C6F PolygonArea

题目描述

计算逆时针给出的 n 个点组成的多边形的面积的两倍

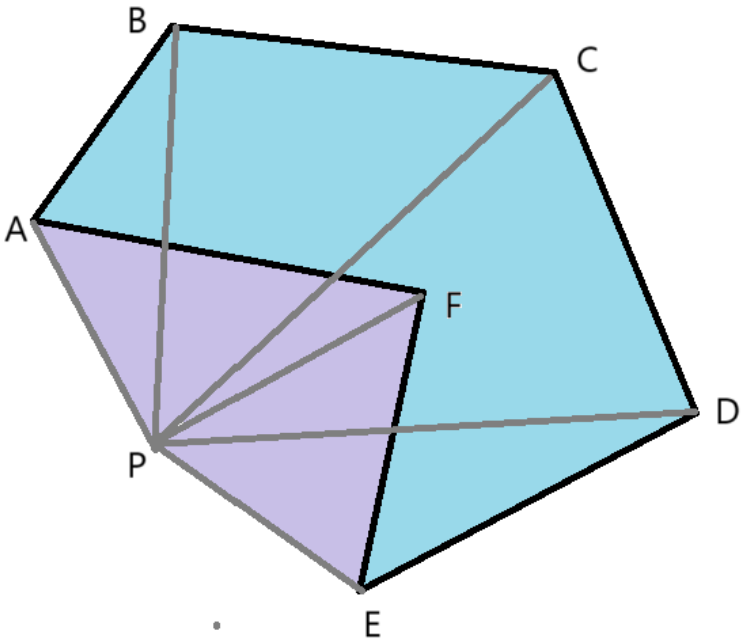
F PolygonArea

时间限制：1000ms 内存限制：65536kb

通过率：121/134 (90.30%) 正确率：121/314 (38.54%)

