```
#include "type.h"

/*
Breakdown pathname into parent path and child path.
Get the ino of that parent pathname.
Load that ino from disk into memory.

Allocate/get inode and populate inode with correct information
Puts inode back onto disk.

Diff from mkdir: mode is REG FILE, no data block size = 0, links_count = 1, dont incr
parent links
*/


void creat_file(char path[124])
{
        int i, ino;
        MINODE *pmip; //mip pointer
        INODE *pip; //inode pointer

        char buf[1024];
        char temp1[1024], temp2[1024];
        char parent_name[1024], child_name[1024];

        strcpy(temp1, path);
        strcpy(temp2, path);

        strcpy(parent_name, dirname(temp1));
        strcpy(child_name, basename(temp2));

        //get parent ino
        ino = get_Inode(running->cwd, parent_name);//get the inode of the current
parent directory
        pmip = iget(dev, ino);//get the memory inode corresuper_ponding to the parent
directory
        pip = &pmip->INODE;//set the parent inode pointer to the memory inode pointer
INODE member

        if(!pmip)//parent pointer doesnt exist
        {
                printf("Error, the parent does not exist.\n");
                return;
        }

        if(!S_ISDIR(pip->i_mode))//checks to see if the parent inode is a file or
directory
        {
                printf("Error, the parent is not a directory!\n");//if its a file
print error then return
                return;
        }

        if(get_Inode(running->cwd, path) != 0)//get_Inode should return zero since we
are hoping that the file does not exist, if it returns anything other than zero then
error
        {
                printf("Error, %s already exists!\n", path);
                return;
        }

        creat_helper(pmip, child_name); //create the file with the name and correct
memory inode parent

        pip->i_links_count++; //increment the parent inode line count
        pip->i_atime = time(0L);//set the parent inode access time
        pmip->dirty = 1; //set parent memory inode pointer dirty to 1

        iput(pmip);//disuper_pose of the parent memory inode pointer

        return;
}

int creat_helper(MINODE *pmip, char *child_name)
{
        int i;
        int ino = ialloc(dev);//allocate an inode

        MINODE *mip = iget(dev, ino);//get the memory pointer for that inode
        INODE *ip = &mip->INODE;//set the inode pointer to that memory inode

        ip->i_mode = 0x81A4; //set file type
        ip->i_uid  = running->uid; //set owner uid
        ip->i_gid  = running->gid; //set group Id
        ip->i_size = 0; //set the size to 0 because it is an empty file
        ip->i_links_count = 1; //set links to one because parent directory
        ip->i_atime = time(0L); //set last access to current time
        ip->i_ctime = time(0L); //set last change to current time
        ip->i_mtime = time(0L); //set last modify to current time

        ip->i_blocks = 0;//set the number of data blocks to 0
        for(i = 0; i < 15; i++)//loop through and set each index to zero, meaning it
is unoccupied
                ip->i_block[i] = 0;

        mip->dirty = 1;//set the memory inode dirty to one
        iput(mip);//dispose of the memory inode pointer

        enter_name(pmip, ino, child_name); //calls enter name on the newly created
inode
        return ino;
}
```