**### \*\*STAP++程序Q4单元功能、算法与实现报告\*\***

**#### \*\*一、 引言 (含所实现的所有功能说明)\*\***

**本项目基于STAP++有限元分析框架，成功设计、实现并调试了二维四节点等参（Q4）单元，使其能够对平面弹性力学问题进行有效的求解和分析。通过该模块，程序具备了以下核心功能：**

**1. \*\*模型输入与定义\*\*:**

**\* \*\*几何与网格\*\*: 程序能够从文本输入文件（`.dat`）中读取模型的节点坐标、单元连接关系，从而定义分析对象的几何外形和有限元网格。**

**\* \*\*边界条件\*\*: 支持对任意节点、任意方向（X、Y、Z）施加位移约束。**

**\* \*\*材料属性\*\*: 支持定义线弹性材料，可指定杨氏模量（E）、泊松比（ν），并支持平面应力（Plane Stress）与平面应C变（Plane Strain）两种分析类型。**

**\* \*\*单元属性\*\*: 支持为Q4单元指定厚度。**

**\* \*\*载荷施加\*\*: 支持在任意节点、任意方向上施加集中载荷。**

**2. \*\*有限元核心计算\*\*:**

**\* \*\*刚度矩阵计算\*\*: 能够基于单元的几何、材料及类型（平面应力/应变），精确计算每个Q4单元的单元刚度矩阵（Ke）。**

**\* \*\*全局矩阵组装\*\*: 采用高效的“天际线（Skyline）”存储格式，将所有单元刚度矩阵组装成全局刚度矩阵（K），并组装全局载荷向量（F）。**

**\* \*\*位移求解\*\*: 高效求解大型线性方程组 `[K]{U} = {F}`，得出模型所有节点的位移响应（U）。**

**3. \*\*结果分析与输出\*\*:**

**\* \*\*位移输出\*\*: 在计算完成后，将每个节点的X、Y方向位移输出到结果文件。**

**\* \*\*应力应变计算\*\*: 能够根据求解出的节点位移，回代计算每个单元内部高斯积分点处的应力（σx, σy, τxy）和应变（εx, εy, γxy）。**

**\* \*\*结果输出\*\*: 将详细的应力、应变计算结果格式化输出到结果文件，并计算等效（Von Mises）应力，为强度校核提供依据。**

**\* \*\*后处理支持\*\*: （规划中）支持将网格、位移、应力等结果数据导出为标准VTK格式文件，以便使用ParaView等专业后处理软件进行云图可视化。**

**#### \*\*二、 算法说明 (简要)\*\***

**Q4单元的求解遵循标准的有限元法（FEM）流程，其核心算法理论如下：**

**1. \*\*离散化与形函数\*\*:**

**首先，将连续的求解域（如悬臂梁）离散为一系列相互连接的Q4单元。在每个单元内部，任一点的位移场 `u(ξ, η)` 通过该单元四个节点的位移 `d` 和等参形函数 `N(ξ, η)` 进行插值得到：`u = N \* d`。其中 `(ξ, η)` 是取值范围为 `[-1, 1]` 的单元自然坐标。**

**2. \*\*几何与物理关系\*\*:**

**\* \*\*应变-位移关系\*\*: 单元的应变 `ε` 通过对位移场求导得出，可以表示为 `ε = B \* d` 的矩阵形式。其中 `B` 矩阵（应变矩阵）是形函数对全局坐标（x, y）偏导数的组合。该偏导数通过雅可比（Jacobian）矩阵 `J` 实现从自然坐标到全局坐标的转换。**

**\* \*\*应力-应变关系\*\*: 材料的本构关系遵循胡克定律，写成矩阵形式为 `σ = D \* ε`。其中 `D` 矩阵（弹性矩阵）根据平面应力或平面应变假设，由材料的杨氏模量 `E` 和泊松比 `ν` 确定。**

**3. \*\*单元刚度矩阵\*\*:**

**单元刚度矩阵 `Ke` 表征了单元抵抗变形的能力。它由虚功原理导出，其计算公式为在单元体积（V）上对 `B^T \* D \* B` 的积分。**

**由于 `B` 矩阵是 `(ξ, η)` 的函数，该积分通常无法获得解析解，因此采用\*\*2x2高斯数值积分\*\*进行精确计算。**

**4. \*\*整体求解\*\*:**

**将所有单元的 `Ke` 按照节点自由度的对应关系，组装（累加）进全局刚度矩阵 `K` 中。同理，施加的外部载荷构成全局载荷向量 `F`。最终，整个结构的平衡方程为 `K \* U = F`。通过求解该大型线性方程组，即可得到全局节点位移向量 `U`。**

