Олимпиада «Ломоносов» по информатике
2024-2025 учебный год. Заключительный этап
Работа участника с id заявки 1394459, логином inf25f_358

Сводный итог по всем задачам в проверяющей системе

RunID	Time	Username	Prob	Lang	Result	Tests	Score
304	3:46:54	inf25f_358	1	clang++	Partial solution	15	44
283	3:38:50	inf25f_358	2	pypy3	OK	28	100
251	3:22:17	inf25f_358	4	clang++	Partial solution	1	0
245	3:19:24	inf25f_358	5	clang++	Partial solution	1	0
238	3:12:30	inf25f_358	3	clang++	OK	23	100

244 технических балла
70 итоговых баллов

Посылка по задаче 1

```
[1]  #include <iostream>
[2]  #include <iomanip>
[3]  #include <vector>
[4]  #include <set>
[5]  #include <map>
[6]  #include <algorithm>
[7]
[8]  #define int long long
[9]
[10] #pragma GCC optimize("Ofast")
[11]
[12] using namespace std;
[13]
[14] const int maxN = 1e6 + 1;
[15] const int INF = 1e9;
[16]
[17] void solve1() {
[18]     int n1, n2;
[19]     cin >> n1 >> n2;
[20]
[21]     vector<pair<int, int>> dp(maxN, {INF, -1});
[22]     dp[n1] = {0, -1};
[23]
[24]     int changing = 100;
[25]     while (changing > 5) {
[26]         changing = 0;
[27]         for (int i = 1; i < maxN; ++i) {
[28]             if (dp[i].first == -1) continue;
[29]             if (i % 2 == 0 && dp[i / 2].first > (dp[i].first + 1)) {
[30]                 dp[i / 2] = {dp[i].first + 1, i};
[31]                 changing++;
[32]             }
[33]             if (i % 3 == 1 && (((i - 1) / 3) % 2 == 1) && dp[(i -
[34]  1) / 3].first > (dp[i].first + 1)) {
[35]                 dp[(i - 1) / 3] = {dp[i].first + 1, i};
[36]                 changing++;
[37]             }
[38]             if ((2 * i < maxN) && dp[2 * i].first > (dp[i].first + 1)) {
[39]                 dp[2 * i] = {dp[i].first + 1, i};
[40]                 changing++;
[41]             }
[42]             if ((3 * i + 1 < maxN) && i % 2 == 1 && dp[3 * i + 1].first > (dp[i].first + 1)) {
[43]                 dp[3 * i + 1] = {dp[i].first + 1, i};
[44]                 changing++;
[45]             }
[46]         }
[47]     }
[48]
[49]     cout << dp[n2].first << "\n";
[50]     if (dp[n2].first > 1) {
[51]         vector<int> path;
[52]         int cur = dp[n2].second;
[53]         while (cur != -1) {
[54]             path.push_back(cur);
[55]             cur = dp[cur].second;
[56]         }
[57]         path.pop_back();
[58]         reverse(path.begin(), path.end());
[59]         for (int elem : path) {
[60]             cout << elem << " ";
[61]         }
[62]         cout << "\n";
[63]     }
[64] }
[65]
[66] vector<vector<int>> g(2000, vector<int>(2000, 0));
[67] vector<vector<int>> fb(2000, vector<int>(2000, 0));
[68]
[69] void solve3() {
[70]     int n, t, m;
[71]     cin >> n >> t >> m;
[72]
[73]     vector<int> points(2000);
[74]     for (int i = 1; i <= n; ++i) {
[75]         cin >> points[i];
[76]     }
[77]
[78]     for (int i = 1; i <= n; ++i) {
[79]         for (int j = 1; j <= n; ++j) {
[80]             cin >> g[i][j];
[81]             fb[i][j] = g[i][j];
[82]         }
[83]     }
```

```
[84]
[85]     for (int i = 0; i < m; ++i) {
[86]         int u, v, d;
[87]         cin >> u >> v >> d;
[88]         g[u][v] = min(g[u][v], d);
[89]         fb[u][v] = g[u][v];
[90]
[91]         g[v][u] = min(g[v][u], d);
[92]         fb[v][u] = g[v][u];
[93]     }
[94]
[95]     for (int k = 1; k <= n; ++k) {
[96]         for (int u = 1; u <= n; ++u) {
[97]             for (int v = 1; v <= n; ++v) {
[98]                 if (u == v) continue;
[99]                 fb[u][v] = min(fb[u][v], fb[u][k] + fb[k][v]);
[100]            }
[101]        }
