```c
#include "type.h"

/*
Gets minode of parent in pathname
Gets minode of child
Then removes dir entry of child from parent

if first/middle, remove dir entry, move all entires after left rec_len of removed,
and add rec_len to last
if last, remove dir entry, add rec_len to previous dir entry
if only, deallocate data block, modify parent's file size, move all subsequent data
blocks left one
*/

void my_unlink(char *path)
{
        int ino, i;
        int parent_ino;

        MINODE *mip;
        MINODE *p_mip;
        INODE *ip;
        INODE *parent_ip;

        char temp1[64], temp2[64];
        char my_dirname[64];
        char my_basename[64];

        strcpy(temp1, path);
        strcpy(temp2, path);
        strcpy(my_dirname, dirname(temp1));
        strcpy(my_basename, basename(temp2));

        if(!path)//checks to see if the path is invalid
        {
                printf("Error, no file name given.\n");
                return;
        }

        ino = get_Inode(running->cwd, path);//gets the inode of the file that we want
to unlink

        if(ino == 0)//checks to see if the file could be found
        {
                printf("error, file does not exist.\n");
                return;
        }

        mip = iget(dev, ino);//get the memory inode of the file we want to unlink

        if(!mip)//checks to see if the mip was found
        {
                printf("Error, missing mip.\n");
                return;
        }

        if(S_ISDIR(mip->INODE.i_mode))//checks to see if it is directory
        {
                printf("Error, can't unlink a directory.\n");
                return;
        }
```

```
        ip = &mip->INODE;


        ip->i_links_count--; //decrease the link count


        for(i = 0; i < 12 && ip->i_block[i] != 0; i++)//walk through the block to
deallocate
                bdealloc(dev, ip->i_block[i]); //deallocate blocks


        idealloc(dev, ino);//deallocate the inode



        //removes file from parent
        parent_ino = get_Inode(running->cwd, my_dirname);//get the parent inode
        p_mip = iget(dev, parent_ino);//get the parent memory inode
        parent_ip = &p_mip->INODE;//set the parent inode to the parent memory inode


        rm_child(p_mip, my_basename);//removes the child

        //update stats on the parent inode
        parent_ip->i_links_count--;
        parent_ip->i_atime = time(0L);
        parent_ip->i_mtime = time(0L);
        p_mip->dirty = 1;
        iput(p_mip);//disuper_pose of the parent inode
        mip->dirty = 1;
        iput(mip);//disuper_poses of the memory inode

        return;
}
```