**点阵有序化程序说明文档**

功能说明：

1. 对焊线点列表所在面的点阵有序排列输出。
2. 输出各点连接的单元。

**基本流程**



1.在构建GeObject对象后，需要焊线点列表、集合体单元统计表、点对应空间位置表、几何体单元所含点阵表、初始点、初始单元和初始法线数据进行初始化。

2.初始化完成后，进入找点连接单元步骤，得到一组各点连接单元的表。

3.处理完成后，进入点阵有序化步骤，得到焊线点列表所在面的点阵序列表。

**主要步骤解释**

1. 找点连接单元

遍历几何体单元所含点阵表中每个单元对应的每个点，构建一张以点为Key、单元集合为Value的表，将每个单元插入到点对应的集合中。

1. 点阵有序化



点阵有序化主要通过solvePointSort函数进行，由左到右、由上到下进行处理，最终形成多列点阵序列表。函数需要输入起始点ID、起始单元ID、起始面法向量。

1. 预处理1部分：①需要为点阵序列表创建新列，并将起始点插入进去；②清理下一点列表；③清理备选单元列表。
2. 层处理1部分：调用solveUnit函数进行层处理，需要输入的参数为起始点ID、起始单元ID、起始面法向量和0。
3. 层处理1结束后，进入一个循环，循环的跳出条件为colTaiFlag为真，即已处理完所有列。
4. 预处理2部分：①重置行相关Flag（rowHeadFlag&rowTailFlag）；②清理备选单元列表；③（当下一节点序列数大于1时）更新参数，起始点ID从下一节点序列表中获取，焊线节点ID由函数findNextWireBondingPointID获取，法向量由函数getNormalVector获取，下一起始单元由函数findNextFirstUnit获取。
5. 层处理2部分：调用solveUnit函数进行层处理，需要输入的参数由第4步骤更新。

**部分函数说明**

solveUnit



函数需要输入起始点ID、起始单元ID、法向量、焊线点ID。

1. 预处理1部分：①移除焊线点列表中的起始点；②移除几何单元统计表（未处理单元）中的起始单元；③组成焊线向量，若输入的焊线点ID为0，则通过函数findNextWireBondingPointID获取。
2. 头层处理：根据rowHeadFlag的值判断当前处理层是否为头层，对头层进行额外处理①将焊线节点加入到下一处理单元列表中。
3. 找到两个目标点，并区分当前点列和下一点列部分。①先通过当前处理单元的其余点与焊线向量形成的面同法向量垂直，得到备选点序列。②若备选点序列数为1，则判断三点形成的图形是左上三角形还是右上三角形，通过函数getPointLinkCount获取起始点连着未处理的单元的数量，若为0则是左上三角，否则为右上三角，接着通过单元连接线找到当前面的剩余的未处理的单元，再找到另一个备选点；③根据两个备选点与焊线向量形成的夹角的大小区分当前点列和下一点列部分。
4. 找下一起始单元，通过函数findUnit获取。
5. 根据rowTailFlag的值判断当前处理层是否到尾层，①对尾层仅进行整理操作将当前点列的下一个点加入到点序列中，然后将下一点列的当前层的下一个点加入到下一点列中。②对非尾层还需要以更新后的值为参数进行递归调用。

findNextWireBondingPointID



函数需要输入起始点ID。

1. 预处理部分：定义res=0；
2. 处理部分：①从点连接单元中查找输入点连接的单元；②遍历各单元，对每个单元的每个点，若在焊线点列表内找到它，则更新res的值；③每次单元遍历结束，若在此次处理中找到了该焊线点，则将该单元加入到备选单元列表中。

findNextFirstUnit



函数需要输入下一起始点ID、焊线ID、法向量。

1. 预处理部分： ①定义返回值res=0；②将两个输入点形成一个临时向量vector1。
2. 遍历备选单元列表，对每个单元都遍历其除输入点外的结构点，判断三点是否能形成与法向量垂直的平面，找到则更新res的值。
3. 返回res。

getPointLinkCount



函数需要输入点ID、法向量。

1.预处理部分：①定义返回值res=0；定义（是否找到该单元）跳出循环的控制变量flag。  
2.处理部分：①在点连接单元列表中找到该点连接的单元；②若未处理单元中包含该单元，则遍历该单元的所有点；③若该单元内有两个点能与输入点形成的平面同输入法线垂直，则更新res的值。