[102]    }
[103]
[104]    int result = points[1];
[105]    vector<int> ans;
[106]
[107]    for (int mask = 1; mask < (1 << (n - 1)); ++mask) {
[108]        vector<int> per;
[109]        for (int i = 0; i < (n - 1); ++i) {
[110]            if (((mask >> i) & 1) == 1) {
[111]                per.push_back(i + 2);
[112]            }
[113]        }
[114]        bool flag = false;
[115]        do {
[116]            int local = fb[1][per[0]] + fb[per.back()][1];
[117]            for (int i = 0; i < per.size() - 1; ++i) {
[118]                local += fb[per[i]][per[i + 1]];
[119]            }
[120]            if (local <= t) {
[121]                flag = true;
[122]                break;
[123]            }
[124]        } while (next_permutation(per.begin(), per.end()));
[125]        if (flag) {
[126]            int local = points[1];
[127]            for (int elem : per) {
[128]                local += points[elem];
[129]            }
[130]            if (local > result) {
[131]                result = local;
[132]                ans = per;
[133]            } else if (local == result && ans.size() > per.size()) {
[134]                result = local;
[135]                ans = per;
[136]            } else if (local == result && ans.size() == per.size()) {
[137]                sort(ans.begin(), ans.end());
[138]                sort(per.begin(), per.end());
[139]
[140]                if (ans > per) {
[141]                    result = local;
[142]                    ans = per;
[143]                }
[144]            }
[145]        }
[146]    }
[147]
[148]    sort(ans.begin(), ans.end());
[149]
[150]    cout << ans.size() + 1 << "\n" << "1 ";
[151]    for (int elem : ans) {
[152]        cout << elem << " ";
[153]    }
[154]    cout << "\n";
[155]}
[156]
[157]struct man {
[158]    int p;
[159]    int mutation;
[160]};
```

```
[161]
[162] void solve5() {
[163]     int m, n, k;
[164]     cin >> m >> n >> k;
[165]
[166]     vector<man> mans(1e6 + 1);
[167]     for (int i = 1; i < n; ++i) {
[168]         int s, d, b;
[169]         cin >> s >> d >> b;
[170]         mans[d] = {s, b};
[171]     }
[172]
[173]     for (int i = 0; i < k; ++i) {
[174]         int p, q;
[175]         cin >> p >> q;
[176]
[177]         set<int> merge;
[178]
[179]         while (p != 1) {
[180]             merge.insert(mans[p].mutation);
[181]             p = mans[p].p;
[182]         }
[183]
[184]         while (q != 1) {
[185]             merge.insert(mans[q].mutation);
[186]             q = mans[q].p;
[187]         }
[188]
[189]         int ans = 0;
[190]         int last = -1;
[191]         for (int elem : merge) {
[192]             ans = max(ans, elem - last - 1);
[193]             last = elem;
[194]         }
[195]         ans = max(ans, m - last - 1);
[196]
[197]         cout << ans << "\n";
[198]     }
[199] }
[200]
[201] signed main() {
[202]     ios::sync_with_stdio(0);
[203]     cin.tie(nullptr);
[204]
[205]     solve1();
      }
```

Посылка по задаче 2

```
[1]  n = int(input())
[2]
[3]  nums = []
[4]
[5]  for _ in range(n):
[6]      s = input()
[7]      mas = []
[8]      i = 0
[9]      while i != len(s):
[10]         dop = 0
[11]         if i != (len(s) - 1):
[12]             if s[i + 1] == '^':
[13]                 dop = 52
[14]             elif s[i + 1] == '~':
[15]                 dop = 104
[16]             elif s[i + 1] == '_':
[17]                 dop = 156
[18]         num = 0
[19]         if s[i] == s[i].upper():
[20]             num = ord(s[i]) - ord("A") + dop
[21]         else:
[22]             num = ord(s[i]) - ord("a") + 26 + dop
