, int room) : type(type), room(room), x(x), s(s), v(v)
19
20     Step change(int pp, int i) {
21         return Step(x, s ^ (1<<i), v+1, pp, (s & (1<<i))?2:1, i);
22     }
23     Step move(int pp, int i) {
24         return Step(i, s, v+1, pp, 3, i);
25     }
26 };
27
28 vector<Step> Q;
29
30 void print(Step step) {
31     if (step.p == -1) return;
32     print(Q[step.p]);
33     if (step.type == 1)
34         cout << "- Switch on light in room " << step.room+1 << "." << endl;
35     if (step.type == 2)
36         cout << "- Switch off light in room " << step.room+1 << "." << endl;
37     if (step.type == 3)
38         cout << "- Move to room " << step.room+1 << "." << endl;
39 }
40
41
42
43 int main() {
44     int tt=0;
45     while(cin >> n >> m1 >> m2, n|m1|m2) {
46         Q = vector<Step>();
47
48         memset(G, 0, sizeof(G));
49         memset(C, 0, sizeof(C));
50         memset(V, 0, sizeof(V));
51
52         int a, b;
53         for(int i=0;i<m1; i++) {
54             cin >> a >> b;
55             a--; b--;
56             G[a][b] = G[b][a] = 1;
57         }
58         for(int i=0;i<m2; i++) {
59             cin >> a >> b;
60             a--;b--;
61             C[a][b] = 1;
62         }
63
64         Q.push_back(Step(0, 1, 0, -1));

```

```

65
66     int ptr = 0;
67     cout << "Villa #" << ++tt << endl;
68     while(ptr < Q.size()) {
69         Step it = Q[ptr];
70         if (it.x == n-1 && it.s == (1<<(n-1))) {
71             cout << "The problem can be solved in " << it.v << " steps:" << endl;
72             print(it);
73             break;
74         }
75
76         if (V[it.x][it.s]) { ptr++; continue; }
77         V[it.x][it.s] = true;
78
79         for(int i=0; i<n; i++) {
80             if (G[it.x][i] && (it.s & (1<<i))) Q.push_back(it.move(ptr, i));
81             if (C[it.x][i] && it.x != i) Q.push_back(it.change(ptr, i));
82         }
83
84         ptr++;
85     }
86     if (ptr == Q.size()) cout << "The problem cannot be solved." << endl;
87     cout << endl;
88 }
89 }

```

graphs/627.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #define MAX 400
6  using namespace std;
7
8  int G[MAX][MAX], n, m;
9  bool V[MAX];
10
11
12  struct Step {
13      int x, v, p;
14      Step() {}
15      Step(int x, int v, int p) : x(x), v(v), p(p) {}
16  };
17
18  vector<Step> Q;
19
20  void print(Step step, bool first) {
21      if (step.p != -1) print(Q[step.p], false);
22      cout << step.x << (first?" ":" ");
23  }
24
25  int main() {
26      while(cin >> n) {
27          cout << "-----" << endl;
28          memset(G, 0, sizeof(G));
29
30          int a, b;
31          for(int i=0; i<n;i++) {
32              cin >> a;
33              while(cin.get()!='\n') {
34                  if (cin.peek() == '\n') break;
35                  cin >> b;
36                  G[a][b] = true;
37              }
38          }
39
40          cin >> m;

```

```

41     for(int i=0;i<m;i++) {
42         memset(V, 0, sizeof(V));
43         cin >> a >> b;
44         Q = vector<Step>();
45         Q.push_back(Step(a, 0, -1));
46
47         int ptr = 0;
48         while(ptr < Q.size()) {
49             Step it = Q[ptr];
50             if (it.x == b) {
51                 print(it, true);
52                 cout << endl;
53                 break;
54             }
55
56             if (V[it.x]) { ptr++; continue; }
57             V[it.x] = true;
58
59             for(int i=1; i<=n; i++)
60                 if (G[it.x][i]) Q.push_back(Step(i,it.v+1,ptr));
61
62             ptr++;
63         }
64         if (ptr == Q.size()) cout << "connection impossible" << endl;
65     }
66 }
67 }

```

graphs/796.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <string>
4  #include <sstream>
5  #include <vector>
6  #include <algorithm>
7  #define MAX 101
8  using namespace std;
9  int G[MAX][MAX], V[MAX], L[MAX], n, gpe;
10
11 struct Ponte {
12     int a, b;
13     Ponte() { }
14     Ponte(int a, int b) : a(min(a, b)), b(max(a, b)) {}
15 };
16 bool comp(const Ponte& a, const Ponte& b) { return a.a < b.a || (a.a==b.a && a.b < b.b); }
17 vector<Ponte> P;
18
19 void dfs(int u, int v) {
20     V[v] = L[v] = ++gpe;
21     for(int i = 0; i < n; i++) {
22         if(G[v][i]) {
23             if(!V[i]){
24                 dfs(v, i);
25                 L[v] = min(L[v], L[i]);
26                 if(L[i] == V[i]) P.push_back(Ponte(v, i));
27             } else if(i != u) {
28                 L[v] = min(L[v], V[i]);
29             }
30         }
31     }
32 }
33
34 int main() {
35     while(cin >> n) {
36         memset(G, 0, sizeof(G));
37         memset(V, 0, sizeof(V));
38         memset(L, 0, sizeof(L));

```

```

39     P.clear();
40     gpe = 0;
41
42     int a, an, b; char skip;
43     for(int i=0; i<n; i++) {
44         cin >> a >> skip >> an >> skip;
45         while(an--) {
46             cin >> b; G[b][a] = G[a][b] = 1;
47         }
48     }
49
50     for(int i=0; i<n; i++)
51         if (!V[i])
52             dfs(i, i);
53
54     cout << P.size() << " critical links" << endl;
55     sort(P.begin(), P.end(), comp);
56
57     for(int i=0; i<P.size(); i++) {
58         cout << P[i].a << " - " << P[i].b << endl;
59     }
60     cout << endl;
61 }
62 }

```

graphs/820.cpp

```

1  #include <iostream>
2  #include <iomanip>
3  #include <cstring>
4  #include <string>
5  #include <climits>
6  #include <cmath>
7  #define MAX 1006
8  using namespace std;
9
10 int G[MAX][MAX], n;
11 int F[MAX][MAX];
12 bool V[MAX];
13
14 int send(int s, int t, int minn) {
15     V[s] = true;
16
17     if (s==t) return minn;
18
19     for(int i=1; i<=n; i++) {
20         int capacity = G[s][i]-F[s][i];
21         if (!V[i] && G[s][i]-F[s][i] > 0) {
22             if (int sent = send(i, t, min(minn, capacity))) {
23                 F[s][i] += sent;
24                 F[i][s] -= sent;
25                 return sent;
26             }
27         }
28     }
29
30     return 0;
31 }
32
33 int main() {
34     int tt=0;
35     while(cin >> n, n) {
36         memset(G, 0, sizeof(G));
37         memset(F, 0, sizeof(F));
38         memset(V, 0, sizeof(V));
39
40         tt++;
41         int s, t, c;

```

```

42     cin >> s >> t >> c;
43     for(int i=0;i<c;i++) {
44         int a, b, f;
45         cin >> a >> b >> f;
46         G[a][b] = G[b][a] += f;
47     }
48
49     int total = 0;
50     while(int sent = send(s, t, INT_MAX)) {
51         total += sent;
52         memset(V, 0, sizeof(V));
53     }
54
55     cout << "Network " << tt << endl;
56     cout << "The bandwidth is " << total << "." << endl;
57     cout << endl;
58 }
59
60
61
62 }

```

graphs/908.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #define MAX 200010
8
9  struct Road {
10     int v, c;
11     Road(int v, int c) : v(v), c(c) {}
12     bool operator < (const Road& that) const { return c > that.c; }
13 };
14
15 using namespace std;
16
17 vector<Road> G[MAX];
18
19 int main() {
20     int n, k, m, t=0;
21     while(cin >> n) {
22         memset(G, 0, sizeof(G));
23
24         int before = 0;
25         for(int i=0; i<n-1; i++) {
26             int a, b, c;
27             cin >> a >> b >> c;
28             G[a].push_back(Road(b, c));
29             G[b].push_back(Road(a, c));
30             before += c;
31         }
32
33         int total=0;
34         cin >> k;
35         for(int i=0; i<k; i++) {
36             int a, b, c;
37             cin >> a >> b >> c;
38
39             int maxx = 0, maxv, side, counter;
40             for(int j=0; j<G[a].size(); j++)
41                 if (G[a][j].c > maxx) {
42                     maxx = G[a][j].c;
43                     maxv = j;
44                     side= a; counter = b;

```



```

45         }
46
47         for(int j=0; j<G[b].size(); j++)
48             if (G[b][j].c > maxx) {
49                 maxx = G[b][j].c;
50                 maxv = j;
51                 side= b; counter = a;
52             }
53
54             if (maxx <= c) continue;
55             total = maxx-c;
56             G[side][maxv].v = counter;
57             G[side][maxv].c = c;
58         }
59
60         cin >> m;
61         for(int i=0; i<m; i++) {
62             int a, b, c;
63             cin >> a >> b >> c;
64         }
65
66         if (t++) cout << endl;
67         cout << before << endl << before-total << endl;
68     }
69     return 0;
70 }

```

graphs/1056.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <string>
4  #include <map>
5  #include <cassert>
6  #define MAX 51
7
8  using namespace std;
9
10 int G[MAX][MAX];
11 int n, m;
12 map<string, int> M;
13
14 int person(string& s) {
15     if (M.find(s) != M.end())
16         return M[s];
17     else
18         return M[s]=M.size();
19 }
20
21
22 int main() {
23     int t=0;
24     while(cin >> n >> m, n|m) {
25         memset(G, 0x1f, sizeof(G));
26         M.clear();
27
28         for(int i=0; i<m; i++) {
29             string p, q;
30             cin >> p >> q;
31             int a = person(p), b=person(q);
32             G[a][b] = G[b][a] = 1;
33         }
34         for(int i=1; i<=n; i++) G[i][i] = 0;
35
36         assert(M.size() <= n);
37
38         for(int k=1; k<=n; k++)
39             for(int i=1; i<=n; i++)

```

```

40         for(int j=1; j<=n; j++)
41             G[i][j] = min(G[i][j], G[i][k] + G[k][j]);
42
43     int maxx = 0;
44     for(int i=1; i<=n; i++)
45         for(int j=1; j<=n; j++)
46             maxx = max(maxx, G[i][j]);
47
48     cout << "Network " << ++t << ": ";
49
50     if (maxx <= n)
51         cout << maxx << endl;
52     else
53         cout << "DISCONNECTED" << endl;
54
55     cout << endl;
56
57 }
58 return 0;
59 }
60

```

graphs/1174.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <string>
5  #include <vector>
6  #include <algorithm>
7  #include <queue>
8  #include <map>
9  #define MAX 200010
10
11 using namespace std;
12
13 struct Road {
14     int v, c;
15     Road(int v, int c) : v(v), c(c) {}
16     inline bool operator < (const Road& that) const { return c > that.c; }
17 };
18
19 vector<Road> G[MAX];
20 priority_queue<Road> Q;
21 int n, m;
22 bool V[MAX];
23 map<string, int> M;
24
25 int city(string& s) {
26     if (M.find(s) != M.end())
27         return M[s];
28     else
29         return M[s]=M.size();
30 }
31
32
33 int main() {
34     int t; cin >> t; t=0;
35
36     while(cin >> n >> m) {
37         memset(V, 0, sizeof(V));
38         memset(G, 0, sizeof(G));
39         M.clear();
40         Q = priority_queue<Road>();
41
42         for(int i=0; i<m; i++) {
43             string p, q; int a, b, c;
44             cin >> p >> q >> c;

```

```

45         a = city(p); b=city(q);
46         G[a].push_back(Road(b, c));
47         G[b].push_back(Road(a, c));
48     }
49
50     int total = 0, totalc=0;
51
52     Q.push(Road(1, 0));
53
54     while(totalc < n) {
55         Road item = Q.top(); Q.pop();
56         if (V[item.v]) continue;
57
58         V[item.v] = true;
59         total += item.c;
60         totalc++;
61
62         for(int j=0; j<G[item.v].size(); j++)
63             if (!V[G[item.v][j].v])
64                 Q.push(G[item.v][j]);
65     }
66
67     if (t++) cout << endl;
68     cout << total << endl;
69 }
70 return 0;
71 }

```

graphs/1197.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  using namespace std;
6
7  vector<int> G[501], P[30001];
8  bool VG[501], VP[30001];
9  int n, m;
10
11 int dfs(int v) {
12     int sum = 1;
13     VP[v] = true;
14
15     for(int i=0; i<P[v].size(); i++) {
16         int g = P[v][i];
17         if (VG[g]) continue;
18         VG[g] = true;
19
20         for(int j=0; j<G[g].size(); j++) {
21             int u = G[g][j];
22             if (VP[u]) continue;
23             sum += dfs(u);
24         }
25     }
26
27     return sum;
28 }
29
30 int main() {
31     while(cin >> n >> m, n|m) {
32         memset(G, 0, sizeof(G));
33         memset(P, 0, sizeof(P));
34         memset(VG, 0, sizeof(VG));
35         memset(VP, 0, sizeof(VP));
36
37         for(int i=0; i<m; i++) {
38             int k; cin >> k;

```

```

39         while(k--) {
40             int a; cin >> a;
41             G[i].push_back(a);
42             P[a].push_back(i);
43         }
44     }
45
46     cout << dfs(0) << endl;
47 }
48 return 0;
49 }

```

graphs/1208.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #define MAX 501
7
8  using namespace std;
9
10 int G[MAX][MAX], n;
11 bool V[MAX];
12 int D[MAX], DO[MAX];
13
14 struct Item {
15     int p, a, b;
16     Item(){}
17     Item(int p, int a, int b) : p(p), a(min(a,b)), b(max(a,b)) {}
18 };
19
20 bool comp(const Item& a, const Item& b) {
21     if (a.p != b.p) return a.p < b.p;
22     if (a.a != b.a) return a.a < b.a;
23     if (a.b != b.b) return a.b < b.b;
24     return false;
25 }
26
27 vector<Item> R;
28
29 int updateD(int i) {
30     D[i] = 0;
31     for(int j=0; j<n; j++) {
32         if (G[i][j] && G[i][j] < D[j]) {
33             D[j] = G[i][j];
34             DO[j] = i;
35         }
36     }
37 }
38
39 int main() {
40     int t; char skip;
41     cin >> t;
42     t = 0;
43     while(cin >> n) {
44         memset(V, 0, sizeof(V));
45         memset(D, 0x3F, sizeof(D));
46         R.clear();
47
48         for(int i=0; i<n; i++) {
49             for(int j=0; j<n; j++) {
50                 cin >> G[i][j];
51                 if (j+1<n) cin >> skip;
52             }
53         }
54

```

```

55     int total = 0;
56
57     V[0] = true;
58     updatedD(0);
59
60     for(int k=1; k<n; k++) {
61         int minn=INT_MAX, minv;
62         for(int i=0; i<n; i++) {
63             if (!V[i] && D[i] < minn) {
64                 minn = D[i];
65                 minv = i;
66             }
67         }
68         R.push_back(Item(minn, DO[minv], minv));
69         V[minv] = true;
70         updatedD(minv);
71         total += minn;
72     }
73
74     sort(R.begin(), R.end(), comp);
75     cout << "Case " << ++t << ":" << endl;
76     for(int i=0; i<R.size(); i++) {
77         cout << (char)(R[i].a+'A') << "-" << (char)(R[i].b+'A') << " " << R[i].p << endl;
78     }
79 }
80 return 0;
81 }

```

graphs/1216.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <cmath>
4  #define MAX 1000
5  using namespace std;
6
7  double G[MAX][MAX];
8  int X[MAX], Y[MAX], n, k;
9  int V[MAX];
10
11 void dfs(int v, int comp, int max) {
12     V[v] = comp;
13     for(int i=0; i<n; i++) {
14         if (!V[i] && G[v][i] <= max)
15             dfs(i, comp, max);
16     }
17 }
18
19 int main() {
20     int t; cin >> t; t=0;
21     while(cin >> k) {
22         n=0;
23
24         double maxd=0;
25         while(cin >> X[n], X[n]!=-1) {
26             cin >> Y[n];
27             for(int i=0; i<n; i++) {
28                 G[i][n] = G[n][i] = sqrt(pow(X[n]-X[i], 2.0)+pow(Y[n]-Y[i], 2.0));
29                 maxd = max(maxd, G[i][n]);
30             }
31             n++;
32         }
33
34         int begin=0, end=(int)ceil(maxd);
35         int best, last = -1;
36         while(begin <= end) {
37             int mid = (begin+end)/2;
38             if (mid == last) break;

```

