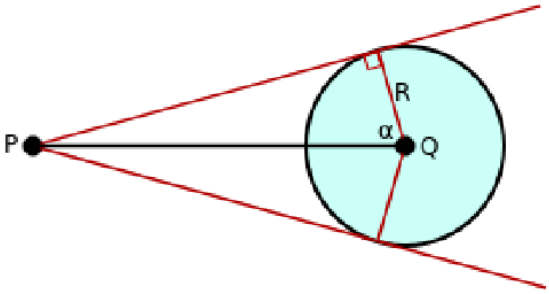
USACO NOV13 Problem 'sight' Analysis

by Bruce Merry and Mark Gordon

Two cows can see each other if and only if there is at least one point on the silo that they can both see. This is not immediately obvious, but it is not difficult to convince yourself of this with a few diagrams. Another diagram can help us actually compute the range of angles visible from a point.



In the diagram PQ forms the hypotenuse of a right triangle. Therefore [PQ] cos(α) = R and we can write α=acos(R/[PQ]).

To avoid double-counting, let's count the ordered pair (A, B) if the counter-clockwise-most point visible from B is within the arc visible from A. Again, draw yourself some diagrams to convince yourself that each unordered pair of visible cows will be counted as either (A, B) or as (B, A), but not both.

Of course, the limits are too large to check each pair individually. Instead, we can create a list of all the CCW-most endpoints and sort it by the angle relative to the origin. This allows a binary search to be used to find all CCW-most endpoints within a given arc. Note that some special handling is needed for arcs that "wrap around".

Another way to think about solving this problem is by visualizing a sweeping tangent around the silo. As the tangent moves around the silo points enter and leave the far side of the tangent (away from the silo). When a point enters (or equivalently, exits) the far side of the silo we can add to our result the number of other points on the far side of the silo. For similar reasons as above this will count each pair exactly once.

Below is Mark Gordon's solution implementing the sweeping tangent idea. The code ends up being very similar to the other approach.

```
#include <algorithm>
#include <cstdio>
#include <vector>
#include <cmath>
#include <queue>

using namespace std;
```

```
#define PI 3.14159265358979323846

int main() {
  freopen("sight.in", "r", stdin);
  freopen("sight.out", "w", stdout);

  int N, R;
  scanf("%d%d", &N, &R);

  vector<pair<double, double> > A;
  for(int i = 0; i < N; i++) {
    int x, y; scanf("%d%d", &x, &y);
    double alpha = acos(R / sqrt(1.0 * x * x + 1.0 * y * y));
    double a0 = atan2(y, x) - alpha;
    if(a0 < 0) a0 += 2 * PI;
    A.push_back(make_pair(a0, a0 + 2 * alpha));
  }
  sort(A.begin(), A.end());

  int result = 0;
  priority_queue<double, vector<double>, greater<double> > q;
  for(int iters = 0; iters < 2; iters++) {
    for(int i = 0; i < N; i++) {
      /* Move the tangent forward to A[i].first and remove everything no
longer
       * visible. */
      while(!q.empty() && q.top() < A[i].first) {
        q.pop();
      }

      if(iters == 1) {
        /* We iterate around the points twice but only count points added the
         * second iteration to ensure the sweep tangent contains what it
         * should. */
        result += q.size();
      }

      q.push(A[i].second);
      A[i].first += 2 * PI;
      A[i].second += 2 * PI;
    }
  }

  printf("%d\n", result);
  return 0;
}
```