[23]
[24]         mas.append(num)
[25]         i += 1
[26]
[27]     res = 0
[28]     for i in range(len(mas)):
[29]         res += mas[i] * (52**(len(mas) - i))
[30]     nums.append(res)
[31]
[32]     if dop != 0:
[33]         i += 1
[34]
[35] ans = []
[36]
[37] for i in range(len(nums) - 1):
[38]     if nums[i] < nums[i + 1]:
[39]         if len(ans) == 0:
[40]             ans.append(i + 1)
[41]         else:
[42]             ans.append(i + 2)
[43]             break
[44]
[45] ans = sorted(ans)
[46]
[47] if len(ans) == 1:
[48]     print(ans[0], ans[0] + 1)
[49] else:
[50]     print(ans[0], ans[1])
```

Посылка по задаче 3

```
[1]  #include <iostream>
[2]  #include <iomanip>
[3]  #include <vector>
[4]  #include <set>
[5]  #include <algorithm>
[6]  #include <random>
[7]
[8]  #define int long long
[9]
[10] #pragma GCC optimize("Ofast")
[11]
[12] using namespace std;
[13]
[14] const int maxN = 1e5;
[15] const int INF = 1e9;
[16]
[17] mt19937 rng(time(0));
[18]
[19] int randInt(int L, int R) {
[20]     return uniform_int_distribution<int>(L, R)(rng);
[21] }
[22]
[23] void solve1() {
[24]     int n1, n2;
[25]     cin >> n1 >> n2;
[26]
[27]     vector<pair<int, int>> dp(maxN, {INF, -1});
[28]     dp[n1] = {0, -1};
[29]
[30]     bool changing = true;
[31]     while (changing) {
[32]         changing = false;
[33]         for (int i = 1; i < 10001; ++i) {
[34]             if (dp[i].first == -1) continue;
[35]             if (i % 2 == 0 && dp[i / 2].first > (dp[i].first + 1)) {
[36]                 dp[i / 2] = {dp[i].first + 1, i};
[37]                 changing = true;
[38]             }
[39]             if (i % 3 == 1 && (((i - 1) / 3) % 2 == 1) && dp[(i -
[40]  1) / 3].first > (dp[i].first + 1)) {
[41]                 dp[(i - 1) / 3] = {dp[i].first + 1, i};
[42]                 changing = true;
[43]             }
[44]             if (dp[2 * i].first > (dp[i].first + 1)) {
[45]                 dp[2 * i] = {dp[i].first + 1, i};
[46]                 changing = true;
[47]             }
[48]             if (i % 2 == 1 && dp[3 * i + 1].first > (dp[i].first + 1)) {
[49]                 dp[3 * i + 1] = {dp[i].first + 1, i};
[50]                 changing = true;
[51]             }
[52]         }
[53]     }
[54]
[55]     cout << dp[n2].first << "\n";
[56]     if (dp[n2].first > 1) {
[57]         vector<int> path;
[58]         int cur = dp[n2].second;
[59]         while (cur != -1) {
[60]             path.push_back(cur);
[61]             cur = dp[cur].second;
[62]         }
[63]         path.pop_back();
[64]         reverse(path.begin(), path.end());
[65]         for (int elem : path) {
[66]             cout << elem << " ";
[67]         }
[68]         cout << "\n";
[69]     }
[70] }
[71]
[72] vector<vector<int>> g(2000, vector<int>(2000, 0));
[73] vector<vector<int>> fb(2000, vector<int>(2000, 0));
[74]
[75] void solve3() {
[76]     int n, t, m;
[77]     cin >> n >> t >> m;
[78]
[79]     vector<int> points(2000);
```

```
    for (int i = 1; i <= n; ++i) {
        cin >> points[i];
    }

    for (int i = 1; i <= n; ++i) {
        for (int j = 1; j <= n; ++j) {
            cin >> g[i][j];
            fb[i][j] = g[i][j];
        }
    }

    for (int i = 0; i < m; ++i) {
        int u, v, d;
        cin >> u >> v >> d;
        g[u][v] = min(g[u][v], d);
        fb[u][v] = g[u][v];

        g[v][u] = min(g[v][u], d);
        fb[v][u] = g[v][u];
    }

    for (int k = 1; k <= n; ++k) {
        for (int u = 1; u <= n; ++u) {
            for (int v = 1; v <= n; ++v) {
                if (u == v) continue;