```

39
40     int comp=0;
41     memset(V, 0, sizeof(V));
42     for(int i=0; i<n; i++)
43         if (!V[i])
44             dfs(i, ++comp, mid);
45
46     last = mid;
47     if (comp > k)
48         begin = mid;
49     else {
50         if (comp == k) best = mid;
51         end = mid;
52     }
53 }
54
55     cout << best << endl;
56 }
57 }

```

graphs/1220.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <map>
4  #include <cstring>
5  #include <vector>
6  #define MAX 205
7  using namespace std;
8
9  map<string, int> E;
10 int emp(string& s) {
11     if (E.find(s) != E.end())
12         return E[s];
13     else
14         return E[s] = E.size()-1;
15 }
16
17 bool L[MAX], L2[MAX];
18 vector<int> G[MAX];
19 int n;
20
21 int dfs(int v) {
22     int acum = 0, illu = 0;
23     for(int i=0; i<G[v].size(); i++) {
24         acum += dfs(G[v][i]);
25         if (L[G[v][i]]) illu++;
26     }
27     if (G[v].size() > 0 && illu < G[v].size())
28         L[v] = true;
29     return acum + L[v];
30 }
31
32 int main()
33 {
34     while(cin >> n, n) {
35         memset(G, 0, sizeof(G));
36         memset(L, 0, sizeof(L));
37         E.clear();
38         string a, b;
39         cin >> a;
40         emp(a);
41         for(int i=1; i<n; i++) {
42             cin >> a >> b;
43             G[emp(b)].push_back(emp(a));
44         }
45
46         int total = dfs(0);

```

```

47 |
48 |     memcpy(L2, L, sizeof(L));
49 |     bool unique = true;
50 |     for(int i=0; i<n; i++) {
51 |         if (!L2[i]) {
52 |             memset(L, 0, sizeof(L));
53 |             L[i] = true;
54 |             if (dfs(0) == total) {
55 |                 unique = false;
56 |                 break;
57 |             }
58 |         }
59 |     }
60 |
61 |     cout << n-total << " " << (unique?"Yes":"No") << endl;
62 | }
63 | return 0;
64 | }

```

graphs/1229.cpp

```

1 | #include <iostream>
2 | #include <map>
3 | #include <string>
4 | #include <cstring>
5 | #include <sstream>
6 | #include <set>
7 | #include <algorithm>
8 | #define MAX 101
9 | using namespace std;
10 |
11 | map<string, int> P;
12 | int word(const string& p) {
13 |     if (P.find(p) != P.end())
14 |         return P[p];
15 |     else
16 |         return P[p] = P.size();
17 | }
18 |
19 | int O[MAX], npv, CO[MAX], GR[MAX];
20 | string W[MAX];
21 | bool G[MAX][MAX], V[MAX];
22 | int n;
23 | set<int> words;
24 | set<string> answer;
25 |
26 | void DFS(int v){
27 |     V[v] = true;
28 |     for(int i = 1; i <= n; i++)
29 |         if (G[v][i] && !V[i])
30 |             DFS(i);
31 |     O[npv++] = v;
32 | }
33 |
34 | int DFSt(int v, int comp){
35 |     int acum = 1;
36 |     V[v] = true; CO[v] = comp;
37 |     for(int i = 1; i <= n; i++)
38 |         if (G[i][v] && !V[i])
39 |             acum += DFSt(i, comp);
40 |     return acum;
41 | }
42 |
43 | void DFSf(int v){
44 |     V[v] = true;
45 |     answer.insert(W[v]);
46 |     for(int i = 1; i <= n; i++)
47 |         if (G[v][i] && !V[i])

```

```

48         DFSf(i);
49     }
50
51     int main() {
52         string s, p, q;
53         while(cin >> n, n) {
54             memset(G, 0, sizeof(G));
55             memset(CO, 0, sizeof(CO));
56             memset(GR, 0, sizeof(GR));
57             P.clear();
58             words.clear();
59             answer.clear();
60             getline(cin, p);
61
62             for(int i=0;i<n; i++) {
63                 getline(cin, s);
64                 stringstream sin(s);
65                 sin >> p;
66                 while(sin >> q) {
67                     G[word(p)][word(q)] = true;
68                     GR[word(p)]++;
69                 }
70                 W[word(p)] = p;
71             }
72
73             npv = 1;
74             memset(V, 0, sizeof(V));
75             memset(O, 0, sizeof(O));
76
77             for(int i = 1; i <= n; i++)
78                 if(!V[i]) DFS(i);
79
80             memset(V, 0, sizeof(V));
81
82             int comp = 0;
83             for(int i = n; i > 0; i--) {
84                 if(!V[O[i]]) {
85                     comp++;
86                     if (DFSf(O[i], comp) > 1 || GR[O[i]] == 0) {
87                         for(int j=1;j<=n;j++) {
88                             if (CO[j] == comp) {
89                                 words.insert(j);
90                             }
91                         }
92                     }
93                 }
94             }
95
96             memset(V, 0, sizeof(V));
97             for(set<int>::iterator it=words.begin(); it!=words.end(); it++) {
98                 DFSf(*it);
99             }
100
101             cout << answer.size() << endl;
102             for(set<string>::iterator it=answer.begin(); it!=answer.end(); it++)
103                 cout << (it!=answer.begin()?" ":"") << *it;
104             cout << endl;
105         }
106
107         return 0;
108     }

```

graphs/1234.cpp

```

1 | #include <iostream>
2 | #include <cstring>
3 | #include <climits>
4 | #include <vector>

```



```

5  #include <algorithm>
6  #include <queue>
7  #define MAX 10005
8
9  using namespace std;
10
11 struct Road {
12     int v, c;
13     Road(int v, int c) : v(v), c(c) {}
14     inline bool operator < (const Road& that) const { return c < that.c; }
15 };
16
17 vector<Road> G[MAX];
18 int CStart[MAX], CCount[MAX], nc;
19 priority_queue<Road> Q;
20 vector<int> R;
21 int n, m;
22 bool V[MAX];
23
24 int dfs(int v) {
25     V[v] = true;
26     int acum = 1;
27     for(int i=0; i<G[v].size(); i++)
28         if (!V[G[v][i].v])
29             acum += dfs(G[v][i].v);
30     return acum;
31 }
32
33 int main() {
34     int t; cin >> t;
35     while(cin >> n >> m, t--) {
36         memset(V, 0, sizeof(V));
37         memset(G, 0, sizeof(G));
38         nc = 0;
39         R.clear();
40
41         for(int i=0; i<m; i++) {
42             int a, b, c;
43             cin >> a >> b >> c;
44             G[a].push_back(Road(b, c));
45             G[b].push_back(Road(a, c));
46         }
47
48         for(int i=1; i<=n; i++) {
49             if (!V[i]) {
50                 CStart[nc]=i;
51                 CCount[nc]=dfs(i);
52                 nc++;
53             }
54         }
55
56         int result=0;
57         for(int i=0; i<nc; i++) {
58             int totalc=0;
59             Q.push(Road(CStart[i], 0));
60             memset(V, 0, sizeof(V));
61
62             while(totalc < CCount[i]) {
63                 Road item = Q.top(); Q.pop();
64                 if (V[item.v]) { result+=item.c; continue; }
65
66                 V[item.v] = true;
67                 totalc++;
68
69                 for(int j=0; j<G[item.v].size(); j++)
70                     if (!V[G[item.v][j].v])
71                         Q.push(G[item.v][j]);
72             }
73             while(!Q.empty()) {
74                 result += Q.top().c;

```

```

75         Q.pop();
76     }
77 }
78     cout << result << endl;
79 }
80     return 0;
81 }

```

graphs/1235.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #define MAX 501
5
6  using namespace std;
7
8  int abs(int a) {
9      return a>0?a:-a;
10 }
11
12 int d(int a, int b) {
13     int result = 0;
14     for(int i=0; i<4; i++) {
15         int aa=a%10, bb=b%10;
16         result += min(abs(aa-bb), 10-abs(aa-bb));
17         a/=10; b/=10;
18     }
19     return result;
20 }
21
22 int K[MAX], G[MAX][MAX], n;
23 bool V[MAX];
24 int D[MAX];
25
26 int updateD(int i) {
27     D[i] = 0;
28     for(int j=0; j<n; j++) {
29         if (G[i][j]) D[j] = min(D[j], G[i][j]);
30     }
31 }
32
33 int main()
34 {
35     int t;
36     cin >> t;
37     while(cin >> n) {
38         memset(V, 0, sizeof(V));
39         memset(D, 0x3F, sizeof(D));
40
41         for(int i=0; i<n; i++)
42             cin >> K[i];
43
44         for(int i=0; i<n; i++)
45             for(int j=i+1; j<n; j++)
46                 G[i][j] = G[j][i] = d(K[i],K[j]);
47
48         int total=INT_MAX;
49         for(int i=0;i<n;i++) total = min(total, d(0, K[i]));
50
51         V[0] = true;
52         updateD(0);
53
54         for(int k=1; k<n; k++) {
55             int minn=INT_MAX, minv;
56             for(int i=0; i<n; i++) {
57                 if (!V[i] && D[i] < minn) {
58                     minn = D[i];

```

```

59         minv = i;
60     }
61 }
62 V[minv] = true;
63 updateD(minv);
64 total += minn;
65 }
66
67     cout << total << endl;
68 }
69 return 0;
70 }

```

graphs/1247.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #define MAX 30
8
9  using namespace std;
10
11 struct Edge {
12     int u, v, c;
13     Edge(int u, int v, int c) : u(u), v(v), c(c) {}
14     inline bool operator < (const Edge& that) const { return c > that.c; }
15 };
16
17 int G[MAX][MAX];
18 int V[MAX];
19 int D[MAX];
20 int n, m;
21
22 void show(int t) {
23     if (D[t] != t) {
24         show(D[t]);
25         cout << " ";
26     }
27     cout << (char)(t+'A');
28 }
29
30
31 int shortest(int a, int b) {
32     memset(V, 0x3f, sizeof(V));
33     priority_queue<Edge> Q;
34     Q.push(Edge(a, a, 0));
35
36     while(!Q.empty()) {
37         Edge item = Q.top(); Q.pop();
38         if (item.c >= V[item.v]) continue;
39         V[item.v] = item.c;
40         D[item.v] = item.u;
41
42         for(int j=0; j<n; j++) {
43             if (G[item.v][j]) {
44                 Edge e = Edge(item.v, j, item.c+G[item.v][j]);
45                 if (e.c <= V[e.v])
46                     Q.push(e);
47             }
48         }
49     }
50     show(b); cout << endl;
51 }
52
53 int main() {

```

```

54     while(cin >> n >> m) {
55         memset(G, 0, sizeof(G));
56
57         for(int i=0; i<m; i++) {
58             char a, b; int c;
59             cin >> a >> b >> c;
60             G[a-'A'][b-'A'] = G[b-'A'][a-'A'] = c;
61         }
62
63         int k; cin >> k;
64         while(k--) {
65             char a, b; cin >> a >> b;
66             shortest(a-'A', b-'A');
67         }
68     }
69     return 0;
70 }

```

graphs/1263.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #define MAX 2001
4  using namespace std;
5
6  int n;
7  bool V[MAX], G[MAX][MAX];
8  int X[MAX], Y[MAX], D[MAX], O[MAX], npv;
9
10 inline int abs(int a) {
11     return a>0?a:-a;
12 }
13
14 void dfs(int v, bool sort){
15     V[v] = true;
16     for(int i = 1; i <= n; i++)
17         if (G[v][i] && !V[i])
18             dfs(i, sort);
19     if (sort)
20         O[++npv] = v;
21 }
22
23 int main() {
24     int t; cin >> t; t=0;
25     while(cin >> n) {
26         memset(G, 0, sizeof(G));
27         for(int i=1; i<=n; i++) {
28             cin >> X[i] >> Y[i] >> D[i];
29         }
30         for(int i=1; i<=n; i++) {
31             for(int j=1; j<=n; j++) {
32                 int r = D[i]/2;
33                 if (abs(X[j]-X[i])<=r && abs(Y[j]-Y[i]) <=r && i!=j)
34                     G[i][j] = true;
35             }
36         }
37
38         npv = 0;
39         memset(V, 0, sizeof(V));
40         memset(O, 0, sizeof(O));
41
42         for(int i = 1; i <= n; i++)
43             if(!V[i]) dfs(i, true);
44
45         memset(V, 0, sizeof(V));
46
47         int comp = 0;
48         for(int i = n; i > 0; i--)

```

```

49         if(!V[O[i]]) {
50             comp++;
51             dfs(O[i], false);
52         }
53
54         cout << comp << endl;
55     }
56
57     return 0;
58 }

```

graphs/10044.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <string>
5  #include <vector>
6  #include <queue>
7  #include <map>
8  #define MAX 5000
9  using namespace std;
10
11 vector<int> G[MAX];
12 int n, m;
13 bool V[MAX];
14 map<string, int> A;
15
16 struct Step {
17     int x, v;
18     Step() {}
19     Step(int x, int v) : x(x), v(v) {}
20 };
21
22 queue<Step> Q;
23
24 int author(const string& a) {
25     if (A.find(a) != A.end())
26         return A[a];
27     else
28         return A[a] = A.size()-1;
29 }
30
31
32 char C[MAX];
33 void parseAuthors(const string& s) {
34     vector<int> TA;
35     int commas = 0, chars=0;
36     for(int i=0;i<s.size();i++) {
37         char c = s[i];
38         if (chars == 0 && c == ' ') continue;
39
40         if ((c==',' || c==':') && ++commas == 2) {
41             TA.push_back(author(string(C, chars)));
42             chars = commas = 0;
43         } else {
44             C[chars++] = c;
45         }
46     }
47     for(int i=0;i<TA.size(); i++) {
48         for(int j=i+1;j<TA.size(); j++) {
49             G[TA[i]].push_back(TA[j]);
50             G[TA[j]].push_back(TA[i]);
51         }
52     }
53
54 }
55

```

```

56 int main() {
57     string s;
58     int t=0, tt;
59     cin >> tt;
60     while(t++ < tt) {
61         cin >> n >> m;
62         memset(G, 0, sizeof(G));
63         A.clear();
64         getline(cin, s);
65         while(n-- > 0) {
66             getline(cin, s);
67             parseAuthors(s);
68         }
69
70         cout << "Scenario " << t << endl;
71         for(int i=0; i<m; i++) {
72             bool stop;
73             memset(V, 0, sizeof(V));
74             getline(cin, s);
75             int b = author(s);
76             Q = queue<Step>();
77             Q.push(Step(author("Erdos, P."), 0));
78             bool found = false;
79
80             while(!Q.empty()) {
81                 Step it = Q.front(); Q.pop();
82                 if (it.x == b) {
83                     cout << s << " " << it.v << endl;
84                     found = true;
85                     break;
86                 }
87
88                 V[it.x] = true;
89
90                 for(int i=0; i<G[it.x].size(); i++)
91                     if (!V[G[it.x][i]]) Q.push(Step(G[it.x][i], it.v+1));
92             }
93             if (!found) cout << s << " infinity" << endl;
94         }
95     }
96     return 0;
97 }

```

graphs/10092.cpp

```

1  #include <iostream>
2  #include <iomanip>
3  #include <cstring>
4  #include <string>
5  #include <cmath>
6  #include <climits>
7  #define MAX 1100
8  using namespace std;
9
10 int G[MAX][MAX], nk, np, n;
11 bool V[MAX];
12
13 int SOURCE() { return 1; }
14 int P(int i) { return 1+i; }
15 int K(int i) { return 1+np+i; }
16 int TARGET() { return 2+np+nk; }
17
18 int send(int s, int t, int minn) {
19     V[s] = true;
20
21     if (s==t) return minn;
22     for(int i=1; i<=n; i++) {
23         if (!V[i] && G[s][i] > 0) {

