Convex Hull

```
//convex hull mediante método Graham Scan (sólo puntos enteros)

#include <algorithm>

using namespace std;

#define MAX 100009
#define i64 long long
#define sq(x) ((x)*(x))

struct point {
    i64 x, y;
} P[MAX], C[MAX], P0;

// P[] contains all points
// C[] contains convex hull ccw

inline i64 TriArea2(point a, point b, point c) {
    return (a.x*(b.y-c.y) + b.x*(c.y-a.y) + c.x*(a.y-b.y));
}

inline i64 sqDist(point a, point b) {
    return (sq(a.x-b.x) + sq(a.y-b.y));
}

bool comp(point a, point b) {
    i64 d = TriArea2(P0, a, b);
    if(d<0) return false;
    if(!d && sqDist(P0, b) > sqDist(P0, a)) return false;
    return true;
}

void ConvexHull(int np, int &cp) {
    int i, j, pos = 0;
    for(i=1; i<np; i++)
        if(P[i].y<P[pos].y || (P[i].y==P[pos].y && P[i].x>P[pos].x))
            pos = i;
    swap(P[0], P[pos]);
    P0 = P[0];
    sort(&P[1], P+np, comp);
    C[0] = P[0], C[1] = P[1], C[2] = P[2];
    for(i=j=3; i<np; i++) {
        while(TriArea2(C[j-2], C[j-1], P[i]) <= 0) j--;
        C[j++] = P[i];
    }
    nc = j;
}
```