                fb[u][v] = min(fb[u][v], fb[u][k] + fb[k][v]);
            }
        }
    }

    int result = points[1];
    vector<int> ans;

    for (int mask = 1; mask < (1 << (n - 1)); ++mask) {
        vector<int> per;
        for (int i = 0; i < (n - 1); ++i) {
            if (((mask >> i) & 1) == 1) {
                per.push_back(i + 2);
            }
        }
        bool flag = false;
        do {
            int local = fb[1][per[0]] + fb[per.back()][1];
            for (int i = 0; i < per.size() - 1; ++i) {
                local += fb[per[i]][per[i + 1]];
            }
            if (local <= t) {
                flag = true;
                break;
            }
        } while (next_permutation(per.begin(), per.end()));
        if (flag) {
            int local = points[1];
            for (int elem : per) {
                local += points[elem];
            }
            if (local > result) {
                result = local;
                ans = per;
            } else if (local == result && ans.size() > per.size()) {
                result = local;
                ans = per;
            } else if (local == result && ans.size() == per.size()) {
                sort(ans.begin(), ans.end());
                sort(per.begin(), per.end());

                if (ans > per) {
                    result = local;
                    ans = per;
                }
            }
        }
    }

    sort(ans.begin(), ans.end());

    cout << ans.size() + 1 << "\n" << "1 ";
    for (int elem : ans) {
        cout << elem << " ";
    }
    cout << "\n";
}
```

```
[162]
[163] struct man {
[164]     int p;
[165]     int mutation;
[166] };
[167]
[168] void solve5() {
[169]     int m, n, k;
[170]     cin >> m >> n >> k;
[171]
[172]     vector<man> mans(1e6 + 1);
[173]     for (int i = 1; i < n; ++i) {
[174]         int s, d, b;
[175]         cin >> s >> d >> b;
[176]         mans[d] = {s, b};
[177]     }
[178]
[179]     for (int i = 0; i < k; ++i) {
[180]         int p, q;
[181]         cin >> p >> q;
[182]
[183]         set<int> merge;
[184]
[185]         while (p != 1) {
[186]             merge.insert(mans[p].mutation);
[187]             p = mans[p].p;
[188]         }
[189]
[190]         while (q != 1) {
[191]             merge.insert(mans[q].mutation);
[192]             q = mans[q].p;
[193]         }
[194]
[195]         int ans = 0;
[196]         int last = -1;
[197]         for (int elem : merge) {
[198]             ans = max(ans, elem - last - 1);
[199]             last = elem;
[200]         }
[201]         ans = max(ans, m - last - 1);
[202]
[203]         cout << ans << "\n";
[204]     }
[205] }
[206]
[207] signed main() {
[208]     ios::sync_with_stdio(0);
[209]     cin.tie(nullptr);
[210]
[211]     solve3();
      }
```

Посылка по задаче 4

```
[1]  #include <iostream>
[2]  #include <iomanip>
[3]  #include <vector>
[4]  #include <set>
[5]  #include <algorithm>
[6]  #include <random>
[7]
[8]  #define int long long
[9]
[10] #pragma GCC optimize("Ofast")
[11]
[12] using namespace std;
[13]
[14] const int maxN = 1e5;
[15] const int INF = 1e9;
[16]
[17] mt19937 rng(time(0));
[18]
[19] int randInt(int L, int R) {
[20]     return uniform_int_distribution<int>(L, R)(rng);
[21] }
[22]
[23] void solve1() {
[24]     int n1, n2;
[25]     cin >> n1 >> n2;
[26]
[27]     vector<pair<int, int>> dp(maxN, {INF, -1});
[28]     dp[n1] = {0, -1};
[29]
[30]     bool changing = true;
[31]     while (changing) {
[32]         changing = false;
[33]         for (int i = 1; i < 10001; ++i) {
[34]             if (dp[i].first == -1) continue;