```

```

24         if (int sent = send(i, t, min(minn, G[s][i]))) {
25             G[s][i] -= sent;
26             G[i][s] += sent;
27             return sent;
28         }
29     }
30 }
31 return 0;
32 }
33
34 int main() {
35     int tmp, tmp2;
36     while(cin >> nk >> np, nk|np) {
37         n = nk+np+2;
38         int expected = 0;
39         memset(G, 0, sizeof(G));
40         memset(V, 0, sizeof(V));
41
42         for(int i=1; i<=nk; i++) {
43             cin >> tmp;
44             expected += tmp;
45             G[K(i)][TARGET()] = tmp;
46         }
47
48         for(int i=1; i<=np; i++) {
49             cin >> tmp;
50             G[SOURCE()][P(i)] = 1;
51             for(int j=0; j<tmp; j++) {
52                 cin >> tmp2;
53                 G[P(i)][K(tmp2)] = 1;
54             }
55         }
56
57         int total = 0;
58         while(int sent = send(SOURCE(), TARGET(), INT_MAX)) {
59             total += sent;
60             memset(V, 0, sizeof(V));
61         }
62         cout << (expected == total ? 1: 0) << endl;
63         if (expected == total) {
64             for(int i=K(1); i<=K(nk); i++) {
65                 bool printed = false;
66                 for(int j=P(1); j<=P(np); j++) {
67                     if (G[i][j]) {
68                         cout << (printed?" ":"") << (j-1);
69                         printed = true;
70                     }
71                 }
72                 cout << endl;
73             }
74         }
75     }
76 }

```

graphs/10243.cpp

```

1  #include <iostream>
2  #include <iomanip>
3  #include <cstring>
4  #include <queue>
5  #include <cmath>
6  #define MAX 1005
7  using namespace std;
8
9  int n;
10 bool G[MAX][MAX];
11 bool L[MAX];
12 bool V[MAX];

```

```

13
14 void dfs(int v, bool start) {
15     //cout << "*" << v << endl;
16     if (V[v]) return;
17     V[v] = true;
18     bool all = true;
19     int children = 0;
20     for(int i=0;i<n;i++) {
21         if (G[v][i] && !V[i]) {
22             dfs(i, false);
23             all &= L[i];
24             children++;
25         }
26     }
27     if (!all && children > 0 || start && children==0)
28         L[v] = true;
29 }
30
31 int main() {
32     int a, b, m;
33
34     while(cin >> n, n) {
35         memset(G, 0, sizeof(G));
36         memset(L, 0, sizeof(L));
37         memset(V, 0, sizeof(V));
38
39         for(int i=0;i<n;i++) {
40             cin >> m;
41             for(int j=0; j<m; j++) {
42                 cin >> a;
43                 a--;
44                 G[i][a] = G[a][i] = true;
45             }
46         }
47
48         int count = 0;
49         for(int i=0;i<n;i++) {
50             dfs(i, true);
51             if (L[i]) count++;
52         }
53
54         cout << count << endl;
55     }
56
57
58
59 }

```

graphs/10278.cpp

```

1  #include <iostream>
2  #include <iomanip>
3  #include <cstring>
4  #include <string>
5  #include <sstream>
6  #include <cmath>
7  #include <climits>
8  #define MAX 502
9  using namespace std;
10
11 int G[MAX][MAX], f, n;
12 bool F[MAX];
13
14 int main() {
15     int t; cin >> t;
16     string s;
17     while(t--) {
18         cin >> f >> n;

```



```

19     memset(G, 0x3F, sizeof(G));
20     memset(F, 0, sizeof(F));
21
22     for(int i=0; i<f; i++) {
23         int a; cin >> a; F[a] = true;
24     }
25     getline(cin, s);
26     while(getline(cin, s), cin && s!="") {
27         int a, b, c;
28         stringstream inter(s);
29         inter >> a >> b >> c;
30         G[a][b] = G[b][a] = c;
31     }
32
33
34     for(int k=1; k<=n; k++) {
35         G[k][k] = 0;
36         for(int i=1; i<=n; i++)
37             for(int j=1; j<=n; j++)
38                 G[i][j] = min(G[i][j], G[i][k] + G[k][j]);
39     }
40
41     int minn = INT_MAX, minv;
42     for(int i=1; i<=n; i++) {
43         int maxx = 0;
44         for(int j=1; j<=n; j++) {
45             int nearest = INT_MAX;
46             for(int k=1; k<=n; k++) {
47                 if (!F[k] && k!=i) continue;
48                 nearest = min(nearest, G[k][j]);
49             }
50             maxx = max(maxx, nearest);
51         }
52         if (maxx < minn) {
53             minn = maxx;
54             minv = i;
55         }
56     }
57     cout << minv << endl;
58     if (t) cout << endl;
59 }
60 }

```

graphs/10389.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #include <cmath>
8  #include <sstream>
9  #include <string>
10 #include <iomanip>
11 #include <cassert>
12 #define MAX 205
13 #define WALK 1
14 #define METRO 4
15
16 using namespace std;
17
18 struct Edge {
19     int v; double c;
20     Edge(int v, double c) : v(v), c(c) {}
21     inline bool operator < (const Edge& that) const { return c > that.c; }
22 };
23

```

```

24 vector<Edge> G[MAX];
25 double V[MAX];
26 double X[MAX], Y[MAX];
27 int n;
28
29
30 double dist(double ax, double ay, double bx, double by) {
31     return sqrt(pow(ax-bx, 2.0) + pow(ay-by, 2.0))*60/10000;
32 }
33
34 int main() {
35     int t; cin >> t; t=0;
36     while(cin >> X[0] >> Y[0] >> X[1] >> Y[1]) {
37         memset(G, 0, sizeof(G));
38
39         G[0].push_back(Edge(1, dist(X[0], Y[0], X[1], Y[1])));
40         G[1].push_back(Edge(0, dist(X[0], Y[0], X[1], Y[1])));
41
42         n = 2;
43         string s;
44         getline(cin, s);
45         while(getline(cin, s) && s!=" " && s[0]!=' ') {
46             stringstream sin(s);
47             int mn=0;
48             while(sin >> X[n] >> Y[n]) {
49                 if (X[n] == -1 && Y[n] == -1) {
50                     assert(mn >= 2);
51                     mn = 0;
52                     break;
53                 }
54                 if (mn > 0) {
55                     double mDist = dist(X[n-1], Y[n-1], X[n], Y[n])/METRO;
56                     G[n-1].push_back(Edge(n, mDist));
57                     G[n].push_back(Edge(n-1, mDist));
58                 }
59                 for(int i=0; i<n; i++) {
60                     double aDist = dist(X[n], Y[n], X[i], Y[i])/WALK;
61                     G[i].push_back(Edge(n, aDist));
62                     G[n].push_back(Edge(i, aDist));
63                 }
64
65                 n++; mn++;
66             }
67
68         }
69
70         int totalc=0;
71
72         for(int i=0; i<n; i++) V[i] = -1;
73
74         priority_queue<Edge> Q;
75         Q.push(Edge(0, 0));
76
77         while(totalc < n && !Q.empty()) {
78             Edge item = Q.top(); Q.pop();
79             if (item.c >= V[item.v] && V[item.v] >= 0) continue;
80
81             V[item.v] = item.c;
82             totalc++;
83
84             for(int j=0; j<G[item.v].size(); j++) {
85                 Edge e = G[item.v][j];
86                 if (item.c + e.c < V[e.v] || V[e.v] == -1)
87                     Q.push(Edge(e.v, item.c + e.c));
88             }
89         }
90
91         if (t++) cout << endl;
92         cout << (int)round(V[1]) << endl;
93     }

```

```
94 |     return 0;
95 | }
```

graphs/10397.cpp

```
1 | #include <iostream>
2 | #include <cstring>
3 | #include <climits>
4 | #include <vector>
5 | #include <algorithm>
6 | #include <queue>
7 | #include <cmath>
8 | #include <iomanip>
9 | #define MAX 200010
10 |
11 | using namespace std;
12 |
13 | struct Road {
14 |     int v; double c;
15 |     Road(int v, double c) : v(v), c(c) {}
16 |     inline bool operator < (const Road& that) const { return c > that.c; }
17 | };
18 |
19 |
20 | vector<Road> G[MAX];
21 | int X[MAX], Y[MAX];
22 | priority_queue<Road> Q;
23 | int n, m;
24 | bool V[MAX];
25 |
26 |
27 | int main() {
28 |     while(cin >> n) {
29 |         memset(V, 0, sizeof(V));
30 |         memset(G, 0, sizeof(G));
31 |         Q = priority_queue<Road>();
32 |
33 |         for(int i=1; i<=n; i++) {
34 |             int x, y;
35 |             cin >> x >> y;
36 |             X[i] = x; Y[i] = y;
37 |             for(int j=1; j<=i; j++) {
38 |                 double d = sqrt(pow(X[i]-X[j], 2.0)+pow(Y[i]-Y[j], 2.0));
39 |                 G[i].push_back(Road(j, d));
40 |                 G[j].push_back(Road(i, d));
41 |             }
42 |         }
43 |
44 |         cin >> m;
45 |         for(int i=0; i<m; i++) {
46 |             int a, b;
47 |             cin >> a >> b;
48 |             G[a].push_back(Road(b, 0));
49 |             G[b].push_back(Road(a, 0));
50 |         }
51 |
52 |         double total = 0; int totalc=0;
53 |         Q.push(Road(1,0));
54 |
55 |         while(totalc < n && !Q.empty()) {
56 |             Road item = Q.top(); Q.pop();
57 |             if (V[item.v]) continue;
58 |
59 |             V[item.v] = true;
60 |             total += item.c;
61 |             totalc++;
62 |
63 |             for(int j=0; j<G[item.v].size(); j++)
```

```

64         if (!V[G[item.v][j].v])
65             Q.push(G[item.v][j]);
66     }
67
68     cout << setprecision(2);
69     cout << fixed << total << endl;
70 }
71 return 0;
72 }

```

graphs/10480.cpp

```

1  #include <iostream>
2  #include <iomanip>
3  #include <cstring>
4  #include <string>
5  #include <cmath>
6  #include <climits>
7  #define MAX 1006
8  using namespace std;
9
10 int G[MAX][MAX], O[MAX][MAX], n, m;
11 bool V[MAX];
12
13 int send(int s, int t, int minn) {
14     V[s] = true;
15
16     if (s==t) return minn;
17     for(int i=1; i<=n; i++) {
18         if (!V[i] && G[s][i] > 0) {
19             if (int sent = send(i, t, min(minn, G[s][i]))) {
20                 G[s][i] -= sent;
21                 G[i][s] += sent;
22                 return sent;
23             }
24         }
25     }
26     return 0;
27 }
28
29 int main() {
30     int tt=0;
31     while(cin >> n >> m, n|m) {
32         if (tt++) cout << endl;
33
34         memset(G, 0, sizeof(G));
35         memset(V, 0, sizeof(V));
36         memset(O, 0, sizeof(O));
37
38         for(int i=0; i<m; i++) {
39             int a, b, f;
40             cin >> a >> b >> f;
41             G[a][b] = G[b][a] += f;
42             O[a][b] += f;
43         }
44
45         int total = 0;
46         while(int sent = send(1, 2, INT_MAX)) {
47             total += sent;
48             memset(V, 0, sizeof(V));
49         }
50         for(int i=1; i<=n; i++) {
51             for(int j=1; j<=n; j++) {
52                 if (O[i][j] > 0 && V[i] != V[j])
53                     cout << i << " " << j << endl;
54             }
55         }
56     }

```

graphs/10511.cpp

```

1  #include <iostream>
2  #include <iomanip>
3  #include <cstring>
4  #include <sstream>
5  #include <string>
6  #include <cmath>
7  #include <map>
8  #include <climits>
9  #define MAX 1300
10 using namespace std;
11
12 int G[MAX][MAX], n;
13 bool V[MAX];
14 map<string, int> EC, EP, EM;
15
16 int SOURCE() { return 1; }
17 int P(string& s) { if (EP.find(s)!=EP.end()) return EP[s]; else { return EP[s]=++n; } }
18 int M(string& s) { if (EM.find(s)!=EM.end()) return EM[s]; else { return EM[s]=++n; } }
19 int C(string& s) { if (EC.find(s)!=EC.end()) return EC[s]; else { return EC[s]=++n; } }
20 int TARGET() { return 2; }
21
22 int send(int s, int t, int minn) {
23     V[s] = true;
24
25     if (s==t) return minn;
26     for(int i=1; i<=n; i++) {
27         if (!V[i] && G[s][i] > 0) {
28             if (int sent = send(i, t, min(minn, G[s][i]))) {
29                 G[s][i] -= sent;
30                 G[i][s] += sent;
31                 return sent;
32             }
33         }
34     }
35     return 0;
36 }
37
38 int main() {
39     int t; cin >> t;
40     string s, sm, sp, sc;
41     getline(cin, s); getline(cin, s);
42     while(t-->0) {
43         EC.clear(); EP.clear(); EM.clear();
44         memset(G, 0, sizeof(G));
45         memset(V, 0, sizeof(V));
46         n=2;
47
48         while(getline(cin, s) && s!=" " && s!=" ") {
49             stringstream sin(s);
50             sin >> sm >> sp;
51             G[P(sp)][M(sm)] = 1;
52             while(sin >> sc) {
53                 G[M(sm)][C(sc)] = 1;
54                 G[C(sc)][TARGET()] = 1;
55             }
56         }
57
58         int maxParty = (EC.size()-1)/2;
59         for(map<string, int>::iterator it=EP.begin(); it!=EP.end(); it++) {
60             G[SOURCE()][it->second] = maxParty;
61         }
62
63         int total = 0;

```

```

65     while(int sent = send(SOURCE(), TARGET(), INT_MAX)) {
66         total += sent;
67         memset(V, 0, sizeof(V));
68     }
69
70     if (total == EC.size()) {
71         for(map<string, int>::iterator i=EM.begin(); i!=EM.end(); i++) {
72             for(map<string, int>::iterator j=EC.begin(); j!=EC.end(); j++) {
73                 if (G[j->second][i->second]) {
74                     cout << i->first << " " << j->first << endl;
75                 }
76             }
77         }
78     } else {
79         cout << "Impossible." << endl;
80     }
81
82     if (t) cout << endl;
83 }
84 }

```

graphs/10986.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #define MAX 200010
8
9  using namespace std;
10
11 struct Edge {
12     int v, c;
13     Edge(int v, int c) : v(v), c(c) {}
14     inline bool operator < (const Edge& that) const { return c > that.c; }
15 };
16
17 vector<Edge> G[MAX];
18 priority_queue<Edge> Q;
19 int n, m, s, t;
20 int V[MAX];
21
22
23 int main() {
24     int tt; cin >> tt; tt=0;
25     while(cin >> n >> m >> s >> t) {
26         int before = 0;
27         memset(V, 0x3f, sizeof(V));
28         memset(G, 0, sizeof(G));
29         Q = priority_queue<Edge>();
30
31         for(int i=0; i<m; i++) {
32             int a, b, c;
33             cin >> a >> b >> c;
34             G[a].push_back(Edge(b, c));
35             G[b].push_back(Edge(a, c));
36             before += c;
37         }
38
39         int totalc=0;
40
41         Q.push(Edge(s, 0));
42
43         while(totalc < n && !Q.empty()) {
44             Edge item = Q.top(); Q.pop();
45             if (item.c >= V[item.v]) continue;

```

```

46
47     V[item.v] = item.c;
48     totalc++;
49
50     for(int j=0; j<G[item.v].size(); j++) {
51         Edge e = G[item.v][j];
52         if (item.c + e.c < V[e.v])
53             Q.push(Edge(e.v, item.c + e.c));
54     }
55 }
56
57 cout << "Case #" << ++tt << ": ";
58 if (V[t] < 0x3f3f3f3f)
59     cout << V[t] << endl;
60 else
61     cout << "unreachable" << endl;
62 }
63 return 0;
64 }

```

graphs/11110.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <sstream>
4  #include <cstring>
5  #define MAX 102
6  using namespace std;
7
8  int G[MAX][MAX];
9  int n;
10
11 int fill(int x, int y, int v) {
12     if (G[x][y] != v) return 0;
13     if (x<=0 || x>n || y<=0 || y>n) return 0;
14
15     G[x][y] = -1;
16     return 1 +
17         fill(x-1, y, v) + fill(x+1, y, v) +
18         fill(x, y-1, v) + fill(x, y+1, v);
19 }
20
21 int main() {
22     while(cin >> n, n) {
23         int a, b; string s;
24         memset(G, 0, sizeof(G));
25
26         getline(cin, s);
27         for(int i=1; i<n; i++) {
28             getline(cin, s);
29             stringstream sin(s);
30             while(sin >> a >> b)
31                 G[a][b] = i;
32         }
33         bool good = true;
34         for(int i=1; i<=n; i++) {
35             for(int j=1; j<=n; j++) {
36                 if (G[i][j] >= 0)
37                     good &= fill(i, j, G[i][j]) == n;
38             }
39         }
40
41         cout << (good?"good":"wrong") << endl;
42
43     }
44     return 0;
45 }

```

graphs/11419.cpp