**5. \*\*后处理\*\*:**

**得到节点位移 `U` 后，可提取出每个单元的节点位移 `d`，进而使用上述关系式 `ε = B \* d` 和 `σ = D \* ε` 计算出每个单元内部任意点（通常是高斯积分点）的应变和应力。**

**#### \*\*三、 实现方案\*\***

**上述算法在STAP++项目中通过一系列C++类协同实现：**

**1. \*\*核心数据管理类\*\*:**

**\* `CDomain`: 作为程序的“大脑”，是管理所有分析数据的单例类。它负责驱动文件读取、矩阵组装、方程求解和结果输出的整个流程。**

**\* `CNode`: 存储节点编号、三维坐标和各自由度的边界约束代码（bcode）。**

**\* `CLoadCaseData`: 存储每个载荷工况下的节点和载荷信息。**

**2. \*\*Q4单元功能实现\*\*:**

**\* `CQ4Element` 类是Q4单元的核心实现，继承自 `CElement` 基类。**

**\* `Read()`: 从输入流读取定义一个单元所需的4个节点编号、材料集编号 `MSet`、单元厚度 `thickness` 和分析类型代码 `type\_code`。**

**\* `ShapeFunctions()`, `ShapeFunctionDerivatives()`, `Jacobian()`: 实现了计算形函数、形函数导数和雅可比矩阵的标准数学公式。**

**\* `BMatrix()`: 基于上述函数，计算给定积分点处的 `B` 矩阵。**

**\* `DMatrixPlaneStress()`, `DMatrixPlaneStrain()`: 根据材料属性和分析类型，构造对应的 `D` 矩阵。**

**\* `ElementStiffness()`: \*\*关键计算函数\*\*。它内部设置了2x2高斯积分点，循环调用 `BMatrix`, `DMatrix` 和 `Jacobian` 函数，执行 `t \* B^T \* D \* B \* det(J) \* W` 的累加，最终得到单元刚度矩阵 `kes`。\*\*（我们在此处修正了未乘以厚度 `t` 的问题）\*\***

**\* `ElementStress()`: 在位移求解后被调用，用同样的方法计算各积分点的应力和应变。**

**\* `CQ4Material` 类继承自 `CMaterial`，用于存储Q4单元特有的材料属性，包括 `E`, `ν`, `thickness` 和 `analysisType`。**

**3. \*\*矩阵与求解器实现\*\*:**

**\* `CSkylineMatrix` 类：一个模板类，实现了“天际线”存储方案，极大地节省了内存。**

**\* `Assembly()`: 负责将 `CQ4Element` 计算出的上三角 `kes` 数组，正确地累加到全局 `data\_` 数组的对应位置。\*\*（我们在此处修正了访问 `kes` 数组时致命的索引计算错误）\*\***

**\* `CSolver` 类：封装了线性方程组的求解算法，如 `LDLT` 分解和回代，用于求解 `K\*U=F`。**

**4. \*\*整体工作流\*\*:**

**1. `main.cpp` 启动程序，调用 `CDomain::ReadData()` 解析输入文件。**

**2. `CDomain` 创建 `CNode`, `CQ4Element`, `CLoadCaseData` 等对象并填充数据。**

**3. `CDomain::AllocateMatrices()` 初始化 `CSkylineMatrix`。**

**4. `CDomain::AssembleStiffnessMatrix()` 遍历所有 `CQ4Element` 实例，依次调用 `ElementStiffness()` 和 `CSkylineMatrix::Assembly()` 完成全局刚度矩阵的构建。**

**5. `CDomain` 调用 `CSolver` 求解位移。**

**6. `CDomain` 调用 `COutputter` 类，并配合 `CQ4Element::ElementStress()`，将位移、应力等结果写入输出文件。**

**---精确的输入数据格式**

**可以参考算例文件：**

**标题**

**节点数 单元类型数 负载数 求解模式**

**输入节点编号，自由度，坐标**

**载荷数量（*Number of loads in \*this\* specific load case*）载荷编号 节点编号 坐标 载荷大小**

**材料属性数 E 泊松比**

**GroupID NumElements MaterialID**

**单元输入部分较为复杂：**

**单元编号 单元节点编号 MSet Thickness AnalysisType**

**正常编译将文件地址作为变量输入即可，求解产生一定的debug信息，同时会输出节点信息，单元刚度矩阵信息等等辅助分析。**