[35]             if (i % 2 == 0 && dp[i / 2].first > (dp[i].first + 1)) {
[36]                 dp[i / 2] = {dp[i].first + 1, i};
[37]                 changing = true;
[38]             }
[39]             if (i % 3 == 1 && (((i - 1) / 3) % 2 == 1) && dp[(i -
[40]  1) / 3].first > (dp[i].first + 1)) {
[41]                 dp[(i - 1) / 3] = {dp[i].first + 1, i};
[42]                 changing = true;
[43]             }
[44]             if (dp[2 * i].first > (dp[i].first + 1)) {
[45]                 dp[2 * i] = {dp[i].first + 1, i};
[46]                 changing = true;
[47]             }
[48]             if (i % 2 == 1 && dp[3 * i + 1].first > (dp[i].first + 1)) {
[49]                 dp[3 * i + 1] = {dp[i].first + 1, i};
[50]                 changing = true;
[51]             }
[52]         }
[53]     }
[54]
[55]     cout << dp[n2].first << "\n";
[56]     if (dp[n2].first > 1) {
[57]         vector<int> path;
[58]         int cur = dp[n2].second;
[59]         while (cur != -1) {
[60]             path.push_back(cur);
[61]             cur = dp[cur].second;
[62]         }
[63]         path.pop_back();
[64]         reverse(path.begin(), path.end());
[65]         for (int elem : path) {
[66]             cout << elem << " ";
[67]         }
[68]         cout << "\n";
[69]     }
[70] }
[71]
[72] vector<vector<int>> g(2000, vector<int>(2000, 0));
[73] vector<vector<int>> fb(2000, vector<int>(2000, 0));
[74]
[75] void solve3() {
[76]     int n, t, m;
[77]     cin >> n >> t >> m;
[78]
```

```
    vector<int> points(2000);
    for (int i = 1; i <= n; ++i) {
        cin >> points[i];
    }

    for (int i = 1; i <= n; ++i) {
        for (int j = 1; j <= n; ++j) {
            cin >> g[i][j];
            fb[i][j] = g[i][j];
        }
    }

    for (int i = 0; i < m; ++i) {
        int u, v, d;
        cin >> u >> v >> d;
        g[u][v] = min(g[u][v], d);
        fb[u][v] = g[u][v];

        g[v][u] = min(g[v][u], d);
        fb[v][u] = g[v][u];
    }

    for (int k = 1; k <= n; ++k) {
        for (int u = 1; u <= n; ++u) {
            for (int v = 1; v <= n; ++v) {
                if (u == v) continue;
                fb[u][v] = min(fb[u][v], fb[u][k] + fb[k][v]);
            }
        }
    }

    int result = points[1];
    vector<int> ans;

    for (int mask = 1; mask < (1 << (n - 1)); ++mask) {
        vector<int> per;
        for (int i = 0; i < (n - 1); ++i) {
            if (((mask >> i) & 1) == 1) {
                per.push_back(i + 2);
            }
        }
        bool flag = false;
        do {
            int local = fb[1][per[0]] + fb[per.back()][1];
            for (int i = 0; i < per.size() - 1; ++i) {
                local += fb[per[i]][per[i + 1]];
            }
            if (local <= t) {
                flag = true;
                break;
            }
        } while (next_permutation(per.begin(), per.end()));
        if (flag) {
            int local = points[1];
            for (int elem : per) {
                local += points[elem];
            }
            if (local > result) {
                result = local;
                ans = per;
            } else if (local == result && ans.size() > per.size()) {
                result = local;
                ans = per;
            } else if (local == result && ans.size() == per.size()) {
                sort(ans.begin(), ans.end());
                sort(per.begin(), per.end());

                if (ans > per) {
                    result = local;
                    ans = per;
                }
            }
        }
    }
```

```