```
1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <string>
5  #include <cstdio>
6  #include <vector>
7  #define MAX 2005
8  using namespace std;
9
10 string VA[MAX], VB[MAX];
11 int G[MAX][MAX], n, r, c, p;
12 vector<int> G2[MAX];
13 bool V[MAX];
14
15 inline int SOURCE() { return 0; }
16 inline int TARGET() { return 1; }
17 inline int R(int i) { return 1+i; }
18 inline int C(int i) { return 1+r+i; }
19
20 int send(int s, int t, int minn) {
21     V[s] = true;
22
23     if (s==t) return minn;
24     for(int i=0; i<G2[s].size(); i++) {
25         int u = G2[s][i];
26         if (!V[u] && G[s][u] > 0) {
27             if (int sent = send(u, t, min(minn, G[s][u]))) {
28                 G[s][u] -= sent;
29                 G[u][s] += sent;
30                 return sent;
31             }
32         }
33     }
34     return 0;
35 }
36
37 void mark(int v, bool side) {
38     V[v] = true;
39     for(int i=0; i<G2[v].size(); i++) {
40         int u = G2[v][i];
41         if (!V[u] && (side && G[v][u] || !side && G[u][v]))
42             mark(u, !side);
43     }
44 }
45
46
47 int main() {
48     while(scanf("%d %d %d", &r, &c, &p), r|c|p) {
49         memset(G, 0, sizeof(G));
50         memset(G2, 0, sizeof(G2));
51         memset(V, 0, sizeof(V));
52
53         for(int i=1; i<=r; i++) {
54             G[SOURCE()][R(i)] = 1;
55             G2[SOURCE()].push_back(R(i));
56         }
57
58         for(int i=1; i<=c; i++) {
59             G[C(i)][TARGET()] = 1;
60             G2[C(i)].push_back(TARGET());
61         }
62
63         for(int i=0; i<p; i++) {
64             int a, b;
65             cin >> a >> b;
```



```

67         G[R(a)][C(b)] = 1;
68         G2[R(a)].push_back(C(b));
69         G2[C(b)].push_back(R(a));
70     }
71
72     n = r+c+1;
73
74     int total = 0;
75     while(int sent = send(SOURCE(), TARGET(), INT_MAX)) {
76         total += sent;
77         memset(V, 0, sizeof(V));
78     }
79
80     V[0] = V[1] = true;
81     for(int i=1; i<=r; i++) {
82         bool inflow = false;
83         for(int j=1; j<=c; j++)
84             inflow |= G[C(j)][R(i)];
85
86         if (!V[R(i)] && !inflow)
87             mark(R(i), true);
88     }
89     printf("%d", total);
90     for(int i=1; i<=r; i++)
91         if (!V[R(i)]) printf(" r%d", i);
92
93     for(int i=1; i<=c; i++)
94         if (V[C(i)]) printf(" c%d", i);
95
96     printf("\n");
97 }
98
99 return 0;
100 }

```

graphs/11518.cpp

```

1  #include <iostream>
2  #include <vector>
3  #include <cstring>
4  #define MAX 10002
5  using namespace std;
6
7  vector<int> G[MAX];
8  bool V[MAX];
9  int n,m,l;
10
11 int dfs(int v) {
12     if (V[v]) return 0;
13     V[v] = true;
14     int r = 1;
15     for(int i=0;i<G[v].size(); i++)
16         r+=dfs(G[v][i]);
17     return r;
18 }
19
20 int main() {
21     int t; cin >> t;
22     while(cin >> n >> m >> l) {
23         memset(G, 0, sizeof(G));
24         memset(V, 0, sizeof(V));
25
26         for(int i=0;i<m;i++) {
27             int a, b;
28             cin >> a >> b;
29             G[a].push_back(b);
30         }
31         int sum = 0;

```

```

32     for(int i=0;i<l;i++) {
33         int a;
34         cin >> a;
35         sum+=dfs(a);
36     }
37
38     cout << sum << endl;
39
40 }
41 return 0;
42 }

```

graphs/11631.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #define MAX 200010
8
9  using namespace std;
10
11 struct Road {
12     int v, c;
13     Road(int v, int c) : v(v), c(c) {}
14     inline bool operator < (const Road& that) const { return c > that.c; }
15 };
16
17 vector<Road> G[MAX];
18 priority_queue<Road> Q;
19 int n, m;
20 bool V[MAX];
21
22
23 int main() {
24     while(cin >> n >> m, n|m) {
25         int before = 0;
26         memset(V, 0, sizeof(V));
27         memset(G, 0, sizeof(G));
28         Q = priority_queue<Road>();
29
30         for(int i=0; i<m; i++) {
31             int a, b, c;
32             cin >> a >> b >> c;
33             G[a].push_back(Road(b, c));
34             G[b].push_back(Road(a, c));
35             before += c;
36         }
37
38         int total = 0, totalc=0;
39
40         Q.push(Road(0, 0));
41
42         while(totalc < n) {
43             Road item = Q.top(); Q.pop();
44             if (V[item.v]) continue;
45
46             V[item.v] = true;
47             total += item.c;
48             totalc++;
49
50             for(int j=0; j<G[item.v].size(); j++)
51                 if (!V[G[item.v][j].v])
52                     Q.push(G[item.v][j]);
53         }
54     }

```

```

55 |         cout << before-total << endl;
56 |     }
57 |     return 0;
58 | }

```

graphs/11686.cpp

```

1 | #include <iostream>
2 | #include <cstdio>
3 | #include <vector>
4 | #include <cstring>
5 | #define MAX 1000001
6 | using namespace std;
7 |
8 | int V[MAX];
9 | int O[MAX], npv;
10 | vector<int> G[MAX];
11 | int n, m;
12 |
13 | bool DFS(int d, int v){
14 |     V[v] = 1;
15 |
16 |     for(int i=0;i<G[v].size(); i++) {
17 |         int u = G[v][i];
18 |         if (V[u] == 1) return false;
19 |         if (!V[u] && !DFS(d, u)) return false;
20 |     }
21 |     O[++npv] = v;
22 |     V[v] = 2;
23 |     return true;
24 | }
25 |
26 |
27 | int main() {
28 |     int a, b;
29 |     while(scanf("%d%d",&n, &m), n|m) {
30 |         for(int i=1;i<=n;i++) G[i].clear();
31 |         npv = 0;
32 |         memset(V, 0, sizeof(V));
33 |         memset(O, 0, sizeof(O));
34 |
35 |         while(m--) {
36 |             scanf("%d%d",&a, &b);
37 |             G[a].push_back(b);
38 |         }
39 |
40 |
41 |         bool ok = true;
42 |         int d = 0;
43 |         for(int i = 1; i <= n; i++)
44 |             if (!V[i])
45 |                 ok &= DFS(++d, i);
46 |
47 |         if (ok)
48 |             for(int i = npv; i > 0; i--)
49 |                 printf("%d\n", O[i]);
50 |         else
51 |             printf("IMPOSSIBLE\n");
52 |     }
53 |
54 |     return 0;
55 | }

```

graphs/11709.cpp

```

1 | #include <iostream>

```

```

2  #include <map>
3  #include <string>
4  #include <cstring>
5  #define MAX 1001
6  using namespace std;
7
8  map<string, int> P;
9  int person(const string& p) {
10     if (P.find(p) != P.end())
11         return P[p];
12     else
13         return P[p] = P.size();
14 }
15
16 bool V[MAX];
17 int O[MAX], npv;
18 bool G[MAX][MAX];
19 int n, m;
20
21 void DFS(int v){
22     V[v] = true;
23     for(int i = 1; i <= n; i++)
24         if (G[v][i] && !V[i])
25             DFS(i);
26     O[++npv] = v;
27 }
28
29 void DFSt(int v){
30     V[v] = true;
31     for(int i = 1; i <= n; i++)
32         if (G[i][v] && !V[i])
33             DFSt(i);
34 }
35
36
37 int main() {
38     int a, b, t; string p, q;
39     while(cin >> n >> m, n|m) {
40         memset(G, 0, sizeof(G));
41         P.clear();
42         getline(cin, p);
43
44         for(int i=0; i<n; i++) getline(cin, p);
45
46         while(m--) {
47             getline(cin, p);
48             getline(cin, q);
49             G[person(p)][person(q)] = true;
50         }
51
52         npv = 0;
53         memset(V, 0, sizeof(V));
54         memset(O, 0, sizeof(O));
55
56         for(int i = 1; i <= n; i++)
57             if(!V[i]) DFS(i);
58
59         memset(V, 0, sizeof(V));
60
61         int comp = 0;
62         for(int i = n; i > 0; i--)
63             if(!V[O[i]]) {
64                 comp++;
65                 DFSt(O[i]);
66             }
67
68         cout << comp << endl;
69     }
70
71     return 0;

```

graphs/11733.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #define MAX 10005
8
9  using namespace std;
10
11 struct Road {
12     int v, c;
13     Road(int v, int c) : v(v), c(c) {}
14     inline bool operator < (const Road& that) const { return c > that.c; }
15 };
16
17 vector<Road> G[MAX];
18 int CStart[MAX], CCount[MAX], nc;
19 priority_queue<Road> Q;
20 int n, m, cca;
21 bool V[MAX];
22
23 int dfs(int v) {
24     V[v] = true;
25     int acum = 1;
26     for(int i=0; i<G[v].size(); i++)
27         if (!V[G[v][i].v])
28             acum += dfs(G[v][i].v);
29     return acum;
30 }
31
32 int main() {
33     int t; cin >> t; t=0;
34     while(cin >> n >> m >> cca) {
35         memset(V, 0, sizeof(V));
36         memset(G, 0, sizeof(G));
37         nc = 0;
38
39         for(int i=0; i<m; i++) {
40             int a, b, c;
41             cin >> a >> b >> c;
42             if (c<cca) {
43                 G[a].push_back(Road(b, c));
44                 G[b].push_back(Road(a, c));
45             }
46         }
47
48         for(int i=1; i<=n; i++) {
49             if (!V[i]) {
50                 CStart[nc]=i;
51                 CCount[nc]=dfs(i);
52                 nc++;
53             }
54         }
55
56         int total = nc*cca;
57
58         for(int i=0; i<nc; i++) {
59             int totalc = 0;
60             Q = priority_queue<Road>();
61             Q.push(Road(CStart[i], 0));
62             memset(V, 0, sizeof(V));
63
64             while(totalc < CCount[i]) {

```

```

65         Road item = Q.top(); Q.pop();
66         if (V[item.v]) continue;
67
68         V[item.v] = true;
69         total += item.c;
70         totalc++;
71
72         for(int j=0; j<G[item.v].size(); j++)
73             if (!V[G[item.v][j].v])
74                 Q.push(G[item.v][j]);
75     }
76 }
77
78     cout << "Case #" << ++t << ": " << total << " " << nc << endl;
79 }
80 return 0;
81 }

```

graphs/11747.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #define MAX 10005
8
9  using namespace std;
10
11 struct Road {
12     int v, c;
13     Road(int v, int c) : v(v), c(c) {}
14     inline bool operator < (const Road& that) const { return c > that.c; }
15 };
16
17 vector<Road> G[MAX];
18 int CStart[MAX], CCount[MAX], nc;
19 priority_queue<Road> Q;
20 vector<int> R;
21 int n, m;
22 bool V[MAX];
23
24 int dfs(int v) {
25     V[v] = true;
26     int acum = 1;
27     for(int i=0; i<G[v].size(); i++)
28         if (!V[G[v][i].v])
29             acum += dfs(G[v][i].v);
30     return acum;
31 }
32
33 int main() {
34     while(cin >> n >> m, n|m) {
35         memset(V, 0, sizeof(V));
36         memset(G, 0, sizeof(G));
37         nc = 0;
38         R.clear();
39
40         for(int i=0; i<m; i++) {
41             int a, b, c;
42             cin >> a >> b >> c;
43             G[a].push_back(Road(b, c));
44             G[b].push_back(Road(a, c));
45         }
46
47         for(int i=1; i<=n; i++) {
48             if (!V[i]) {

```

```

49         CStart[nc]=i;
50         CCount[nc]=dfs(i);
51         nc++;
52     }
53 }
54
55 for(int i=0; i<nc; i++) {
56     int totalc=0;
57     Q.push(Road(CStart[i], 0));
58     memset(V, 0, sizeof(V));
59
60     while(totalc < CCount[i]) {
61         Road item = Q.top(); Q.pop();
62         if (V[item.v]) { R.push_back(item.c); continue; }
63
64         V[item.v] = true;
65         totalc++;
66
67         for(int j=0; j<G[item.v].size(); j++)
68             if (!V[G[item.v][j].v])
69                 Q.push(G[item.v][j]);
70     }
71     while(!Q.empty()) {
72         R.push_back(Q.top().c);
73         Q.pop();
74     }
75 }
76 sort(R.begin(), R.end());
77 if (R.size()==0) {
78     cout << "forest" << endl;
79 } else {
80     cout << R[0];
81     for(int i=1; i<R.size(); i++)
82         cout << " " << R[i];
83     cout << endl;
84 }
85 }
86 return 0;
87 }

```

graphs/11770.cpp

```

1  #include <iostream>
2  #include <vector>
3  #include <cstring>
4  #define MAX 10001
5  using namespace std;
6
7  bool V[MAX];
8  int O[MAX], npv;
9  vector<int> G[MAX];
10 int n, m;
11
12 void DFS(int v){
13     if (V[v]) return;
14     V[v] = true;
15     for(int i = 0; i < G[v].size(); i++)
16         DFS(G[v][i]);
17     O[++npv] = v;
18 }
19
20 void DFSt(int v){
21     if (V[v]) return;
22     V[v] = true;
23     for(int i = 0; i < G[v].size(); i++)
24         DFSt(G[v][i]);
25 }
26

```

```

27
28 int main() {
29     int a, b;
30     int t; cin >> t; t=0;
31     while(cin >> n >> m) {
32         memset(G, 0, sizeof(G));
33
34         while(m--) {
35             cin >> a >> b;
36             G[a].push_back(b);
37         }
38
39         npv = 0;
40         memset(V, 0, sizeof(V));
41         memset(O, 0, sizeof(O));
42
43         for(int i = 1; i <= n; i++)
44             if(!V[i]) DFS(i);
45
46         memset(V, 0, sizeof(V));
47
48         int comp = 0;
49         for(int i = n; i > 0; i--)
50             if(!V[O[i]]) {
51                 comp++;
52                 DFSt(O[i]);
53             }
54
55         cout << "Case " << ++t << ": " << comp << endl;
56     }
57
58     return 0;
59 }

```

graphs/11833.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #define MAX 252
8
9  using namespace std;
10
11 struct Edge {
12     int v, c;
13     Edge(int v, int c) : v(v), c(c) {}
14     inline bool operator < (const Edge& that) const { return c > that.c; }
15 };
16
17 int G[MAX][MAX];
18 int V[MAX], S[MAX];
19 int n, m, cc, kk;
20
21 int main() {
22     while(cin >> n >> m >> cc >> kk, n|m|cc|kk) {
23         memset(V, 0x3f, sizeof(V));
24         memset(S, 0, sizeof(S));
25         memset(G, -1, sizeof(G));
26
27         for(int i=0; i<m; i++) {
28             int a, b, c;
29             cin >> a >> b >> c;
30             G[a][b] = G[b][a] = c;
31         }
32

```



```

33     for(int i=cc-2; i>=0; i--) {
34         S[i] = S[i+1] + G[i][i+1];
35     }
36
37     int totalc=0;
38
39     priority_queue<Edge> Q;
40     Q.push(Edge(kk, 0));
41
42     while(totalc < n && !Q.empty()) {
43         Edge item = Q.top(); Q.pop();
44         if (item.c >= V[item.v]) continue;
45         V[item.v] = item.c;
46         totalc++;
47         if (item.v < cc) continue;
48         for(int j=0; j<n; j++) {
49             if (G[item.v][j]>=0) {
50                 Edge e = Edge(j, G[item.v][j]);
51                 if (item.c + e.c < V[e.v])
52                     Q.push(Edge(e.v, item.c + e.c));
53             }
54         }
55     }
56
57     int minn = 0x3f3f3f3f;
58     for(int i=0; i<cc; i++) {
59         minn = min(minn, V[i]+S[i]);
60     }
61     cout << minn << endl;
62 }
63 return 0;
64 }

```

graphs/11838.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #define MAX 1001
4  using namespace std;
5
6  bool V[MAX];
7  int O[MAX], npv;
8  bool G[MAX][MAX];
9  int n, m;
10
11 void DFS(int v){
12     V[v] = true;
13     for(int i = 1; i <= n; i++)
14         if (G[v][i] && !V[i])
15             DFS(i);
16     O[++npv] = v;
17 }
18
19 void DFSt(int v){
20     V[v] = true;
21     for(int i = 1; i <= n; i++)
22         if (G[i][v] && !V[i])
23             DFSt(i);
24 }
25
26
27 int main() {
28     int a, b, t;
29     while(cin >> n >> m, n|m) {
30         memset(G, 0, sizeof(G));
31
32         while(m--) {
33             cin >> a >> b >> t;

