层级建模中的人形模型（Humanoid Model – Tree Data Structure）解释

一、核心思想

这段代码是计算机图形学中“层级建模（Hierarchical Modeling）”的典型实现，用树结构（tree structure）表示复杂模型（例如人形 humanoid），并用递归绘制整棵结构。

每个节点（treenode）代表一个身体部件，例如 torso（躯干）、head（头）、arm（手臂）等，并包含以下内容：

- m[16]：局部变换矩阵（local transformation matrix）
- f()：绘制函数指针（函数指向如何画这个部件）
- sibling：指向兄弟节点
- child：指向子节点

二、树结构关系

典型人形模型的树结构：

```
Torso
 ├── Head
 ├── LeftUpperArm
 │    └── LeftLowerArm
 └── RightUpperArm
      └── RightLowerArm
```

每个节点的 child 指向第一个子部件，sibling 指向同层下一个部件。

三、构建节点示例

1. 躯干节点（torso_node）：

```
glRotatef(theta[0], 0.0, 1.0, 0.0);
glGetFloatv(GL_MODELVIEW_MATRIX, torso_node.m);
torso_node.f = torso;
torso_node.sibling = NULL;
torso_node.child = &head_node;
```

2. 上臂节点（lua_node）：

```
glTranslatef(-(TORSO_RADIUS+UPPER_ARM_RADIUS), 0.9*TORSO_HEIGHT, 0.0);
glRotatef(theta[3], 1.0, 0.0, 0.0);
glGetFloatv(GL_MODELVIEW_MATRIX, lua_node.m);
lua_node.f = left_upper_arm;
lua_node.sibling = &rua_node;
lua_node.child = &lla_node;
```

四、递归绘制函数 traverse()

```
void traverse(treenode* root)
{
  if(root==NULL) return;
  glPushMatrix();
  glMultMatrixf(root->m);
  root->f();
  if(root->child!=NULL) traverse(root->child);
  glPopMatrix();
  if(root->sibling!=NULL) traverse(root->sibling);
}
```

这个函数会：

1. 保存当前矩阵状态（glPushMatrix）
2. 应用该节点的变换（glMultMatrixf）
3. 调用绘制函数 f()
4. 递归绘制子节点
5. 恢复矩阵状态（glPopMatrix）
6. 再递归绘制兄弟节点

五、执行顺序示意

```
traverse(&torso_node);
```

1. 绘制 torso
2. 绘制 head（child）
3. 绘制 left_upper_arm → left_lower_arm
4. 绘制 right_upper_arm → right_lower_arm

六、总结

这段代码实现了一个通用的层级模型系统：

- 用 treenode 表示每个身体部件；
- 用 m 矩阵保存其局部变换；
- 用 child/sibling 表示树形关系；
- 用 traverse() 递归绘制整个结构。

优点：

- 表示层级关系清晰
- 父节点变换自动影响子节点
- 通用、可扩展，适用于人形、机械臂等模型