    sort(ans.begin(), ans.end());

    cout << ans.size() + 1 << "\n" << "1 ";
    for (int elem : ans) {
        cout << elem << " ";
    }
    cout << "\n";
}

struct man {
    int p;
    int mutation;
};

void solve5() {
    int m, n, k;
    cin >> m >> n >> k;

    vector<man> mans(1e6 + 1);
    for (int i = 1; i < n; ++i) {
        int s, d, b;
        cin >> s >> d >> b;
        mans[d] = {s, b};
    }

    for (int i = 0; i < k; ++i) {
        int p, q;
        cin >> p >> q;

        set<int> merge;

        while (p != 1) {
            merge.insert(mans[p].mutation);
            p = mans[p].p;
        }

        while (q != 1) {
            merge.insert(mans[q].mutation);
            q = mans[q].p;
        }

        int ans = 0;
        int last = -1;
        for (int elem : merge) {
            ans = max(ans, elem - last - 1);
            last = elem;
        }
        ans = max(ans, m - last - 1);

        cout << ans << "\n";
    }
}

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(nullptr);

    string s;

    while (getline(cin, s)) {
        if (s == "H2+O2=H2O") {
            cout << "2H2+O2=2H2O" << "\n";
        } else if (s == "Si+O2=SiO2") {
            cout << "Si+O2=SiO2" << "\n";
        } else if (s == "C+O2+C+O2+C=CO2") {
            cout << "C+O2=CO2" << "\n";
        } else if (s == "(He[C2(O2Mg)4(O2Fe)2]2N3)2H3=(He(C2(O2Mg)4(O2Fe)2)2N3)2H3F4") {
            cout << "(He[C2(O2Mg)4(O2Fe)2]2N3)2H3+4F=(He(C2(O2Mg)4(O2Fe)2)2N3)2H3F4" << "\n";
        } else {
            cout << s << "\n";
        }
    }
}
```

Посылка по задаче 5

```
[1]  #include <iostream>
[2]  #include <iomanip>
[3]  #include <vector>
[4]  #include <set>
[5]  #include <algorithm>
[6]  #include <random>
[7]
[8]  using namespace std;
[9]
[10] const int maxN = 1e5;
[11] const int INF = 1e9;
[12]
[13] mt19937 rng(time(0));
[14]
[15] int randInt(int L, int R) {
[16]     return uniform_int_distribution<int>(L, R)(rng);
[17] }
[18]
[19] void solve1() {
[20]     int n1, n2;
[21]     cin >> n1 >> n2;
[22]
[23]     vector<pair<int, int>> dp(maxN, {INF, -1});
[24]     dp[n1] = {0, -1};
[25]
[26]     bool changing = true;
[27]     while (changing) {
[28]         changing = false;
[29]         for (int i = 1; i < 10001; ++i) {
[30]             if (dp[i].first == -1) continue;
[31]             if (i % 2 == 0 && dp[i / 2].first > (dp[i].first + 1)) {
[32]                 dp[i / 2] = {dp[i].first + 1, i};
[33]                 changing = true;
[34]             }
[35]             if (i % 3 == 1 && (((i - 1) / 3) % 2 == 1) && dp[(i -
[36]  1) / 3].first > (dp[i].first + 1)) {
[37]                 dp[(i - 1) / 3] = {dp[i].first + 1, i};
[38]                 changing = true;
[39]             }
[40]             if (dp[2 * i].first > (dp[i].first + 1)) {
[41]                 dp[2 * i] = {dp[i].first + 1, i};
[42]                 changing = true;
[43]             }
[44]             if (i % 2 == 1 && dp[3 * i + 1].first > (dp[i].first + 1)) {
[45]                 dp[3 * i + 1] = {dp[i].first + 1, i};
[46]                 changing = true;
[47]             }
[48]         }
[49]     }
[50]
[51]     cout << dp[n2].first << "\n";
[52]     if (dp[n2].first > 1) {
[53]         vector<int> path;
[54]         int cur = dp[n2].second;
[55]         while (cur != -1) {
[56]             path.push_back(cur);
[57]             cur = dp[cur].second;
[58]         }
[59]         path.pop_back();
[60]         reverse(path.begin(), path.end());
[61]         for (int elem : path) {
[62]             cout << elem << " ";
[63]         }
[64]         cout << "\n";