```

```

34         G[a][b] = true;
35         if (t==2)
36             G[b][a] = true;
37     }
38
39     npv = 0;
40     memset(V, 0, sizeof(V));
41     memset(O, 0, sizeof(O));
42
43     for(int i = 1; i <= n; i++)
44         if(!V[i]) DFS(i);
45
46     memset(V, 0, sizeof(V));
47
48     int comp = 0;
49     for(int i = n; i > 0; i--)
50         if(!V[O[i]]) {
51             comp++;
52             DFSt(O[i]);
53         }
54
55     cout << (comp==1) << endl;
56 }
57
58 return 0;
59 }

```

graphs/12101.cpp

```

1  #include <iostream>
2  #include <queue>
3  #include <cstring>
4  #include <string>
5  using namespace std;
6
7  bool P[10000], V[10000];
8
9  struct Step {
10     int a, b, c, d, w;
11     Step() {}
12     Step(int a, int b, int c, int d, int w) : a(a), b(b), c(c), d(d), w(w) {}
13
14     int number() { return a*1000+b*100+c*10+d; }
15     bool valid() { return a && P[number()]; }
16
17     Step atA(int n) { return Step(n, b, c, d, w+1); }
18     Step atB(int n) { return Step(a, n, c, d, w+1); }
19     Step atC(int n) { return Step(a, b, n, d, w+1); }
20     Step atD(int n) { return Step(a, b, c, n, w+1); }
21 };
22
23 Step makestep(int n) {
24     int a, b, c, d;
25     d = n%10; n/=10;
26     c = n%10; n/=10;
27     b = n%10; n/=10;
28     a = n%10; n/=10;
29     return Step(a,b,c,d,0);
30 }
31
32 int main() {
33     memset(P, true, sizeof(P));
34     P[0] = P[1] = false;
35     for(int i=2; i<10000; i++) {
36         if (P[i]) {
37             for(int j=i*i; j<10000; j+=i)
38                 P[j] = false;
39         }

```

```

40     }
41     int t, a, b;
42     cin >> t;
43     while(cin >> a >> b) {
44         memset(V, 0, sizeof(V));
45         queue<Step> Q;
46         Q.push(makestep(a));
47         bool found = false;
48         while(!Q.empty()) {
49             Step step = Q.front(); Q.pop();
50             int n = step.number();
51             if (V[n]) continue;
52             V[n] = true;
53             if (n == b) {
54                 cout << step.w << endl;
55                 found = true;
56                 break;
57             }
58             for(int i=0;i<=9;i++) {
59                 Step sa = step.atA(i);
60                 Step sb = step.atB(i);
61                 Step sc = step.atC(i);
62                 Step sd = step.atD(i);
63                 if (sa.valid()) Q.push(sa);
64                 if (sb.valid()) Q.push(sb);
65                 if (sc.valid()) Q.push(sc);
66                 if (sd.valid()) Q.push(sd);
67             }
68         }
69         if (!found) cout << "Impossible" << endl;
70     }
71 }
72 }

```

graphs/12135.cpp

```

1  #include <iostream>
2  #include <queue>
3  #include <cstring>
4  #include <string>
5  #define MAX 33000
6  using namespace std;
7
8  vector<int> G[MAX];
9  int V[MAX];
10
11 int n, m;
12 struct Step {
13     int x, w;
14     Step() {}
15     Step(int x, int w) : x(x), w(w) {}
16 };
17
18 int main() {
19     int t; cin >> t; int tt=0;
20     while(cin >> n >> m, t--) {
21         memset(G, 0, sizeof(G));
22         memset(V, -1, sizeof(V));
23
24         n = 1<<n;
25
26         for(int i=0; i<m; i++) {
27             int a, b, mask=0;
28             cin >> a;
29             while(a--) {
30                 cin >> b;
31                 mask = mask | (1<<b);
32             }

```

```

33         for(int i=0; i<n; i++)
34             G[i].push_back(i^mask);
35     }
36
37     queue<Step> Q;
38     Q.push(Step(0, 0));
39     while(!Q.empty()) {
40         Step step = Q.front(); Q.pop();
41         if (V[step.x] >= 0) continue;
42
43         V[step.x] = step.w;
44         for(int i=0; i<G[step.x].size(); i++)
45             Q.push(Step(G[step.x][i], step.w+1));
46     }
47
48     cout << "Case " << ++tt << ":" << endl;
49     int q; string s;
50     cin >> q;
51     while(q-->0) {
52         int b = 0;
53         cin >> s;
54         for(int i=0; i<s.size(); i++)
55             b = b*2 + (s[i]-'0');
56
57         cout << V[b] << endl;
58     }
59     cout << endl;
60 }
61
62 }

```

graphs/12144.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <vector>
5  #include <algorithm>
6  #include <queue>
7  #define MAX 501
8
9  using namespace std;
10
11 struct Edge {
12     int u, v, c;
13     Edge(int u, int v, int c) : u(u), v(v), c(c) {}
14     inline bool operator < (const Edge& that) const { return c > that.c; }
15 };
16
17 int G[MAX][MAX];
18 int V[MAX];
19 vector<int> D[MAX];
20 int n, m, s, t;
21
22 void remove(int t) {
23     if (D[t].size() == 0 || t == D[t][0]) return;
24     for(int i=0; i<D[t].size(); i++) {
25         G[D[t][i]][t] = 0;
26         remove(D[t][i]);
27     }
28 }
29
30 int shortest() {
31     memset(V, 0x3f, sizeof(V));
32     memset(D, 0, sizeof(D));
33     priority_queue<Edge> Q;
34     Q.push(Edge(s, s, 0));
35

```

```

36     while(!Q.empty()) {
37         Edge item = Q.top(); Q.pop();
38         if (item.c > V[item.v]) continue;
39         V[item.v] = item.c;
40         D[item.v].push_back(item.u);
41
42         for(int j=0; j<n; j++) {
43             if (G[item.v][j]) {
44                 Edge e = Edge(item.v, j, item.c+G[item.v][j]);
45                 if (e.c <= V[e.v])
46                     Q.push(e);
47             }
48         }
49     }
50     remove(t);
51     if (V[t] < 0x3f3f3f3f)
52         return V[t];
53     else
54         return -1;
55 }
56
57 int main() {
58     while(cin >> n >> m, n|m) {
59         cin >> s >> t;
60         memset(G, 0, sizeof(G));
61
62         for(int i=0; i<m; i++) {
63             int a, b, c;
64             cin >> a >> b >> c;
65             G[a][b] = c;
66         }
67
68         shortest();
69         cout << shortest() << endl;
70     }
71     return 0;
72 }
73

```

graphs/12159.cpp

```

1  #include <iostream>
2  #include <iomanip>
3  #include <cstring>
4  #include <vector>
5  #include <cmath>
6  #include <climits>
7  #include <vector>
8  #include <cassert>
9  #define MAX 306
10 using namespace std;
11
12 int X[MAX], Y[MAX], P[MAX], G[MAX][MAX], n, r, a, b;
13 bool V[MAX];
14
15 bool team(int c) {
16     return (X[b] - X[a])*(Y[c] - Y[a]) - (Y[b] - Y[a])*(X[c] - X[a]) > 0;
17 }
18
19 int sqrdist(int a, int b) {
20     return (X[a]-X[b])*(X[a]-X[b])+(Y[a]-Y[b])*(Y[a]-Y[b]);
21 }
22
23 int send(int s, int t, int minn) {
24     V[s] = true;
25
26     if (s==t) return minn;
27     for(int i=0; i<n; i++) {

```

```

28         if (!V[i] && G[s][i] > 0) {
29             if (int sent = send(i, t, min(minn, G[s][i]))) {
30                 G[s][i] -= sent;
31                 G[i][s] += sent;
32                 return sent;
33             }
34         }
35     }
36     return 0;
37 }
38
39 int main() {
40     int t=0;
41     while(cin >> n, n) {
42         memset(G, 0, sizeof(G));
43
44         for(int i=1;i<=n;i++)
45             cin >> X[i] >> Y[i] >> P[i];
46         cin >> a >> b >> r;
47
48         vector<int> A, B;
49         for(int i=1;i<=n;i++) {
50             if (P[i] == 0) continue;
51             if (team(i))
52                 B.push_back(i);
53             else
54                 A.push_back(i);
55         }
56         if (A.size() > B.size()) A.swap(B);
57
58         for(int i=0; i<A.size(); i++) {
59             int u=A[i];
60             G[0][u] = 1;
61             for(int j=0; j<B.size(); j++) {
62                 int v = B[j];
63                 G[v][n+1] = 1;
64                 if (sqrdist(u, v) <= r*r && P[u] > P[v])
65                     G[u][v] = 1;
66             }
67         }
68         n++;
69
70         memset(V, 0, sizeof(V));
71         int total = 0;
72         while(int sent = send(0, n, INT_MAX)) {
73             total += sent;
74             memset(V, 0, sizeof(V));
75         }
76         cout << "Case " << ++t << ": " << total << endl;
77     }
78 }

```

graphs/12160.cpp

```

1  #include <iostream>
2  #include <queue>
3  #include <cstring>
4  #include <string>
5  using namespace std;
6
7  bool V[10000];
8  int R[10];
9
10 struct Step {
11     int x, w;
12     Step() {}
13     Step(int x, int w) : x(x), w(w) {}
14

```

```

15     Step sum(int n) {
16         return Step((x+n)%10000, w+1);
17     }
18 };
19
20 int main() {
21     int a, b, n, t=0;
22     while(cin >> a >> b >> n, a|b|n) {
23         for(int i=0;i<n;i++)
24             cin >> R[i];
25
26         cout << "Case " << ++t << ": ";
27         memset(V, 0, sizeof(V));
28         queue<Step> Q;
29         Q.push(Step(a, 0));
30         bool found = false;
31         while(!Q.empty()) {
32             Step step = Q.front(); Q.pop();
33             if (V[step.x]) continue;
34             V[step.x] = true;
35             if (step.x == b) {
36                 cout << step.w << endl;
37                 found = true;
38                 break;
39             }
40             for(int i=0;i<n;i++)
41                 Q.push(step.sum(R[i]));
42         }
43         if (!found) cout << "Permanently Locked" << endl;
44     }
45 }
46 }

```

graphs/12168.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <climits>
4  #include <string>
5  #define MAX 505
6  using namespace std;
7
8  string V1[MAX], V2[MAX];
9  int G[MAX][MAX], n;
10 bool V[MAX];
11
12 int send(int s, int t, int minn) {
13     V[s] = true;
14
15     if (s==t) return minn;
16     for(int i=0; i<=n; i++) {
17         if (!V[i] && G[s][i] > 0) {
18             if (int sent = send(i, t, min(minn, G[s][i]))) {
19                 G[s][i] -= sent;
20                 G[i][s] += sent;
21                 return sent;
22             }
23         }
24     }
25     return 0;
26 }
27
28 int main() {
29     int t; cin >> t;
30     int c, d, v;
31
32     while(cin >> c >> d >> v, t--) {
33         memset(G, 0, sizeof(G));

```

```

34     memset(V, 0, sizeof(V));
35
36     string s1, s2;
37     for(int i=1; i<=v; i++) {
38         cin >> s1 >> s2;
39         V1[i] = s1; V2[i] = s2;
40
41         bool dog = s1[0] == 'D';
42
43         if (dog)
44             G[0][i] = 1;
45         else
46             G[i][v+1] = 1;
47
48         for(int j=1; j<i; j++) {
49             if (s1 == V2[j] || s2 == V1[j])
50                 if (dog)
51                     G[i][j] = 1;
52                 else
53                     G[j][i] = 1;
54         }
55     }
56     n = v+1;
57
58     int total = 0;
59     while(int sent = send(0, n, INT_MAX)) {
60         total += sent;
61         memset(V, 0, sizeof(V));
62     }
63     cout << v-total << endl;
64 }
65
66 return 0;
67 }

```

graphs/12179.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <cmath>
4  #include <iomanip>
5  #define MAX 101
6  using namespace std;
7
8  int G[MAX][MAX], n, r, c;
9  double P[101][10001];
10
11 int main() {
12     int t; cin >> t; t=0;
13     cout << fixed << setprecision(6);
14
15     while(cin >> n >> r) {
16         memset(G, 0x3F, sizeof(G));
17         memset(P, 0, sizeof(P));
18
19         char cc;
20         for(int i=0; i<n; i++) {
21             for(int j=0; j<n; j++) {
22                 cin >> cc;
23                 if (cc=='Y') G[i][j] = 1;
24             }
25         }
26
27         for(int k=0; k<n; k++)
28             for(int i=0; i<n; i++)
29                 for(int j=0; j<n; j++)
30                     G[i][j] = min(G[i][j], G[i][k] + G[k][j]);
31     }

```



```

32     P[0][0] = 1;
33     double pp = 1.0/r;
34     for(int i=1; i<=100; i++)
35         for(int k=1; k<=r; k++)
36             for(int j=k; j<=100*r; j++)
37                 P[i][j] += P[i-1][j-k] * pp;
38
39     cout << "Case " << ++t << endl;
40     cin >> c;
41     while(c--) {
42         int a, b, m;
43         cin >> a >> b >> m;
44         a--; b--;
45
46         int d=G[a][b];
47
48         double total = 0;
49         for(int i=0; i<=m; i++)
50             total += P[d][i];
51         cout << total << endl;
52     }
53     cout << endl;
54 }
55 return 0;
56 }

```

graphs/12186.cpp

```

1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  #include <cstring>
5  #include <cmath>
6  #define MAX 100002
7  using namespace std;
8
9  vector<int> G[MAX];
10 int n, t;
11
12 int dfs(int v) {
13     if (G[v].empty()) return 1;
14     vector<int> mins;
15     for(int i=0; i<G[v].size(); i++)
16         mins.push_back(dfs(G[v][i]));
17     sort(mins.begin(), mins.end());
18
19     int get = (int)ceil(G[v].size()*t/100.0);
20     int sum = 0;
21     for(int i=0; i<get; i++) sum+=mins[i];
22     return sum;
23 }
24
25 int main() {
26     int boss;
27     while(cin >> n >> t, n|t) {
28         memset(G, 0, sizeof(G));
29         for(int i=1; i<=n; i++) {
30             cin >> boss; G[boss].push_back(i);
31         }
32         cout << dfs(0) << endl;
33     }
34     return 0;
35 }

```

Math

- GCD
 - [UVa 12184 - Transcribed Books](#)
- Sieve
 - [UVa 1246 - Find Terrorists](#)
- Prime Factorization
 - [UVa 12137 - Puzzles of Triangles](#)
- Geometry
 - [UVa 12300 - Smallest Regular Polygon](#)
 - [UVa 12194 - Isosceles Triangles](#)

math/1246.cpp

```
1  #include <iostream>
2  #include <vector>
3  #include <cstring>
4  using namespace std;
5
6  bool P[100];
7  int T[1000001];
8  vector<int> W;
9
10 long long real_mod(long long a, long long b) {
11     long long c = a%b;
12     if (c<0) c+=b;
13     return c;
14 }
15
16 int main() {
17     int n, k;
18
19     memset(P, true, sizeof(P));
20     P[0] = P[1] = false;
21     for(int i=2; i<100; i++) {
22         if (P[i]) {
23             W.push_back(i);
24             for(int j=i*i; j>=0 && j<100; j+=i)
25                 P[j] = false;
26         }
27     }
28
29     int t; cin >> t; t=0;
30     int a, b;
31     while(cin >> a >> b) {
32         memset(T, 0, sizeof(int)*(b-a+1));
33
34         if (a==0) { T[0]-=2; T[1] -= 1; }
35         if (a==1) { T[0]-=1; }
36
37         for(long long i=2; i*i<=b; i++) {
38             for(long long j=max(real_mod(i*i+i-a, i), i*i+i-a); j<=(b-a); j+=i) {
39                 T[j]+=2;
40             }
41             int tmp = i*i-a;
42             if (tmp >= 0 && tmp <= (b-a))
43                 T[tmp]++;
44         }
45
46         int cnt=0;
47         for(int i=0; i<=(b-a);i++) {
48             if (P[T[i]+2]) {
49                 cout << (cnt++?" ":"") << i+a;
50             }
51         }
52         if (!cnt) cout << -1;
53         cout << endl;
54     }
55 }
56 }
```

math/12137.cpp

