

凸包

Convex Hull

Idea: 水平序 Graham 扫描法：首先将所有的点以 x 为第一关键字、 y 为第二关键字排序，然后分上下凸包分别求解：求解下凸包时，维护一个栈存储当前凸包的点，顺序扫描每个点，每扫描到一个点便判断其与栈内前两个点的转向关系，若成左转关系，则该点入栈，否则弹出栈顶元素继续判断，直至判断到头或形成左转关系为止；同理，逆序扫描点求解上凸包。

Complexity: $O(n \lg n)$ （瓶颈在于排序的复杂度，若对于特殊情况采用基数排序可优化复杂度）

Code:

```
1 void ConvexHull(int n, Point p[], Point sta[], int &staid){
2     // there're n points stored in p[], the points on convex hull
   will be saved in sta[]
3     sort(p+1, p+n+1);
4     n = unique(p+1, p+n+1) - (p+1);
5     staid = 0;
6     for(int i = 1; i <= n; i++){
7         // while(staid > 1 && sgn((sta[staid]-sta[staid-1]) ^ (p[i]-
   sta[staid-1])) < 0) staid--; // points on edge
8         while(staid > 1 && sgn((sta[staid]-sta[staid-1]) ^ (p[i]-
   sta[staid-1])) <= 0) staid--; // no points on edge
9         sta[++staid] = p[i];
10    }
11    int k = staid;
12    for(int i = n-1; i >= 1; i--){
13        // while(staid > k && sgn((sta[staid]-sta[staid-1]) ^ (p[i]-
   sta[staid-1])) < 0) staid--; // points on edge
14        while(staid > k && sgn((sta[staid]-sta[staid-1]) ^ (p[i]-
   sta[staid-1])) <= 0) staid--; // no points on edge
15        sta[++staid] = p[i];
16    }
17    if(n > 1) staid--;
18 }
```