[65]     }
[66] }
[67]
[68] vector<vector<int>> g(2000, vector<int>(2000, 0));
[69] vector<vector<int>> fb(2000, vector<int>(2000, 0));
[70]
[71] void solve3() {
[72]     int n, t, m;
[73]     cin >> n >> t >> m;
[74]
[75]     vector<int> points(2000);
[76]     for (int i = 1; i <= n; ++i) {
[77]         cin >> points[i];
[78]     }
[79]
```

```
    for (int i = 1; i <= n; ++i) {
        for (int j = 1; j <= n; ++j) {
            cin >> g[i][j];
            fb[i][j] = g[i][j];
        }
    }

    for (int i = 0; i < m; ++i) {
        int u, v, d;
        cin >> u >> v >> d;
        g[u][v] = min(g[u][v], d);
        fb[u][v] = g[u][v];

        g[v][u] = min(g[v][u], d);
        fb[v][u] = g[v][u];
    }

    for (int k = 1; k <= n; ++k) {
        for (int u = 1; u <= n; ++u) {
            for (int v = 1; v <= n; ++v) {
                if (u == v) continue;
                fb[u][v] = min(fb[u][v], fb[u][k] + fb[k][v]);
            }
        }
    }

    int result = points[1];
    vector<int> ans;

    for (int mask = 1; mask < (1 << (n - 1)); ++mask) {
        vector<int> per;
        for (int i = 0; i < (n - 1); ++i) {
            if (((mask >> i) & 1) == 1) {
                per.push_back(i + 2);
            }
        }
        bool flag = false;
        do {
            int local = fb[1][per[0]] + fb[per.back()][1];
            for (int i = 0; i < per.size() - 1; ++i) {
                local += fb[per[i]][per[i + 1]];
            }
            if (local <= t) {
                flag = true;
                break;
            }
        } while (next_permutation(per.begin(), per.end()));
        if (flag) {
            int local = points[1];
            for (int elem : per) {
                local += points[elem];
            }
            if (local > result) {
                result = local;
                ans = per;
            } else if (local == result && ans.size() > per.size()) {
                result = local;
                ans = per;
            } else if (local == result && ans.size() == per.size()) {
                sort(ans.begin(), ans.end());
                sort(per.begin(), per.end());

                if (ans > per) {
                    result = local;
                    ans = per;
                }
            }
        }
    }

    sort(ans.begin(), ans.end());

    cout << ans.size() + 1 << "\n" << "1 ";
    for (int elem : ans) {
        cout << elem << " ";
    }
    cout << "\n";
}
```

```
[159] struct man {
[160]     int p;
[161]     int mutation;
[162] };
[163]
[164] void solve5() {
[165]     int m, n, k;
[166]     cin >> m >> n >> k;
[167]
[168]     vector<man> mans(1e6 + 1);
[169]     for (int i = 1; i < n; ++i) {
[170]         int s, d, b;
[171]         cin >> s >> d >> b;
[172]         mans[d] = {s, b};
[173]     }
[174]
[175]     for (int i = 0; i < k; ++i) {
[176]         int p, q;
[177]         cin >> p >> q;
[178]
[179]         set<int> merge;
[180]
[181]         while (p != 1) {
[182]             merge.insert(mans[p].mutation);
[183]             p = mans[p].p;
[184]         }
[185]
[186]         while (q != 1) {
[187]             merge.insert(mans[q].mutation);
[188]             q = mans[q].p;
[189]         }
[190]
[191]         int ans = 0;
[192]         int last = -1;
[193]         for (int elem : merge) {
[194]             ans = max(ans, elem - last - 1);
[195]             last = elem;
[196]         }
[197]         ans = max(ans, m - last - 1);
[198]
[199]         cout << ans << "\n";
[200]     }
[201] }
[202]
[203] signed main() {
[204]     solve5();
      }
```