```
1  #include <string.h>
2  #include <stdio.h>
3  #define ull unsigned long long
4
5  int W[] = { 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 473, 479, 487, 491, 499, 503, 509, 521, 523, 527, 531, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 623, 627, 631, 637, 641, 643, 647, 653, 659, 661, 667, 671, 673, 677, 683, 687, 691, 693, 697, 701, 703, 707, 709, 713, 719, 727, 733, 739, 743, 749, 751, 757, 761, 763, 767, 773, 779, 787, 791, 793, 797, 803, 809, 811, 817, 821, 823, 827, 829, 833, 837, 839, 843, 847, 851, 853, 857, 859, 863, 867, 869, 873, 877, 881, 883, 887, 893, 897, 901, 907, 911, 913, 917, 919, 923, 927, 931, 937, 941, 943, 947, 953, 959, 961, 967, 971, 973, 977, 983, 989, 991, 993, 997, 1003, 1009, 1013, 1017, 1019, 1021, 1023, 1027, 1031, 1033, 1037, 1039, 1043, 1047, 1049, 1053, 1057, 1061, 1063, 1067, 1069, 1073, 1077, 1081, 1087, 1091, 1093, 1097, 1103, 1107, 1113, 1117, 1121, 1123, 1127, 1129, 1133, 1137, 1141, 1147, 1151, 1153, 1157, 1163, 1167, 1171, 1173, 1177, 1181, 1183, 1187, 1193, 1197, 1201, 1203, 1207, 1211, 1213, 1217, 1219, 1223, 1227, 1231, 1237, 1241, 1243, 1247, 1249, 1253, 1257, 1259, 1261, 1267, 1271, 1273, 1277, 1283, 1287, 1291, 1293, 1297, 1301, 1303, 1307, 1309, 1313, 1317, 1321, 1323, 1327, 1329, 1333, 1337, 1341, 1343, 1347, 1349, 1353, 1357, 1361, 1363, 1367, 1369, 1373, 1377, 1381, 1383, 1387, 1393, 1397, 1401, 1403, 1407, 1409, 1411, 1413, 1417, 1421, 1423, 1427, 1429, 1433, 1437, 1441, 1443, 1447, 1449, 1453, 1457, 1461, 1463, 1467, 1469, 1473, 1477, 1481, 1483, 1487, 1493, 1497, 1501, 1503, 1507, 1511, 1513, 1517, 1519, 1523, 1527, 1531, 1537, 1541, 1543, 1547, 1549, 1553, 1557, 1561, 1563, 1567, 1569, 1573, 1577, 1581, 1583, 1587, 1593, 1597, 1601, 1603, 1607, 1609, 1613, 1617, 1621, 1623, 1627, 1629, 1633, 1637, 1641, 1643, 1647, 1649, 1653, 1657, 1661, 1663, 1667, 1669, 1673, 1677, 1681, 1683, 1687, 1693, 1697, 1701, 1703, 1707, 1709, 1713, 1717, 1721, 1723, 1727, 1729, 1733, 1737, 1741, 1743, 1747, 1749, 1753, 1757, 1761, 1763, 1767, 1769, 1773, 1777, 1781, 1783, 1787, 1793, 1797, 1801, 1803, 1807, 1811, 1813, 1817, 1821, 1823, 1827, 1829, 1833, 1837, 1841, 1843, 1847, 1849, 1853, 1857, 1861, 1863, 1867, 1869, 1873, 1877, 1881, 1883, 1887, 1893, 1897, 1901, 1903, 1907, 1909, 1913, 1917, 1921, 1923, 1927, 1929, 1933, 1937, 1941, 1943, 1947, 1949, 1953, 1957, 1961, 1963, 1967, 1969, 1973, 1977, 1981, 1983, 1987, 1993, 1997, 2003, 2009, 2011, 2013, 2017, 2019, 2023, 2027, 2031, 2033, 2037, 2039, 2043, 2047, 2051, 2053, 2057, 2059, 2063, 2067, 2071, 2073, 2077, 2081, 2083, 2087, 2093, 2097, 2101, 2103, 2107, 2109, 2113, 2117, 2121, 2123, 2127, 2129, 2133, 2137, 2141, 2143, 2147, 2149, 2153, 2157, 2161, 2163, 2167, 2169, 2173, 2177, 2181, 2183, 2187, 2193, 2197, 2201, 2203, 2207, 2209, 2213, 2217, 2221, 2223, 2227, 2229, 2233, 2237, 2241, 2243, 2247, 2249, 2253, 2257, 2261, 2263, 2267, 2269, 2273, 2277, 2281, 2283, 2287, 2293, 2297, 2301, 2303, 2307, 2309, 2313, 2317, 2321, 2323, 2327, 2329, 2333, 2337, 2341, 2343, 2347, 2349, 2353, 2357, 2361, 2363, 2367, 2369, 2373, 2377, 2381, 2383, 2387, 2393, 2397, 2401, 2403, 2407, 2409, 2413, 2417, 2421, 2423, 2427, 2429, 2433, 2437, 2441, 2443, 2447, 2449, 2453, 2457, 2461, 2463, 2467, 2469, 2473, 2477, 2481, 2483, 2487, 2493, 2497, 2501, 2503, 2507, 2509, 2513, 2517, 2521, 2523, 2527, 2529, 2533, 2537, 2541, 2543, 2547, 2549, 2553, 2557, 2561, 2563, 2567, 2569, 2573, 2577, 2581, 2583, 2587, 2593, 2597, 2601, 2603, 2607, 2609, 2613, 2617, 2621, 2623, 2627, 2629, 2633, 2637, 2641, 2643, 2647, 2649, 2653, 2657, 2661, 2663, 2667, 2669, 2673, 2677, 2681, 2683, 2687, 2693, 2697, 2701, 2703, 2707, 2709, 2713, 2717, 2721, 2723, 2727, 2729, 2733, 2737, 2741, 2743, 2747, 2749, 2753, 2757, 2761, 2763, 2767, 2769, 2773, 2777, 2781, 2783, 2787, 2793, 2797, 2801, 2803, 2807, 2809, 2813, 2817, 2821, 2823, 2827, 2829, 2833, 2837, 2841, 2843, 2847, 2849, 2853, 2857, 2861, 2863, 2867, 2869, 2873, 2877, 2881, 2883, 2887, 2893, 2897, 2901, 2903, 2907, 2909, 2913, 2917, 2921, 2923, 2927, 2929, 2933, 2937, 2941, 2943, 2947, 2949, 2953, 2957, 2961, 2963, 2967, 2969, 2973, 2977, 2981, 2983, 2987, 2993, 2997, 3001, 3003, 3007, 3009, 3013, 3017, 3021, 3023, 3027, 3029, 3033, 3037, 3041, 3043, 3047, 3049, 3053, 3057, 3061, 3063, 3067, 3069, 3073, 3077, 3081, 3083, 3087, 3093, 3097, 3101, 3103, 3107, 3109, 3113, 3117, 3121, 3123, 3127, 3129, 3133, 3137, 3141, 3143, 3147, 3149, 3153, 3157, 3161, 3163, 3167, 3169, 3173, 3177, 3181, 3183, 3187, 3193, 3197, 3201, 3203, 3207, 3209, 3213, 3217, 3221, 3223, 3227, 3229, 3233, 3237, 3241, 3243, 3247, 3249, 3253, 3257, 3261, 3263, 3267, 3269, 3273, 3277, 3281, 3283, 3287, 3293, 3297, 3301, 3303, 3307, 3309, 3313, 3317, 3321, 3323, 3327, 3329, 3333, 3337, 3341, 3343, 3347, 3349, 3353, 3357, 3361, 3363, 3367, 3369, 3373, 3377, 3381, 3383, 3387, 3393, 3397, 3401, 3403, 3407, 3409, 3413, 3417, 3421, 3423, 3427, 3429, 3433, 3437, 3441, 3443, 3447, 3449, 3453, 3457, 3461, 3463, 3467, 3469, 3473, 3477, 3481, 3483, 3487, 3493, 3497, 3501, 3503, 3507, 3509, 3513, 3517, 3521, 3523, 3527, 3529, 3533, 3537, 3541, 3543, 3547, 3549, 3553, 3557, 3561, 3563, 3567, 3569, 3573, 3577, 3581, 3583, 3587, 3593, 3597, 3601, 3603, 3607, 3609, 3613, 3617, 3621, 3623, 3627, 3629, 3633, 3637, 3641, 3643, 3647, 3649, 3653, 3657, 3661, 3663, 3667, 3669, 3673, 3677, 3681, 3683, 3687, 3693, 3697, 3701, 3703, 3707, 3709, 3713, 3717, 3721, 3723, 3727, 3729, 3733, 3737, 3741, 3743, 3747, 3749, 3753, 3757, 3761, 3763, 3767, 3769, 3773, 3777, 3781, 3783, 3787, 3793, 3797, 3801, 3803, 3807, 3809, 3813, 3817, 3821, 3823, 3827, 3829, 3833, 3837, 3841, 3843, 3847, 3849, 3853, 3857, 3861, 3863, 3867, 3869, 3873, 3877, 3881, 3883, 3887, 3893, 3897, 3901, 3903, 3907, 3909, 3913, 3917, 3921, 3923, 3927, 3929, 3933, 3937, 3941, 3943, 3947, 3949, 3953, 3957, 3961, 3963, 3967, 3969, 3973, 3977, 3981, 3983, 3987, 3993, 3997, 4001, 4003, 4007, 4009, 4013, 4017, 4021, 4023, 4027, 4029, 4033, 4037, 4041, 4043, 4047, 4049, 4053, 4057, 4061, 4063, 4067, 4069, 4073, 4077, 4081, 4083, 4087, 4093, 4097, 4101, 4103, 4107, 4109, 4113, 4117, 4121, 4123, 4127, 4129, 4133, 4137, 4141, 4143, 4147, 4149, 4153, 4157, 4161, 4163, 4167, 4169, 4173, 4177, 4181, 4183, 4187, 4193, 4197, 4201, 4203, 4207, 4209, 4213, 4217, 4221, 4223, 4227, 4229, 4233, 4237, 4241, 4243, 4247, 4249, 4253, 4257, 4261, 4263, 4267, 4269, 4273, 4277, 4281, 4283, 4287, 4293, 4297, 4301, 4303, 4307, 4309, 4313, 4317, 4321, 4323, 4327, 4329, 4333, 4337, 4341, 4343, 4347, 4349, 4353, 4357, 4361, 4363, 4367, 4369, 4373, 4377, 4381, 4383, 4387, 4393, 4397, 4401, 4403, 4407, 4409, 4413, 4417, 4421, 4423, 4427, 4429, 4433, 4437, 4441, 4443, 4447, 4449, 4453, 4457, 4461, 4463, 4467, 4469, 4473, 4477, 4481, 4483, 4487, 4493, 4497, 4501, 4503, 4507, 4509, 4513, 4517, 4521, 4523, 4527, 4529, 4533, 4537, 4541, 4543, 4547, 4549, 4553, 4557, 4561, 4563, 4567, 4569, 4573, 4577, 4581, 4583, 4587, 4593, 4597, 4601, 4603, 4607, 4609, 4613, 4617, 4621, 4623, 4627, 4629, 4633, 4637, 4641, 4643, 4647, 4649, 4653, 4657, 4661, 4663, 4667, 4669, 4673, 4677, 4681, 4683, 4687, 4693, 4697, 4701, 4703, 4707, 4709, 4713, 4717, 4721, 4723, 4727, 4729, 4733, 4737, 4741, 4743, 4747, 4749, 4753, 4757, 4761, 4763, 4767, 4769, 4773, 4777, 4781, 4783, 4787, 4793, 4797, 4801, 4803, 4807, 4809, 4813, 4817, 4821, 4823, 4827, 4829, 4833, 4837, 4841, 4843, 4847, 4849, 4853, 4857, 4861, 4863, 4867, 4869, 4873, 4877, 4881, 4883, 4887, 4893, 4897, 4901, 4903, 4907, 4909, 4913, 4917, 4921, 4923, 4927, 4929, 4933, 4937, 4941, 4943, 4947, 4949, 4953, 4957, 4961, 4963, 4967, 4969, 4973, 4977, 4981, 4983, 4987, 4993, 4997, 5001, 5003, 5007, 5009, 5013, 5017, 5021, 5023, 5027, 5029, 5033, 5037, 5041, 5043, 5047, 5049, 5053, 5057, 5061, 5063, 5067, 5069, 5073, 5077, 5081, 5083, 5087, 5093, 5097, 5101, 5103, 5107, 5109, 5113, 5117, 5121, 5123, 5127, 5129, 5133, 5137, 5141, 5143, 5147, 5149, 5153, 5157, 5161, 5163, 5167, 5169, 5173, 5177, 5181, 5183, 5187, 5193, 5197, 5201, 5203, 5207, 5209, 5213, 5217, 5221, 5223, 5227, 5229, 5233, 5237, 5241, 5243, 5247, 5249, 5253, 5257, 5261, 5263, 5267, 5269, 5273, 5277, 5281, 5283, 5287, 5293, 5297, 5301, 5303, 5307, 5309, 5313, 5317, 5321, 5323, 5327, 5329, 5333, 5337, 5341, 5343, 5347, 5349, 5353, 5357, 5361, 5363, 5367, 5369, 5373, 5377, 5381, 5383, 5387, 5393, 5397, 5401, 5403, 5407, 5409, 5413, 5417, 5421, 5423, 5427, 5429, 5433, 5437, 5441, 5443, 5447, 5449, 5453, 5457, 5461, 5463, 5467, 5469, 5473, 5477, 5481, 5483, 5487, 5493, 5497, 5501, 5503, 5507, 5509, 5513, 5517, 5521, 5523, 5527, 5529, 5533, 5537, 5541, 5543, 5547, 5549, 5553, 5557, 5561, 5563, 5567, 5569, 5573, 5577, 5581, 5583, 5587, 5593, 5597, 5601, 5603, 5607, 5609, 5613, 5617, 5621, 5623, 5627, 5629, 5633, 5637, 5641, 5643, 5647, 5649, 5653, 5657, 5661, 5663, 5667, 5669, 5673, 5677, 5681, 5683, 5687, 5693, 5697, 5701, 5703, 5707, 5709, 5713, 5717, 5721, 5723, 5727, 5729, 5733, 5737, 5741, 5743, 5747, 5749, 5753, 5757, 5761, 5763, 5767, 5769, 5773, 5777, 5781, 5783, 5787, 5793, 5797, 5801, 5803, 5807, 5809, 5813, 5817, 5821, 5823, 5827, 5829, 5833, 5837, 5841, 5843, 5847, 5849, 5853, 5857, 5861, 5863, 5867, 5869, 5873, 5877, 5881, 5883, 5887, 5893, 5897, 5901, 5903, 5907, 5909, 5913, 5917, 5921, 5923, 5927, 5929, 5933, 5937, 5941, 5943, 5947, 5949, 5953, 5957, 5961, 5963, 5967, 5969, 5973, 5977, 5981, 5983, 5987, 5993, 5997, 6001, 6003, 6007, 6009, 6013, 6017, 6021, 6023, 6027, 6029, 6033, 6037, 6041, 6043, 6047, 6049, 6053, 6057, 6061, 6063, 6067, 6069, 6073, 6077, 6081, 6083, 6087, 6093, 6097, 6101, 6103, 6107, 6109, 6113, 6117, 6121, 6123, 6127, 6129, 6133, 6137, 6141, 6143, 6147, 6149, 6153, 6157, 6161, 6163, 6167, 6169, 6173, 6177, 6181, 6183, 6187, 6193, 6197, 6201, 6203, 6207, 6209, 6213, 6217, 6221, 6223, 6227, 6229, 6233, 6237, 6241, 6243, 6247, 6249, 6253, 6257, 6261, 6263, 6267, 6269, 6273, 6277, 6281, 6283, 6287, 6293, 6297, 6301, 6303, 6307, 6309, 6313, 6317, 6321, 6323, 6327, 6329, 6333, 6337, 6341, 6343, 6347, 6349, 6353, 6357, 6361, 6363, 6367, 6369, 6373, 6377, 6381, 6383, 6387, 6393, 6397, 6401, 6403, 6407, 6409, 6413, 6417, 6421, 6423, 6427, 6429, 6433, 6437, 6441, 6443, 6447, 6449, 6453, 6457, 6461, 6463, 6467, 6469, 6473, 6477, 6481, 6483, 6487, 6493, 6497, 6501, 6503, 6507, 6509, 6513, 6517, 6521, 6523, 6527, 6529, 6533, 6537, 6541, 6543, 6547, 6549, 6553, 6557, 6561, 6563, 6567, 6569, 6573, 6577, 6581, 6583, 6587, 6593, 6597, 6601, 6603, 6607, 6609, 6613, 6617, 6621, 6623, 6627, 6629, 6633, 6637, 6641, 6643, 6647, 6649, 6653, 6657, 6661, 6663, 6667, 6669, 6673, 6677, 6681, 6683, 6687, 6693, 6697, 6701, 6703, 6707, 6709, 6713, 6717, 6721, 6723, 6727, 6729, 6733, 6737, 6741, 6743, 6747, 6749, 6753, 6757, 6761, 6763, 6767, 6769, 6773, 6777, 6781, 6783, 6787, 6793, 6797, 6801, 6803, 6807, 6809, 6813, 6817, 6821, 6823, 6827, 6829, 6833, 6837, 6841, 6843, 6847, 6849, 6853, 6857, 6861, 6863, 6867, 6869, 6873, 6877, 6881, 6883, 6887, 6893, 6897, 6901, 6903, 6907, 6909, 6913, 6917, 6921, 6923, 6927, 6929, 6933, 6937, 6941, 6943, 6947, 6949, 6953, 6957, 6961, 6963, 6967, 6969, 6973, 6977, 6981, 6983, 6987, 6993, 6997, 7001, 7003, 7007, 7009, 7013, 7017, 7021, 7023, 7027, 7029, 7033, 7037, 7041, 7043, 7047, 7049, 7053, 7057, 7061, 7063, 7067, 7069, 7073, 7077, 
```