解题思路

任意取点 P，按顺序遍历边，计算每一条边与 P 组成的三角形的面积（逆时针为正，顺时针为负），得到的面积即为所求目标的 $\frac{1}{2}$



代码实现

```cpp
#include <bits/stdc++.h>

using namespace std;

using ll = long long;
using db = long double;  // or double, if TL is tight
```

```
using str = string;
using u32 = unsigned int;
using u64 = unsigned long long;
using u128 = unsigned __int128;

using pii = pair<int, int>;
using pll = pair<ll, ll>;
using pdd = pair<db, db>;
#define fi first
#define se second

#define vc vector
using vi = vc<int>;
using vb = vc<bool>;
using vll = vc<ll>;
using vd = vc<db>;
using vs = vc<str>;
using vch = vc<char>;
using vpii = vc<pii>;
using vpll = vc<pll>;
using vpdd = vc<pdd>;
using vvi = vc<vc<int>>;
using vvll = vc<vc<ll>>;
using vvch = vc<vc<char>>;
using vvb = vc<vc<bool>>;
using vvpii = vc<vc<pii>>;
using vvpll = vc<vc<pll>>;
using vvvi = vc<vc<vc<int>>>;

const int mod = 998244353;  // 1e9 + 7;
const int INF = 0x3f3f3f3f;
const ll BIG = 1e18;  // not too close to LLONG_MAX
const db PI = acos((db)-1);
const int dx[4]{1, 0, -1, 0}, dy[4]{0, 1, 0, -1};  // for
every grid problem!!
mt19937
rng32((uint32_t)chrono::steady_clock::now().time_since_epo
ch().count());
mt19937_64
rng64((uint64_t)chrono::steady_clock::now().time_since_epo
ch().count());

#define cl(x, y) memset(x, y, sizeof(x))
#define de(x) cerr << #x << " = " << x << " "
#define del(x) cerr << #x << " = " << x << endl

#define overload3(a, b, c, d, ...) d
#define rep1(a) for(int i = 0; i < (a); i++)
#define rep2(i, a) for (int i = 0; i < (a); i++)
#define rep3(i, a, b) for (int i = (a); i <= (b); i++)
#define per1(a) for(int i = (a) - 1; i >= 0; i--)
#define per2(i, a) for (int i = (a) - 1; i >= 0; i--)
```

```
#define per3(i, a, b) for (int i = (a); i >= (b); i--)
#define rep(args...) overload3(args, rep3, rep2, rep1)
(args)
#define per(args...) overload3(args, per3, per2, per1)
(args)
#define for_subset(t, s) \
    for (ll t = (s); t >= 0; t = (t == 0 ? -1 : (t - 1) &
(s)))

#define co cout
#define dl '\n'
#define ret return
#define pb push_back
#define tcT template <class T
#define YES cout << "YES" << endl
#define Yes cout << "Yes" << endl
#define NO cout << "NO" << endl
#define No cout << "No" << endl
#define INT(args...) \
    int args;        \
    rd(args)
#define LL(args...) \
    ll args;        \
    rd(args)

#define all(s) s.begin(), s.end()
#define rall(s) s.rbegin(), s.rend()
#define sz(v) int(v.size())
#define SUM(a) accumulate(all(a), 0ll)
#define MIN(v) (*min_element(all(v)))
#define MAX(v) (*max_element(all(v)))
#define unisort(v) \
    sort(all(v));  \
    v.resize(unique(all(v)) - v.begin())

int popcnt(int x) { return __builtin_popcount(x); }
int popcnt(u32 x) { return __builtin_popcount(x); }
int popcnt(ll x) { return __builtin_popcountll(x); }
int popcnt(u64 x) { return __builtin_popcountll(x); }

#define LOWBIT(x) ((x) & (-x))
// (0, 1, 2, 3, 4) -> (-1, 0, 1, 1, 2)
int topbit(int x) { return (x == 0 ? -1 : 31 -
__builtin_clz(x)); }
int topbit(u32 x) { return (x == 0 ? -1 : 31 -
__builtin_clz(x)); }
int topbit(ll x) { return (x == 0 ? -1 : 63 -
__builtin_clzll(x)); }
int topbit(u64 x) { return (x == 0 ? -1 : 63 -
__builtin_clzll(x)); }
// (0, 1, 2, 3, 4) -> (-1, 0, 1, 0, 2)
```

```
int lowbit(int x) { return (x == 0 ? -1 :
__builtin_ctz(x)); }
int lowbit(u32 x) { return (x == 0 ? -1 :
__builtin_ctz(x)); }
int lowbit(ll x) { return (x == 0 ? -1 :
__builtin_ctzll(x)); }
int lowbit(u64 x) { return (x == 0 ? -1 :
__builtin_ctzll(x)); }
tcT > T cdiv(T &a, T &b) {
    return a / b + ((a ^ b) > (T)(0) && a % b);
}  // divide a by b rounded up
tcT > T fdiv(T &a, T &b) {
    return a / b - ((a ^ b) < (T)(0) && a % b);
}  // divide a by b rounded down

tcT, class S > inline ostream &operator<<(ostream &i,
const pair<T, S> &a) { return i << a.first << ' ' <<
a.second; }
tcT, class S > inline istream &operator>>(istream &i,
pair<T, S> &a) { return i >> a.first >> a.second; }
tcT > inline ostream &operator<<(ostream &i, const
vector<T> &a) {
    if (a.size())
        i << a[0];
    for (int j = 1; j < int(a.size()); j++)
        i << ' ' << a[j];
    return i;
}
tcT > inline istream &operator>>(istream &i, vector<T> &a)
{
    for (auto &j : a)
        i >> j;
    return i;
}
void err(istream_iterator<string> it) {
it.~istream_iterator(); }
template <typename T, typename... Args>
void err(istream_iterator<string> it, T a, Args... args) {
    cerr << *it << " = " << a << endl;
    err(++it, args...);
}
void wt() { cout << '\n'; }
tcT, class... Args > void wt(const T &a, const Args
&...args) {
    cout << a;
    (void)(cout << ... << (cout << ' ', args));
    cout << '\n';
}
tcT > void wt(vector<vector<T>> &a) {
    for (auto &i : a)
        wt(i);
}
```

```
tcT, class S > void wt(vector<pair<T, S>> &a) {
    for (auto &i : a)
        wt(i);
}
template <class... T>
void rd(T &...a) { (cin >> ... >> a); }

#define LOCAL_TEST freopen("out.txt", "w", stdout)
#define IOS
ios::sync_with_stdio(false);cin.tie(0);cout.tie(0)
#define FIX(x) cout << fixed << setprecision(x)

const int maxn = 1e5 + 5;

inline void INIT() {

}

template <class T>
int sgn(T x) { return (x > 0) - (x < 0); }

template <class T>
struct Point {
    typedef Point P;
    T x, y;

    explicit Point(T x = 0, T y = 0) : x(x), y(y) {}
    bool operator<(P p) const { return tie(x, y) <
tie(p.x, p.y); }
    bool operator==(P p) const { return tie(x, y) ==
tie(p.x, p.y); }
    P operator+(P p) const { return P(x + p.x, y + p.y); }
    P operator-(P p) const { return P(x - p.x, y - p.y); }
    P operator*(T d) const { return P(x * d, y * d); }
    P operator/(T d) const { return P(x / d, y / d); }
    T dot(P p) const { return x * p.x + y * p.y; }
    T cross(P p) const { return x * p.y - y * p.x; }
    T cross(P a, P b) const { return (a - *this).cross(b -
*this); }
    T dist2() const { return x * x + y * y; }
    double dist() const { return sqrt((double)dist2()); }
    // angle to x-axis in interva l [-pi , pi ]
    double angle() const { return atan2(y, x); }
    P unit() const { return *this / dist(); } // makes d
is t ()=1
    P perp() const { return P(-y, x); }       // rotates
+90 degrees
    P normal() const { return perp().unit(); }
    // returns point rotated 'a ' radians ccw around the
origin
    P rotate(double a) const {
```

```
        return P(x * cos(a) - y * sin(a), x * sin(a) + y *
cos(a));
    }
    friend ostream &operator<<(ostream &os, P p) {
        return os << "( " << p.x << ", " << p.y << " )";
    }
};

/* Returns twice the signed area of a polygon. Clockwise
enumeration gives negative area */
template<class T>
T polygonArea2(vector<Point<T>>& v) {
    T a = v.back().cross(v[0]);
    rep(i, sz(v) - 1) a += v[i].cross(v[i + 1]);
    return a;
}

inline void SOLVE() {

    INT(n);
    vc<Point<ll>> a(n);
    rep(n) {
        cin >> a[i].x >> a[i].y;
    }
    wt(polygonArea2(a));

}

/*

*/

int main() {

    IOS; FIX(20);

    INIT();
    int tt = 1;
    // cin >> tt;
    while(tt--)
        SOLVE();

    // cout << "program ends..." << endl;
    // system("pause");
    return 0;
}
```