实验概述

首先在 `DiskInode` 中增加一个 u32 类型的变量 link，作为硬链接计数用，初始化为1，同时修改 `INODE_DIRECT_COUNT` 使其减少一。

在 `File` trait 中新增一个函数 `fn inode(&self) -> Option<Arc<Inode>>` 默认返回None，但 `OSInode` 要正确返回其 Inode 的引用。

1. sys_linkat

首先判断参数是否正确，旧文件是否存在，新名字长度是否合规等问题。检查通过后，再构建一个新 `DirEntry`，`name` 为新名字，`inode_id` 为旧文件的inode_id，然后通过 `ROOT_INODE` 的 `write_at` 函数将新 entry 写入数据块尾部。之后通过旧文件的 Inode 将硬链接计数加一。

2. sys_unlinkat

首先查找该文件是否存在，如果不存在返回-1。找到该文件后，先在根目录下删除对应的 `DIrEntry` 。由于没有提供指定删除某段内容的方法，为了简化实现，先用 `ls` 方法获取所有项后调用 `clear` 清除所有项，再依次调用 `write_at` 将要保留的项再重新写入。之后通过该文件的 Inode 将硬链接计数减一。当计数为0时，调用 `clear` 清除所有数据，然后调用自己实现的 `EasyFileSystem.dealloc_inode()` 释放 Inode

3. sys_fstat

由于在 `File` trait中增加了 `inode` 方法，可以直接用 fd 获取到对应的文件信息。

问答作业

1. root inode 实际成为了整个文件系统的初始化入口和访问入口，如果root inode中内容损坏，则可能造成整个文件系统不可用。

荣誉准则

1. 在完成本次实验的过程（含此前学习的过程）中，我曾分别与 **以下各位** 就（与本次实验相关的）以下方面做过交流，还在代码中对应的位置以注释形式记录了具体的交流对象及内容：

   > 无

2. 此外，我也参考了 **以下资料** ，还在代码中对应的位置以注释形式记录了具体的参考来源及内容：

   > 无

3. 我独立完成了本次实验除以上方面之外的所有工作，包括代码与文档。 我清楚地知道，从以上方面获得的信息在一定程度上降低了实验难度，可能会影响起评分。

4. 我从未使用过他人的代码，不管是原封不动地复制，还是经过了某些等价转换。 我未曾也不会向他人（含此后各届同学）复制或公开我的实验代码，我有义务妥善保管好它们。 我提交至本实验的评测系统的代码，均无意于破坏或妨碍任何计算机系统的正常运转。 我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的实验成绩将按“-100”分计。

(optional)

对文件系统的管理如果能和多核中的并发与互斥问题结合起来更好。