The basics of programming in C

To do at home

An ext2 file system has a file that is 1024 blocks long. The blocks are located sequentially. One block is 4K bytes long.

How much time will it take to read that file? Assume a typical HDD and consider the following two scenarios:

1. Blocks are read one at a time: map the logical offset within a file to the LBA, read that block, move on to the next one.
2. Read the inode and the block with indirect pointers into the RAM first, make a batch request to read file blocks, and issue a read of 4 megabytes in one request.

```
        if (... any error ...)
            goto cleanup;
    }
cleanup:
    for (int i = 0; i < MAX_INFLIGHT_READS; i++)
        free(read_buffers[i]);
    close(in);
    close(out);
    io_uring_queue_exit(&ring);
```

```
      if (... any error ...)
         goto cleanup;
   }
cleanup:
   for (int i = 0; i < MAX_INFLIGHT_READS; ++i)
      free(read_buffers[i]);
   close(in);
   close(out);
   io_uring_queue_exit(&ring);
```

This releases buffers and closes files before closing an io_uring that references them.

inode.h

int init_inode(…) {…}

inode.h:

```
int init_inode(…) {…}
```

If `inode.h` is included into multiple `.c` files, then `init_inode()` will be duplicated in multiple object files, and linking them will fail.

Header files may define only static inline functions. All other functions may only be declared in headers, and need to be defined in `.c` files.

```
int init_inode(…)
{
    …
    if (inode->i_links_count == 0) {
        ret = -ENOENT;
        goto out;
    }
    …
}
```

```
int init_inode(…)
{
    …
    if (inode->i_links_count == 0) {
        ret = -ENOENT;
        goto out;
    }
    …
}
```

This is a use-after-free.

If you intend to verify whether an inode is allocated, then use the inode bitmap.