Geometry – II

نویسنده رضا آشتیانی

نقطه :
از نظر مفهوم که نقطه چیز خاصی برای گفتن نداره ، ولی از این بابت که چند ضلعی هارو با یه سری نقطه نشون میدیم خیلی مهم اند .
برای ذخیره کردن نقطه در برنامه از کلاس **complex** استفاده می کنیم .
چون در ادامه کار احتیاج به مقایسه نقاط با هم داریم ، برای **namespace std** باید **operator** کوچک تر رو **overload** کنیم .
نمونه کد :

```
typedef complex < double > point;
namespace std
{
    bool operator < ( const point &a, const point &b )
    {
        return real(a) != real(b) ? real(a) < real(b) : imag(a) < imag(b);
    }
}
```

خط ، نیم خط و پاره خط :
خط هم مثل نقطه مفهوم خاصی نداره و فقط با ارائه یه نمونه کد ازش می گذریم .
نمونه کد :

```
struct L : public vector < point >
{
    L( point &a , point &b )
    {
        push_back(a);
        push_back(b);
    }
};
```

معمولا سوالاتی که از بخش خط مطرح میشه در مورد تقاطع چند خط یا چک کردن هم خط بودن یه سری نقطه یا ... هستند . در ادامه چند تابع خوب که تو حل سوالات خیلی به کار میان رو معرفی می کنیم
نحوه بدست آوردن وضعیت دو خط نسبت به هم :
در حالت کلی دو خط نسبت به هم سه حالت دارن
1– خطوط موازی و منطبق باشند : برای چک کردن موازی دو روش داریم
الف : شیب خط هارو با هم مقایسه کنیم .
ب : اگر دو خط موازی باشند ، زاویه بین اونها **0** یا **180** خواهد بود ، که در این صورت از تعریف ضرب خارجی نتیجه می گیریم که مقدار حاصلضرب خارجی دو خط موازی **0** میشه .

چون در محاسبه شیب خط ممکنه تقسیم به صفر داشته باشیم ، برای اینکه با این مورد مواجه نشیم از **rational number** استفاده می کنیم ولی این راه خیلی طولانی میشه و فقط کار رو برای ما سخت می کنه ، پس ما برای چک کردن توازی دو خط از ضرب خارجی استفاده می کنیم .
برای هم خط بودن هم ، به سادگی یه نقطه از خط اول و یه نقطه از خط دوم بر می داریم و با یکی از خطوط مقایسه می کنیم .
2 – خطوط موازی و غیر منطبق باشند : این مورد خیلی شبیه قسمت بالاست .
3 – خطوط موازی نباشند : تو این حالت تقاطع دارند و معمولا تو سوالات بدست آوردن محل تقاطع نکته مهمیه .
برای بدست آوردن تقاطع ، مثلا نقطه **X** ، از تناسب استفاده می کنیم .
فرض می کنیم نقاط خط اول **1LO, L** و نقاط خط دوم **1MO, M** باشند .
چون نقطه تقاطع روی خط **M** هست ، نسبت فاصله **OM** تا **X** رو به کل خط **M** بدست میاریم .
برداری که از **OM** شروع میشه و به **X** ختم میشه رو می سازیم ، به وضوح می دونیم این بردار و خط **M** در یک راستا هستند و اندازه این بردار برابر است با طول **M** ضرب در نسبتی که بدست آوردیم .
حالا که این بردار رو داریم ، مختصات نقطه **X** میشه ، **OM +** برداری که بدست آوردیم .
برای بدست آوردن تمام این موارد یه تابع می نویسیم :
این تابع اگر دو خط موازی باشند ، **false** و در غیر این صورت **true** بر می گردونه ، به عنوان یه آرگومان ادرس یه نقطه رو به تابع پاس میدیم . اگر خروجی تابع **true** بود ، مقداری که در اون نقطه ذخیره شده ، نقطه تقاطع دو خط هستش .
نمونه کد :

```
bool crosspoint( L &l , L&m , point &x )
{
    double A = cross( l[1] - l[0] , m[1] - m[0] );
    double B = cross( l[1] - l[0] , l[1] - m[0] );
    if( abs(A) < EPS && abs(B) < EPS )//same line
    {
        x = m[0];
        return true;
    }
    if( abs(A) < EPS )//parallel line
        return false;
    x = m[0] + B/A * ( m[1] - m[0] );
    return true;
}
```

اگر فقط بخوایم بدونیم که دو تا خط با هم تقاطعی دارند یا نه ، از تابع زیر استفاده می کنیم .
نمونه کد :

```
bool intersectLL( L &l , L&m )
{
    return abs( cross( l[1] - l[0] , m[1] - m[0] ) ) > EPS //non-parallel
        || abs( cross( l[1] - l[0] , l[1] - m[0] ) ) < EPS;//same line
}
```

به همین ترتیب می تونید توابع **intersectLS** , **intersectSS** , **intersectSP** , **intersectLP** رو بنویسید .

فاصله دو خط :

برای این کار اول باید چک کنیم که دو خط تقاطع دارند یا نه . برای این کار از توابع زیر استفاده می کنیم .

نمونه کد :

```
point projection( L &l , point &p )
{
    double t = dot( p - l[0] , l[0] - l[1] ) / norm( l[0] - l[1] );
    return l[0] + t * ( l[0] - l[1] );
}
double distanceLP( L &l , point &p )
{
    return abs( p - projection( l , p ) );
}
double distanceLL( L &l , L&m )
{
    return intersectLL( l , m ) ? 0 : distanceLP( l , m[0] );
}
```

تابع **projection** ، مختصات تصویر نقطه **p** روی خط **L** رو بدست میاره .

به همین ترتیب می تونید توابع دیگه ای که نیاز دارید از جمله **distanceSS** , **distanceSP** , **distancePP** رو بنویسید .