mkdir_creat.c

int make_dir(char *pathname)

First checks if the pathname is in root directory or current running directory. Then breaks the pathname into parent and child and loads the parent's inode information into a MINODE (pip).
Next we check that the parent minode is a Directory and use our search function to make sure the child doesn't already exist.
Finally, we call mymkdir(pip, child) and increment parents link count, update the time, and mark as dirty to flag iput to write the parent minode back to the disk.

t mymkdir(MINODE *pip, char *name)

First allocate an INODE and disk block (ialloc, balloc).
Then load the new ino number into a MINODE (mip) and set mip->INODE information to Directory specifics.
Then mark mip as dirty and call iput to write back new INODE to disk. Next we create . and .. directories inside of directory we just created and write them back to the disk.
Finally, call enter_name(pip, ino, name) to enter the dir to the parent INODE with correct length and in the correct position.

int enter_name (MINODE *pip, int myino, char *myname)

First We calculate the need length for our new entry (myname). Next we step through all the direct blocks. For each block, we get the contents into a buffer and step to the end (last entry) of the block. We keep an int ideal_length updating each iteration and by the end will contain the ideal length of the last entry in the block. We calculate the remaining length of the block left to see if we have enough space to enter the name. If remaining length is greater than or equal to the need length of our entry, we trim previous last entry to its ideal length, advance 1 more time by that length to get to the new end, and insert our entry info. Then put the block back to the disk. If remaining length is less than need length of our entry, we must allocate a new block and attach it to the parent. Then we enter our entry as the first in the new block and put the block back to the disk.

int my_creat (MINODE *pip, char *name)

This Algorithm is exactly the same as mymkdir except the INODE's mode is set to file type (0X81A4), links count is only 1 and the size is 0. Also don't inc parent's link count. And No Data block is allocated, thus . and .. aren't created.

int creat_file (char *pathname)

This Algorithm is exactly the same as make_dir except INODE mode is set to regular file type, size is set to 0, links count is only 1. Also don't inc parent's link count.