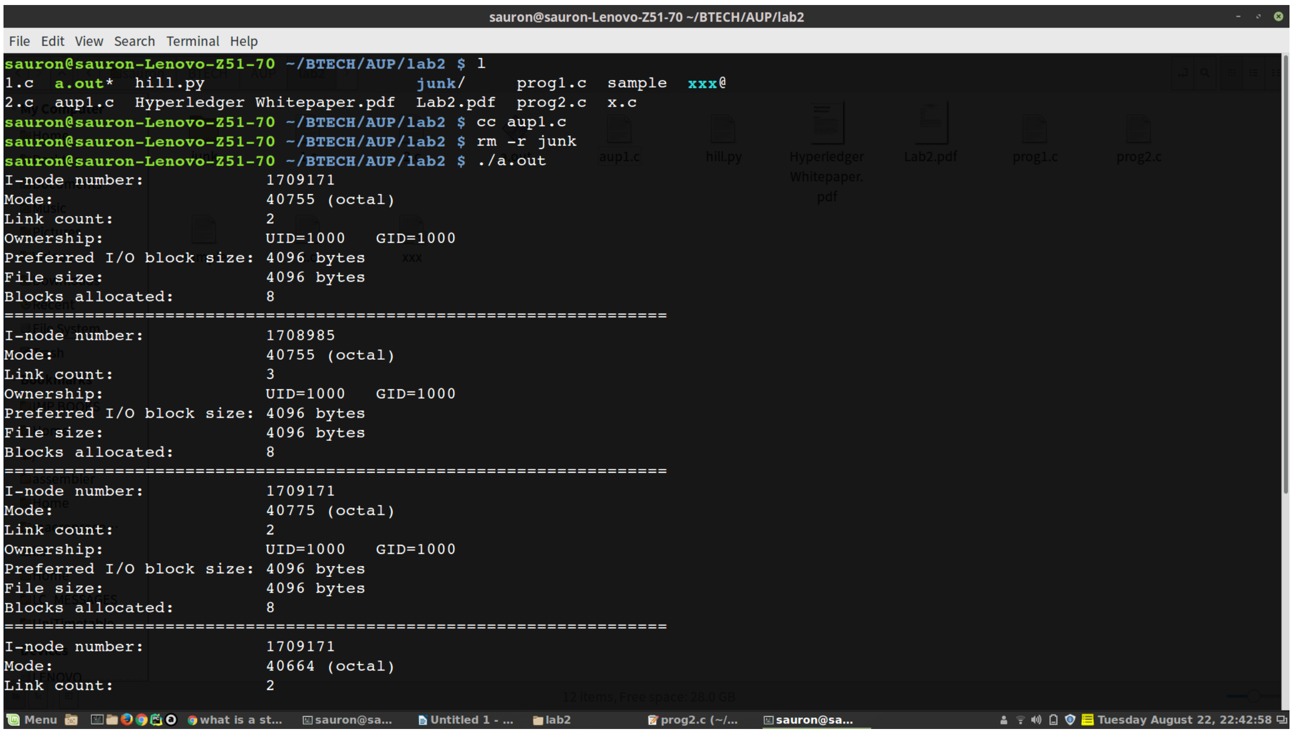
111403003 Akash Sarda
111403023 Aditya Malu
111403019 Jassim Abdul Rehman

LAB 2.

