Symlink.C

int symlink(char *old_file, char *new_file)

First, I use getino twice to check the old_file does exist and the new_file does not exist. Next, I call creat_file to create the new_file inside the parents data block. I call getino twice more to get ino number of parent to new file and new file and iget to load a parent MINODE (pip) and ~~old~~ new_file minode (mip). I change ~~of~~ newfile minode->INODE Mode to LNK type (0xA000) and use memcpy to copy the contents of the old_file into the new iblock. I set new file size to the size of the old file name, mark new_file minode as dirty and call iput(mip). Finally, I mark new_file's Parents minode as dirty and call iput(pip)

int readlink(char *name, char buffer[], int number)

// use global bool isreadlink to block print statements in getino
First, I get inonumber using getino and ~~call~~ iget to load a MINODE pointer to name. I use S_ISLNK to verify the file is a link type. Then we use memcpy to copy the filename from the iblock[] into the buffer and finally return the size of the file.