```

7 | inline ull div(const ull& a, const ull& b, ull &r) {
8 |     r = a/b;
9 |     return a-r*b;
10| }
11|
12| inline ull pow(const ull& a, const int b) {
13|     if (b==0) return 1;
14|     ull tmp = b&1 ? a : 1;
15|     ull r = pow(a, b>>1);
16|     return tmp*r*r;
17| }
18|
19| int main() {
20|     ull n;
21|     int t=0;
22|     while(scanf("%llu", &n), n) {
23|         ull ncopy = n;
24|         ull step = 1;
25|         for(int i=0; ncopy>1 && i<wn; i++) {
26|             int power=0;
27|             ull divr;
28|             while(div(ncopy, W[i], divr)==0) {
29|                 ncopy = divr;
30|                 power++;
31|             }
32|             step *= pow(W[i], (power+1)/2);
33|         }
34|         step *= ncopy;
35|
36|         ull result;
37|         if (div(n, step, result)==0) result--;
38|         result *= 8;
39|
40|         if(result)
41|             printf("Case %d: %llu\n",++t, result);
42|         else
43|             printf("Case %d: Impossible\n", ++t);
44|     }
45| }

```

math/12184.cpp

```

1 | #include <iostream>
2 | using namespace std;
3 |
4 | long gcd(long a, long b) {
5 |     while(b) {
6 |         long c = a%b;
7 |         a = b;
8 |         b = c;
9 |     }
10|     return a;
11| }
12|
13| int main() {
14|     int t; cin >> t;
15|     int n;
16|     while(cin >> n) {
17|         long result = 0;
18|         long maxSerial = 0;
19|         for(int i=0; i<n; i++) {
20|             long s=0, tmp;
21|             for(int j=0; j<9; j++) {
22|                 cin >> tmp; s+=tmp;
23|             }
24|             cin >> tmp;
25|             s -= tmp;
26|             maxSerial = max(maxSerial, tmp);

```

```

27         result = gcd(result, s);
28     }
29     if (result>1 && maxSerial < result)
30         cout << result << endl;
31     else
32         cout << "impossible" << endl;
33 }
34 }

```

math/12194.cpp

```

1  #include <cstdio>
2  #include <algorithm>
3  #include <cstring>
4  #define MAX 1010
5  using namespace std;
6
7  int X[MAX], Y[MAX];
8  long T[MAX][MAX];
9  int C[MAX];
10
11 inline long sqr(long v) { return v*v; }
12
13 int main(){
14     int n;
15     while(scanf("%d", &n), n) {
16         memset(C, 0, sizeof(C));
17
18         for(int i=0; i<n; i++)
19             scanf("%d %d", &X[i], &Y[i]);
20
21         int sum = 0;
22         for(int i=0; i<n; i++) {
23             for(int j=0; j<n; j++)
24                 T[i][C[i]++] = sqr(X[i]-X[j])+sqr(Y[i]-Y[j]);
25             sort(T[i], T[i]+C[i]);
26             long last=-1L;
27             int cnt=0;
28             for(int j=0; j<C[i]; j++) {
29                 if (T[i][j] != last) {
30                     sum += cnt*(cnt-1)/2;
31                     cnt = 0;
32                 }
33                 last = T[i][j];
34                 cnt++;
35             }
36             sum += cnt*(cnt-1)/2;
37         }
38
39         printf("%d\n", sum);
40     }
41 }
42 }

```

math/12300.cpp

```

1  #include <iostream>
2  #include <cmath>
3  #include <iomanip>
4  #define PI 3.141592653589793238462
5  using namespace std;
6
7  double cot(double angle) {
8      return cos(angle)/sin(angle);
9  }
10

```

```

11 | int main(){
12 |     int x1, y1, x2, y2, n;
13 |     while(cin >> x1 >> y1 >> x2 >> y2 >> n, x1 | y1 | x2 | y2 | n) {
14 |         double d = sqrt(pow(x2-x1, 2.0)+pow(y2-y1, 2.0));
15 |         int k = n/2;
16 |         double s = sin(PI/n)/sin(PI*k/n)*d;
17 |         double A = 0.25*n*s*s*cot(PI/n);
18 |         setprecision(6);
19 |         cout << fixed << A << endl;
20 |     }
21 |
22 | }

```

Misc

- STL map
 - [UVa 10420 - List of Conquests](#)
 - [UVa 11629 - Ballot evaluation](#)
- String parsing
 - [UVa 1200 - A DP Problem](#)
- Priority queue
 - [UVa 1203 - Argus](#)
- Binary Manipulation
 - [UVa 11532 - Simple Adjacency Maximization](#)
- Binary Search
 - [UVa 12190 - Electric Bill](#)
 - [UVa 12192 - Grapevine](#)
 - [UVa 1215 - String Cutting](#)
- Segment Tree
 - [UVa 1232 - SKYLINE](#)
- Sort
 - [UVa 11157 - Dynamic Frog](#)
 - [UVa 12189 - Dinner Hall](#)
- Greed
 - [UVa 12172 - Matchsticks](#)
- Linked List
 - [UVa 245 - Uncompress](#)
- Union-Find
 - [UVa 11503 - Virtual Friends](#)
 - [UVa 10158 - War](#)
 - [UVa 11966 - Galactic Bonding](#)
- Ad hoc
 - [UVa 11494 - Queen](#)
 - [UVa 11597 - Spanning Subtree](#)
 - [UVa 12148 - Electricity](#)
 - [UVa 12195 - Jingle Composing](#)
 - [UVa 12155 - ASCII Diamondi](#)
 - [UVa 12196 - Klingon Levels](#)

misc/245.cpp

```
1  | #include <iostream>
2  | #include <sstream>
3  | #include <string>
4  | #include <climits>
5  | #include <cstring>
6  | #include <cstdio>
7  | #include <list>
8  | #define MAX 1000
9  | using namespace std;
10 |
11 | list<string> W;
12 | stringstream sin;
13 | int curnum=0;
14 | bool word=false, number=false;
15 |
16 | void finishWord() {
17 |     W.push_back(sin.str());
18 |     sin.str("");
19 |     word = false;
20 | }
21 |
22 | void finishNumber() {
23 |     list<string>::iterator it = W.end();
24 |     while(curnum--)
25 |         it--;
26 |
27 |     cout << *it;
28 |     W.push_back(*it);
29 |     W.erase(it);
30 |
31 |     curnum = 0;
32 |     number = false;
33 | }
34 |
35 | int main() {
36 |     string s;
37 |
38 |     while(getline(cin, s), s!="") {
39 |         for(int i=0; i<s.size(); i++) {
40 |             char c = s[i];
41 |             if (c >= 'a' && c <= 'z' || c >= 'A' && c <= 'Z') {
42 |                 sin << c;
43 |                 word = true;
44 |             } else if (word) finishWord();
45 |
46 |             if (c >= '0' && c <= '9') {
47 |                 curnum *= 10; curnum += c-'0';
48 |                 number = true;
49 |             } else if (number) finishNumber();
50 |
51 |             if (!number)
52 |                 cout << c;
53 |         }
54 |         if (word) finishWord();
55 |         if (number) finishNumber();
56 |
57 |         cout << endl;
58 |     }
59 | }
```

misc/1200.cpp

```
1  | #include <iostream>
2  | #include <string>
3  | #include <cmath>
```



```

4  using namespace std;
5
6  int getSign(string& s, int &i) {
7      if (s[i] == '+') { i++; return 1; }
8      if (s[i] == '-') { i++; return -1; }
9      return 1;
10 }
11
12 int getNumber(string& s, int& i, bool& got) {
13     int result = 0;
14     while(s[i] >='0' && s[i] <= '9') {
15         result = result*10 + (s[i]-'0');
16         i++;
17         got = true;
18     }
19     return result;
20 }
21
22 bool getX(string& s, int& i) {
23     return i<s.size() && s[i] == 'x' && ++i;
24 }
25
26 bool willChange(string& s, int& i) {
27     return i<s.size() && s[i] == '=' && ++i;
28 }
29
30
31 int main()
32 {
33     int t;
34     string s;
35     cin >> t;
36
37     while(cin >> s) {
38         int i=0, A=0, B=0, masterSign = 1;
39         while(i<s.size()) {
40             int sign = getSign(s, i);
41             bool got = false;
42             int number = getNumber(s, i, got);
43             bool isX = getX(s, i);
44             if (isX && !got) number = 1;
45             // cout << masterSign << " " << sign << " " << number << " " << isX << endl;
46
47             if (isX)
48                 B += -1*masterSign*sign*number;
49             else
50                 A += masterSign*sign*number;
51             if (willChange(s, i)) masterSign *= -1;
52         }
53         if (A==0 && B==0) {
54             cout << "IDENTITY" << endl;
55         } else if (B==0) {
56             cout << "IMPOSSIBLE" << endl;
57         } else {
58             cout << (int)floor(((double)A/B))<< endl;
59         }
60
61     }
62
63     return 0;
64 }

```

misc/1203.cpp

```

1  #include<cstdio>
2  #include<iostream>
3  #include<queue>
4  #include<string>

```

```

5  using namespace std;
6
7  #define SZ 3200
8
9  struct Item{
10     int p, q, b;
11
12     Item() {}
13     Item(int q, int p) : p(p), q(q), b(p) {}
14     Item(int q, int p, int b) : p(p), q(q), b(b) {}
15
16     inline bool operator < (const Item &d) const{
17         if(this->p==d.p) return d.q<this->q;
18         return this->p>d.p;
19     }
20
21     Item next() {
22         return Item(q, p+b, b);
23     }
24 };
25
26 priority_queue<Item> Q;
27 int q, p;
28
29 int main(void) {
30     string s;
31     int q, p, k;
32
33     while(cin >> s, s!="#") {
34         cin >> q >> p;
35         Q.push(Item(q, p));
36     }
37
38     cin >> k;
39     for(int i=0; i<k; ++i) {
40         Item item = Q.top(); Q.pop();
41         cout << item.q << endl;
42         Q.push(item.next());
43     }
44
45     return 0;
46 }
47

```

misc/1215.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <set>
4  using namespace std;
5
6  int T[10001][26];
7  int C[1001];
8  set<int> K;
9
10 int main() {
11     int t; cin >> t; t=0;
12     int k; string s;
13     while(cin >> k) {
14         K.clear();
15         for(int i=0; i<k; i++)
16             cin >> C[i];
17
18         cin >> s;
19         for(int i=1; i<=s.size(); i++)
20             for(int j=0; j<26; j++)
21                 T[i][j] = T[i-1][j] + (s[i-1] == j+'a');
22

```

```

23
24     K.insert(0);
25     K.insert(s.size());
26
27     int total = 0;
28     for(int i=0; i<k; i++) {
29         int mid = C[i];
30         set<int>::iterator it = K.lower_bound(mid);
31         int hi = *it; it--;
32         int lo = *it;
33
34         for(int j=0; j<26; j++) {
35             int sidea = T[mid][j]-T[lo][j];
36             int sideb = T[hi][j]-T[mid][j];
37
38             if (sidea>0 ^ sideb>0) total++;
39         }
40         K.insert(mid);
41     }
42     cout << total << endl;
43 }
44 }

```

misc/1232.cpp

```

1  #include <iostream>
2  #include <string>
3  #include <set>
4  using namespace std;
5
6  struct Node {
7      int a, b, h;
8      bool leaf;
9      Node() {}
10     Node(int a, int b, int h, bool leaf=true) : a(a), b(b), h(h), leaf(leaf) {}
11 };
12
13 Node H[5000005];
14 inline int left(int i) { return 2*i; }
15 inline int right(int i) { return 2*i+1; }
16
17 inline void cut(int v, int x) {
18     H[left(v)] = Node(H[v].a, x, H[v].h);
19     H[right(v)] = Node(x, H[v].b, H[v].h);
20     H[v].leaf = false;
21 }
22
23 int dfs(int v, int a, int b, int h) {
24     a = max(a, H[v].a);
25     b = min(b, H[v].b);
26     if (b<=a) return 0;
27
28     if (!H[v].leaf)
29         return dfs(left(v), a, b, h) + dfs(right(v), a, b, h);
30
31     if (H[v].h > h) return 0;
32     if (H[v].a < a) return cut(v, a), dfs(v, a, b, h);
33     if (b < H[v].b) return cut(v, b), dfs(v, a, b, h);
34
35     H[v].h = h;
36     return b-a;
37 }
38
39 int main() {
40     int n, t; cin >> t; t=0;
41     while(cin >> n, n) {
42         H[1] = Node(0, 100000, 0);
43     }

```

```

44 |         int sum = 0;
45 |         while(n--) {
46 |             int a, b, h;
47 |             cin >> a >> b >> h;
48 |             sum += dfs(1, a, b, h);
49 |         }
50 |         cout << sum << endl;
51 |     }
52 | }
53 | }

```

misc/10158.cpp

```

1 | #include <iostream>
2 | #include <map>
3 | #include <string>
4 | #include <cstring>
5 | #include <algorithm>
6 | using namespace std;
7 |
8 | int P[20000];
9 |
10 | inline int enemy(int v) { return v+10000; }
11 |
12 | inline int findset(int v) {
13 |     if (P[v] != -1 && P[v] != v)
14 |         return P[v] = findset(P[v]);
15 |     return v;
16 | }
17 |
18 | inline int unionset(int x, int y) {
19 |     int a = findset(x), b = findset(y);
20 |     if (a<b) swap(a,b);
21 |     P[b] = a;
22 | }
23 |
24 | int main() {
25 |     memset(P, -1, sizeof(P));
26 |     int n, c, x, y;
27 |     cin >> n;
28 |     while(cin >> c >> x >> y, c|x|y) {
29 |         if (c==1) {
30 |             if (findset(x) == findset(enemy(y))) { cout << -1 << endl; continue; }
31 |             unionset(x, y);
32 |             unionset(enemy(x), enemy(y));
33 |         } else if (c==2) {
34 |             if (findset(x) == findset(y)) { cout << -1 << endl; continue; }
35 |             unionset(x, enemy(y));
36 |             unionset(enemy(x), y);
37 |         } else if (c==3) {
38 |             cout << (findset(x) == findset(y)) << endl;
39 |         } else if (c==4) {
40 |             cout << (findset(x) == findset(enemy(y))) << endl;
41 |         }
42 |     }
43 | }

