Code submission: 100%

```
#include <CGAL/QP_models.h>
#include <CGAL/QP_functions.h>
#include <CGAL/Gmpz.h>
#include <vector>

using namespace std;

typedef CGAL::Gmpz ET;
typedef CGAL::Quadratic_program<long> Program;
typedef CGAL::Quadratic_program_solution<ET> Solution;

void solve() {
  int n, m; long s;
  cin >> n >> m >> s;

  vector<pair<int, int>> nobles(n);
  vector<pair<int, int>> commons(m);

  long sum_nx = 0, sum_ny = 0;
  long sum_cx = 0, sum_cy = 0;

  for (int i = 0; i < n; i++) {
    int x, y; cin >> x >> y;
    sum_nx += x; sum_ny += y;
    nobles[i] = make_pair(x, y);
  }

  for (int i = 0; i < m; i++) {
    int x, y; cin >> x >> y;
    sum_cx += x; sum_cy += y;
    commons[i] = make_pair(x, y);
  }


  Program lp(CGAL::SMALLER, false, 0, false, 0);
  const int a = 0, b = 1, c = 2, d = 3, l = 4;
  int row = 0;

  for (int i = 0; i < n; i++) {
    // ax + by + c <= 0
    lp.set_a(a, row, nobles[i].first);
    lp.set_a(b, row, nobles[i].second);
    lp.set_a(c, row, 1);
```

```
  row++;
}

for (int i = 0; i < m; i++) {
  // ax + by + c >= 0 <=> -ax - by - c <= 0
  lp.set_a(a, row, -commons[i].first);
  lp.set_a(b, row, -commons[i].second);
  lp.set_a(c, row, -1);
  row++;
}

// encorce a != 0
lp.set_l(a, true, 1);
lp.set_u(a, true, 1);

// Linearly separable data
Solution s1 = CGAL::solve_linear_program(lp, ET());

if (s1.is_infeasible()) {
  cout << "Yuck!" << endl;
  return;
}

if (s != -1) {
  lp.set_a(b, row, sum_cy - sum_ny);
  lp.set_a(c, row, m - n);
  lp.set_b(row, s - (sum_cx - sum_nx));
  row++;

  // Sum of all sewage pipes <= s
  Solution s2 = CGAL::solve_linear_program(lp, ET());

  if (s2.is_infeasible()) {
    cout << "Bankrupt!" << endl;
    return;
  }
}

for (int i = 0; i < n; i++) {
  // length nobles water pipe <= l
  lp.set_a(b, row, nobles[i].first);
  lp.set_a(d, row, 1);
  lp.set_a(l, row, -1);
  lp.set_b(row, nobles[i].second);
```

```
    row++;
    lp.set_a(b, row, -nobles[i].first);
    lp.set_a(d, row, -1);
    lp.set_a(l, row, -1);
    lp.set_b(row, -nobles[i].second);
    row++;
  }

  for (int i = 0; i < m; i++) {
    // length commons water pipe <= l
    lp.set_a(b, row, commons[i].first);
    lp.set_a(d, row, 1);
    lp.set_a(l, row, -1);
    lp.set_b(row, commons[i].second);
    row++;
    lp.set_a(b, row, -commons[i].first);
    lp.set_a(d, row, -1);
    lp.set_a(l, row, -1);
    lp.set_b(row, -commons[i].second);
    row++;
  }

  // Ensure positive length
  lp.set_l(l, true, 0);

  // Minimize length
  lp.set_c(l, 1);

  // Minimization problem on longest water pipe
  Solution s3 = CGAL::solve_linear_program(lp, ET());

  cout << fixed << setprecision(0) << ceil(CGAL::to_double(s3.objective_value())) << endl;
}

int main() {
  ios_base::sync_with_stdio(false);
  int t; cin >> t;
  for (int i = 0; i < t; i++) solve();
}
```