半平面交

Halfplane Intersection

Idea：用双端队列维护，将直线（即代表半平面）按极角排序后，每加入一个新的半平面，弹出队首队尾无用的半平面（前一个交点在当前直线的右边则无用）

Application：求多边形的核，凸多边形最大内切圆，线性规划区域

Complexity：$O(n \lg n)$

ATT：我的代码中，如果半平面交为一个点，并不会视之为空，尽管其面积为零。

Code：

```
// Attention: In my code, if the intersection is a point, it will not be seen as empty, though its area is 0.
// if m >= 3, then the intersection exists (including the situation where ths intersection is a point).
// if a point is not regarded valid in a particular problem, you should calculate the area.
Point P[N]; // p[i] is the intersection of line q[i] and q[i+1]
Line Q[N]; // deque
void HalfplaneIntersection(Line L[], int n, Point res[], int &m){
    // L[] are the lines, n is the number of lines, res[] stores the result of the intersection (a polygon)
    // m is the number of points of the intersection (which is a polygon)
    sort(L + 1, L + n + 1);
    int head, tail;
    Q[head = tail = 0] = L[1];
    for(int i = 2; i <= n; i++){
        while(head < tail && PointOnRight(P[tail - 1], L[i]))   tail--;
        while(head < tail && PointOnRight(P[head], L[i]))   head++;
        Q[++tail] = L[i];
        if(sgn(Q[tail].v ^ Q[tail - 1].v) == 0){ // parallel, the inner one remains
            tail--;
            if(!PointOnRight(L[i].p, Q[tail]))    Q[tail] = L[i];
        }
        if(head < tail) P[tail - 1] = GetLineIntersection(Q[tail-1], Q[tail]);
    }
    while(head < tail && PointOnRight(P[tail - 1], Q[head])) tail--; // delete useless plane
    P[tail] = GetLineIntersection(Q[tail], Q[head]);

    m = 0;
    for(int i = head; i <= tail; i++)   res[++m] = P[i];
}
```