```

misc/10420.cpp

```

1 | #include <iostream>
2 | #include <string>
3 | #include <cstring>
4 | #include <cmath>
5 | #include <map>
6 | #define MAX 105
7 | using namespace std;

```

```

8
9 map<string, int> women;
10 int main() {
11     int n;
12     string s;
13     cin >> n;
14     while(n-->0) {
15         cin >> s;
16         women[s]++;
17         getline(cin, s);
18     }
19
20     for(map<string, int>::const_iterator it = women.begin(); it != women.end(); it++) {
21         cout << it->first << " " << it->second << endl;
22     }
23
24     return 0;
25 }

```

misc/11157.cpp

```

1 #include <iostream>
2 #include <algorithm>
3 #include <vector>
4 using namespace std;
5
6 vector<int> V;
7
8 int main() {
9     int t, n, d;
10    cin >> t; t=0;
11
12    while(cin >> n >> d) {
13        char a; int b;
14        V.clear();
15        V.push_back(0);
16        V.push_back(d);
17        for(int i=0; i<n; i++) {
18            cin >> a; cin.ignore(); cin >> b;
19            V.push_back(b);
20            if (a=='B')
21                V.push_back(b);
22        }
23        sort(V.begin(), V.end());
24
25        int maxx = 0;
26        for(int i=3; i<V.size(); i+=2)
27            maxx = max(maxx, V[i]-V[i-2]);
28
29        for(int i=2; i<V.size(); i+=2)
30            maxx = max(maxx, V[i]-V[i-2]);
31
32        cout << "Case " << ++t << ": " << maxx << endl;
33    }
34
35 }

```

misc/11494.cpp

```

1 #include <iostream>
2 #include <cstring>
3 #include <iomanip>
4 using namespace std;
5
6 int main() {
7     int x, y, a, b;

```

```

8 |     while(cin >> x >> y >> a >> b, x|y|a|b) {
9 |         if (x==a && y==b)
10 |             cout << 0 << endl;
11 |         else if (x==a || y==b || x+y == a+b || x-y==a-b)
12 |             cout << 1 << endl;
13 |         else
14 |             cout << 2 << endl;
15 |     }
16 |     return 0;
17 | }

```

misc/11503.cpp

```

1 | #include <iostream>
2 | #include <map>
3 | #include <string>
4 | #include <cstring>
5 | #include <algorithm>
6 | using namespace std;
7 |
8 | int P[200001], C[200001];
9 | map<string, int> M;
10 |
11 | int parent(int v) {
12 |     if (P[v] != v) {
13 |         int p = P[v] = parent(P[v]);
14 |         C[v] = C[p];
15 |         return p;
16 |     } else {
17 |         return v;
18 |     }
19 | }
20 |
21 | int person(string& s) {
22 |     if (M.find(s) != M.end())
23 |         return M[s];
24 |     else {
25 |         int r = M[s] = M.size();
26 |         C[r] = 1; P[r] = r;
27 |         return r;
28 |     }
29 | }
30 |
31 | int main() {
32 |     int t; cin >> t; t=0;
33 |     int n;
34 |
35 |     while(cin >> n) {
36 |         M.clear();
37 |         while(n--) {
38 |             string p, q;
39 |             cin >> p >> q;
40 |             int a = person(p), b=person(q);
41 |             int pa = parent(a), pb=parent(b);
42 |             if (pa==pb) {
43 |                 cout << C[pa] << endl;
44 |                 continue;
45 |             }
46 |             if (pa < pb) swap(pa, pb);
47 |
48 |             P[pb] = pa;
49 |             cout << (C[pa]+=C[pb]) << endl;
50 |         }
51 |     }
52 | }
53 |

```

misc/11532.cpp

```
1 | #include <iostream>
2 | #include <cstring>
3 | #include <iomanip>
4 | using namespace std;
5 |
6 | long long T[51][51];
7 |
8 | int main() {
9 |     for(int i=1;i<=50;i++) {
10 |         for(int j=0;i+j<=50;j++) {
11 |             int p=i, q=j;
12 |             long long n = 0L;
13 |             if (p%2!=0 && p/2<q) {
14 |                 n = 1;
15 |                 p--;
16 |             }
17 |
18 |             for(;p>1;p-=2) {
19 |                 if (q>0)
20 |                     n = (n<<3) | 5L;
21 |                 else
22 |                     n = (n<<2) | 3L;
23 |                 q--;
24 |             }
25 |
26 |             if (p==1) n = (n<<1) | 1L;
27 |             T[i][j] = n;
28 |         }
29 |     }
30 |
31 |
32 |     int t, p, q;
33 |     cin >> t;
34 |     while(cin >> p >> q) {
35 |         cout << T[p][q] << endl;
36 |     }
37 |
38 |     return 0;
39 | }
```

misc/11597.cpp

```
1 | #include <iostream>
2 | using namespace std;
3 |
4 | int main() {
5 |     int n, t=0;
6 |     while(cin >> n, t++, n) {
7 |         cout << "Case " << t << ": " << n/2 << endl;
8 |     }
9 |
10 |     return 0;
11 | }
```

misc/11629.cpp

```
1 | #include <iostream>
2 | #include <map>
3 | #include <cstring>
4 | using namespace std;
5 |
6 | map<string, int> P;
7 |
```

```

8  int main() {
9      int n, g;
10     while(cin >> n >> g) {
11         P.clear();
12         for(int i=0; i<n; i++) {
13             string s; int a, b;
14             cin >> s >> a; cin.get(); cin >> b;
15             P[s] = a*10+b;
16         }
17
18         for(int i=1; i<=g; i++) {
19             string s = "+"; int d=0; int r;
20
21             while(s=="+") {
22                 cin >> s;
23                 d += P[s];
24                 cin >> s;
25             }
26             cin >> r; r*=10;
27
28             bool result;
29             if ( s=="<") result = d < r;
30             if ( s=="<=") result = d <= r;
31             if ( s==">") result = d > r;
32             if ( s==">=") result = d >= r;
33             if ( s=="=") result = d == r;
34             cout << "Guess #" << i << " was " << (result?"correct":"incorrect") << "." << endl;
35         }
36     }
37 }
38 }

```

misc/11966.cpp

```

1  #include <iostream>
2  #include <map>
3  #include <string>
4  #include <cstring>
5  #include <algorithm>
6  #include <cmath>
7  using namespace std;
8
9  int P[1000];
10 double X[1000], Y[1000];
11
12 inline int findset(int v) {
13     if (P[v] == v) return v;
14     return P[v] = findset(P[v]);
15 }
16
17 inline bool unionset(int x, int y) {
18     int a = findset(x), b = findset(y);
19     if (a==b) return false;
20     P[b] = a;
21     return true;
22 }
23
24 inline double dist(int a, int b) {
25     return pow(X[a]-X[b], 2.0)+pow(Y[a]-Y[b], 2.0);
26 }
27
28 int main() {
29     int t; cin >> t; t=0;
30
31     int n; double d;
32     while(cin >> n >> d) {
33         for(int i=0; i<n; i++) P[i] = i;
34

```



```

35     int sets = n;
36     for(int i=0; i<n; i++) {
37         cin >> X[i] >> Y[i];
38         for(int j=0; j<i; j++)
39             if (dist(i,j)<=d*d && unionset(i, j))
40                 sets--;
41     }
42
43     cout << "Case " << ++t << ": " << sets << endl;
44 }
45 }

```

misc/12148.cpp

```

1  #include <iostream>
2  using namespace std;
3
4  int M[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
5
6  bool oneday(int ad, int am, int ay, int bd, int bm, int by) {
7      if (--bd == 0) {
8          if (--bm == 0) {
9              --by;
10             bm=12;
11         }
12
13         bd = M[bm-1];
14
15         bool isleap = (by%4==0 && (by%100!=0 || by%400==0));
16         if (bm==2 && isleap) bd=29;
17     }
18     return ad==bd && am==bm && ay==by;
19 }
20
21
22 int main() {
23     int n, ad=0, am=0, ay=0, ac=0;
24     while(cin >> n, n) {
25         int sum = 0, count=0;
26         while(n--) {
27             int bd, bm, by, bc;
28             cin >> bd >> bm >> by >> bc;
29             if (oneday(ad, am, ay, bd, bm, by)) {
30                 sum += bc-ac; count++;
31             }
32             ad = bd; am = bm; ay = by; ac = bc;
33         }
34         cout << count << " " << sum << endl;
35     }
36
37     return 0;
38 }

```

misc/12155.cpp

```

1  #include <iostream>
2  using namespace std;
3
4  inline int abs(int n) { return n>0?n:-n; }
5
6  inline char charAt(int n, int x, int y) {
7      x%=n*2-1; y%=n*2-1;
8      int dist = abs(n-x-1)+abs(n-y-1);
9      if (dist < n)
10         return (char)(dist%26+'a');
11     else

```

```

12         return '.';
13     }
14
15     int main() {
16         int n, ax, ay, bx, by, t=0;
17         while(cin >> n, n) {
18             cin >> ax >> ay >> bx >> by;
19             cout << "Case " << ++t << ":" << endl;
20             for(int i=ax; i<=bx; i++) {
21                 for(int j=ay; j<=by; j++) {
22                     cout << charAt(n, i, j);
23                 }
24                 cout << endl;
25             }
26         }
27     }
28 }

```

misc/12172.cpp

```

1  #include <iostream>
2  #include <cstring>
3  #include <cmath>
4  #define MAX 101
5  using namespace std;
6
7  void printMax(int n) {
8      if (n&1) { cout << "7"; n-=3; }
9      for(;n-=2) cout << "1";
10 }
11
12 void printMin(int n) {
13     switch(n) {
14         case 2: cout << "1"; return;
15         case 3: cout << "7"; return;
16         case 4: cout << "4"; return;
17         case 5: cout << "2"; return;
18         case 6: cout << "6"; return;
19     }
20
21     switch(n%7) {
22         case 1: cout << "10"; n-=8; break;
23         case 2: cout << "1"; n-=2; break;
24         case 3:
25             if (n==10) {
26                 cout << "22"; n-= 10;
27             } else{
28                 cout << "200"; n-=17;
29             }
30             break;
31         case 4: cout << "20"; n-= 11; break;
32         case 5: cout << "2"; n-= 5; break;
33         case 6: cout << "6"; n-= 6; break;
34     }
35     for(;n-=7) cout << "8";
36 }
37
38 int main() {
39
40
41     int n;
42     int t; cin >> t; t=0;
43
44     while(cin >> n) {
45         printMin(n);
46         cout << " ";
47         printMax(n);
48         cout << endl;

```

```

49 |     }
50 |
51 |     return 0;
52 | }

```

misc/12189.cpp

```

1 | #include <iostream>
2 | #include <vector>
3 | #include <algorithm>
4 | using namespace std;
5 |
6 | struct Event {
7 |     int s; char t;
8 |     Event() {}
9 |     Event(int s, char t) : s(s), t(t) {}
10 |     int entry() { return t=='E'?1:0; }
11 |     int exit() { return t=='X'?1:0; }
12 |     int unknown() { return t=='?'?1:0; }
13 | };
14 |
15 | bool compare(const Event& a, const Event& b) {
16 |     return a.s < b.s;
17 | }
18 |
19 | vector<Event> V;
20 |
21 | int main() {
22 |     int n;
23 |     while(cin >> n, n) {
24 |         int entries=0, exits=0, unknowns=0;
25 |         int a, b, c; char t;
26 |         V.clear();
27 |         for(int i=0; i<n; i++) {
28 |             cin >> a >> t >> b >> t >> c >> t;
29 |             Event e = Event(a*60+b*60+c, t);
30 |             entries += e.entry();
31 |             exits += e.exit();
32 |             unknowns += e.unknown();
33 |             V.push_back(e);
34 |         }
35 |         sort(V.begin(), V.end(), compare);
36 |
37 |         int maxEntries = (unknowns-(entries-exits))/2;
38 |         int maxx = 0, current=0;
39 |         for(int i=0; i<V.size(); i++) {
40 |             if (V[i].entry()) current++;
41 |             if (V[i].exit()) current--;
42 |             if (V[i].unknown()) {
43 |                 if (maxEntries) { current++; maxEntries--; }
44 |                 else { current--; }
45 |             }
46 |             maxx = max(maxx, current);
47 |         }
48 |         cout << maxx << endl;
49 |     }
50 | }

```

misc/12190.cpp

```

1 | #include <iostream>
2 | using namespace std;
3 |
4 | int C(int price) {
5 |     int cons = 0;
6 |     cons += min(max(0, price/2), 100); price -= 2*100;

```

```

7   cons += min(max(0, price/3), 9900); price -= 3*9900;
8   cons += min(max(0, price/5), 990000); price -= 5*990000;
9   cons += max(0, price/7);
10  return cons;
11 }
12
13 int V(int cons) {
14     int price = 0;
15     price += min(max(0, cons*2), 2*100); cons -= 100;
16     price += min(max(0, cons*3), 3*9900); cons -= 9900;
17     price += min(max(0, cons*5), 5*990000); cons -= 990000;
18     price += max(0, cons*7);
19     return price;
20 }
21
22 int main() {
23     int a, b;
24     while(cin >> a >> b, a|b) {
25         int total = C(a);
26         int begin = 0, end = total;
27         int answer = 0;
28         while(begin < end) {
29             int mine = (begin+end)/2;
30             int diff = V(total-mine)-V(mine);
31             if (diff > b)
32                 begin = mine;
33             else if (diff < b)
34                 end = mine;
35             else { answer = mine; break; }
36         }
37
38         cout << V(answer) << endl;
39     }
40
41     return 0;
42 }

```

misc/12192.cpp

```

1   #include <iostream>
2   #include <cstring>
3   #include <vector>
4   #include <algorithm>
5   using namespace std;
6
7   int T[1001][501];
8   int S[1001];
9
10  int main() {
11      int n, m, q;
12      while(cin >> n >> m, n|m) {
13          memset(S, 0, (m+n)*sizeof(int));
14
15          for(int i=0; i<n; i++)
16              for(int j=0; j<m; j++)
17                  cin >> T[i-j+m][S[i-j+m]++];
18
19          cin >> q;
20          while(q-- > 0) {
21              int L, U;
22              cin >> L >> U;
23              int maxx = 0;
24              for(int i=0; i<m+n; i++) {
25                  int a = lower_bound(T[i], T[i]+S[i], L) - T[i];
26                  int b = upper_bound(T[i], T[i]+S[i], U) - T[i];
27                  maxx = max(maxx, b-a);
28              }
29              cout << maxx << endl;

```

```

30 |         }
31 |
32 |         cout << "-" << endl;
33 |     }
34 | }

```

misc/12195.cpp

```

1 | #include <iostream>
2 | #include <string>
3 | using namespace std;
4 |
5 | int duration(char c) {
6 |     switch(c) {
7 |         case 'W': return 64;
8 |         case 'H': return 32;
9 |         case 'Q': return 16;
10 |        case 'E': return 8;
11 |        case 'S': return 4;
12 |        case 'T': return 2;
13 |        case 'X': return 1;
14 |    }
15 | }
16 |
17 | int main() {
18 |     string s;
19 |     while(cin >> s, s!="") {
20 |         int d=0, r=0;
21 |         for(int i=1; i<s.size(); i++) {
22 |             if (s[i] == '/') {
23 |                 if (d==64) r++;
24 |                 d = 0;
25 |                 continue;
26 |             }
27 |             d+=duration(s[i]);
28 |         }
29 |         cout << r << endl;
30 |     }
31 |
32 |     return 0;
33 | }

```

misc/12196.cpp

```

1 | #include <iostream>
2 | #include <climits>
3 | #include <cstring>
4 | using namespace std;
5 |
6 | int T[10001][1001];
7 | int N[10001];
8 |
9 | inline long abs(long n) { return n>0?n:-n;}
10 |
11 | int main() {
12 |     int n, tmp;
13 |     while(cin >> n, n) {
14 |         memset(T, 0, n*sizeof(T[0]));
15 |         for(int i=0; i<n; i++) {
16 |             cin >> N[i];
17 |             for(int j=0; j<N[i]; j++) {
18 |                 cin >> tmp;
19 |                 T[i][tmp]++;
20 |             }
21 |             for(int j=1; j<=1000; j++)
22 |                 T[i][j] += T[i][j-1];

```

```

23     }
24
25     long minn = INT_MAX;
26     for(int t=0;t<=1000;t++) {
27         long sum=0;
28         for(int i=0; i<n; i++) {
29             sum += abs(N[i] - 2*T[i][t]);
30         }
31         minn = min(minn, sum);
32     }
33     cout << minn << endl;
34 }
35 }

```