

## 1 Yinset

### 1.1 成员数据

**vector<GluingClosedSurface> vecGCS**

这里存放了构成殷集边界的有向黏合紧曲面

**vector<HasseNode> Hasse**

存放了有向黏合紧曲面之间的包含关系

### 1.2 成员函数

**Yinset meet(const Yinset&) const**

实现了两个殷集的求交

**Yinset join(const Yinset&) const**

实现了两个殷集的求并

**Yinset complement() const**

实现了殷集求补集

**buildHasse()**

计算黏合紧曲面的包含关系，输出 Hasse 图

## 2 GluingClosedSurface

表示一个黏合紧曲面

### 2.1 成员数据

**vector<Triangle> vecTriangle**

存放了黏合紧曲面的三角剖分

**bool orientation**

存放了黏合紧曲面的方向

## 3 SurfacePatch

表示切割后的曲面片

### 3.1 成员数据

**vector<Triangle> vecTriangle**

存放了曲面片的三角剖分

**vector<pair<Segment> boundary**

存放了曲面片的边界

### 3.2 成员函数

**reverse()**

将曲面片反向，也就是将所有三角形的顶点顺序取反

## 4 PrePaste

实现了将曲面沿闭合交线切割成曲面片的过程

### 4.1 成员数据

**vector<GluingClosedSurface> vecGCS**

存放了不需要进行切割的黏合紧曲面

**vector<SurfacePatch> vecSP**

存放了切割以后得到的曲面片

### 4.2 成员函数

**operator()(const vector<Triangle>&)**

将一个股集中所有三角形放在一起作为输入，将这些三角形黏合起来，直到遇到边界，这个过程等效于将黏合紧曲面沿交线进行切割。

## 5 Paste

实现了将曲面片沿边界黏合成黏合紧曲面的过程

### 5.1 成员函数

**vector<GluingClosedSurface> operator()(const vector<SurfacePatch>&)**

将输入的曲面片沿边界黏合成黏合紧曲面并输出

## 6 Locate

### 6.1 成员函数

**bool operator()(const Point&, const GluingClosedSurface&)**

判断一个点是否在一个黏合紧曲面的有界补集内部

## 7 TriangleIntersect

### 7.1 成员数据

**vector<pair<vector<Segment>, vector<vector<Triangle>::iterator>>**

**: resultA, reasultB**

存放了两个股集的所有三角形之间相交的信息和重合的信息

### 7.2 成员函数

**operator()(const Triangle&, const Triangle&)**

实现两个三角形的求交

**vector<Triangle> collapse()** 将所有三角形根据相交信息进行三角

剖分

## 8 Triangulate

### 8.1 成员函数

**bool operator()(const Triangle&, const vector<Segment>&)**

将输入的三角形根据交线进行三角剖分

## 9 Triangle

实现了三角剖分所需的三角形

### 9.1 成员数据

**vector<Point> vecPoint**

存放了三角形三个顶点，顶点顺序与定向有关

**pairt<int,int> InFace** 记录在哪一个曲面中

### 9.2 成员函数

**Triangle<2> project(int n)**

将三维空间三角形投影到某个坐标平面

**intersect(const Line&)**

实现空间中三角形与一条直线求交

**intersectCoplane(const Line<2>&)** 实现平面中三角形和直线求交

**Triangle reverse()** 将三角形顶点顺序反向

## 10 Plane

表示三角形所在平面

### 10.1 成员数据

**Real para[Dim+1]**

存放了平面方程的四个参数

### 10.2 成员函数

**Real angle(const Plane&)**

求两个平面的夹角

**Line intersect(const Plane&**

实现两个平面求交，输出交的直线

## 11 Line

表示一条直线

### 11.1 成员数据

**Point fixPoint**

存放了直线上一点坐标

**Vec direct**

存放了直线的方向向量

### 11.2 成员函数

**Line<2> project(int n)**

将空间中直线投影到某个坐标平面

## 12 Edge

表示一条线段

### 12.1 成员数据

**Point endPoint[2]**

表示线段的两个端点

### 12.2 成员函数

**Edge<2> project(int n)**

将空间中线段投影到某个坐标平面

## 13 Segment

表示一条交线

### 13.1 成员数据

**Point endPoint[2]**

表示交线的两个端点

**vector<Triangle>**

存放了交线对应的两个三角形

## 14 Point

表示空间中一个点

### 14.1 成员数据

**Real coord[Dim]**

表示点的坐标

## 15 Vec

表示一个向量

### 15.1 成员数据

**Real p[Dim]**

表示向量的各个分量

### 15.2 成员函数

**Real dot(const Vec&)**

实现向量点乘

**Real cross(const Vec&)**

实现向量叉乘