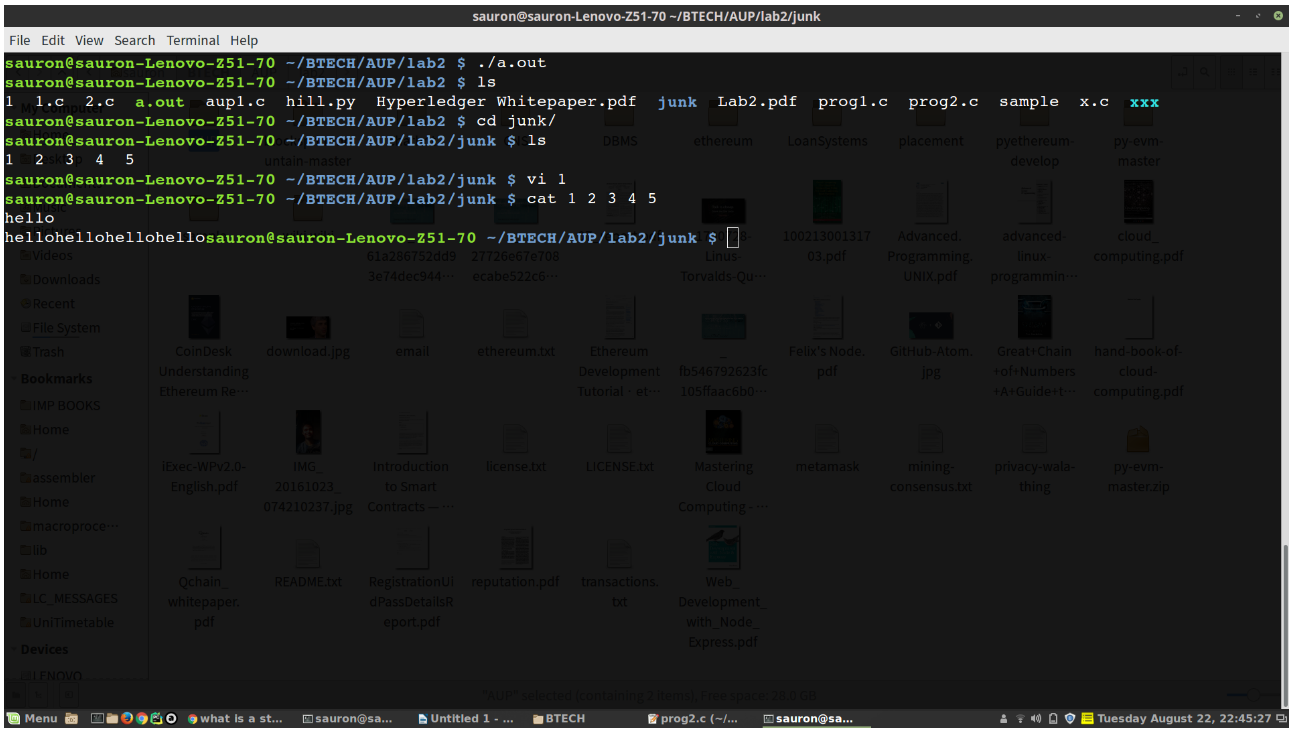
Code 1:

```
#include <stdio.h>
#include <sys/stat.h>
#include <string.h>
#include<stdlib.h>

void ls (char *pathname) {
        struct stat sb = {0};
        int s = stat(pathname, &sb);
        printf("I-node number:           %ld\n", (long) sb.st_ino);
        printf("Mode:                    %lo (octal)\n",(unsigned long) sb.st_mode);
        printf("Link count:              %ld\n", (long) sb.st_nlink);
        printf("Ownership:               UID=%ld   GID=%ld\n",               (long) sb.st_uid, (long)
sb.st_gid);
        printf("Preferred I/O block size: %ld bytes\n", (long) sb.st_blksize);
        printf("File size:               %lld bytes\n",(long long) sb.st_size);
        printf("Blocks allocated:        %lld\n", (long long) sb.st_blocks);

printf("=============================================================
===\n");

}

int main() {
            mkdir("junk", 0775);
            int i;
            FILE *fp;
            char *dest1 = "./junk/";
            char temp[20];
            char dest2[20];
            for(i = 1; i <= 5; i++) {
                        sprintf(dest2, "%d", i);
                        strcpy(temp, dest1);
                        strcat(temp, dest2);
                        fp = fopen(temp, "w+");
                        fwrite("hello" , 1 , 5 , fp );
                        fclose(fp);
            }
            ls("./junk");
            char mode[100];
        strcpy(mode, "0331");
        char buf[100] = "junk";
```

```
    i = strtol(mode, 0, 8);
    chmod (buf,i);
    ls("./");

    strcpy(mode , "0775");
    i = strtol(mode, 0, 8);
    chmod(buf,i);
    ls("./junk");

    strcpy(mode , "0664");
    i = strtol(mode, 0, 8);
    chmod(buf,i);
    ls("./junk");

    strcpy(mode , "0775");
    i = strtol(mode, 0, 8);
    chmod(buf,i);
    ls("./junk");

    return 0;
}
```


sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2
File Edit View Search Terminal Help

```
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2 $ l
1.c  a.out*  hill.py                    junk/     prog1.c  sample  xxx@
2.c  aup1.c  Hyperledger Whitepaper.pdf  Lab2.pdf  prog2.c  x.c
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2 $ cc aup1.c
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2 $ rm -r junk
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2 $ ./a.out
I-node number:            1709171
Mode:                     40755 (octal)
Link count:               2
Ownership:                UID=1000   GID=1000
Preferred I/O block size: 4096 bytes
File size:                4096 bytes
Blocks allocated:         8
==================================================================
I-node number:            1708985
Mode:                     40755 (octal)
Link count:               3
Ownership:                UID=1000   GID=1000
Preferred I/O block size: 4096 bytes
File size:                4096 bytes
Blocks allocated:         8
==================================================================
I-node number:            1709171
Mode:                     40775 (octal)
Link count:               2
Ownership:                UID=1000   GID=1000
Preferred I/O block size: 4096 bytes
File size:                4096 bytes
Blocks allocated:         8
==================================================================
I-node number:            1709171
Mode:                     40664 (octal)
Link count:               2
```



```
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2 $ ./a.out
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2 $ ls
1  1.c  2.c  a.out  aup1.c  hill.py  Hyperledger Whitepaper.pdf  junk  Lab2.pdf  prog1.c  prog2.c  sample  x.c  xxx
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2 $ cd junk/
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2/junk $ ls
1  2  3  4  5
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2/junk $ vi 1
sauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2/junk $ cat 1 2 3 4 5
hello
hellohellohellohellosauron@sauron-Lenovo-Z51-70 ~/BTECH/AUP/lab2/junk $ 
```

Code 2:

```
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <fcntl.h>
#include <unistd.h>
#include <dirent.h>
#include <string.h>
#include <time.h>
#include <pwd.h>
#include <grp.h>

int main(int argc, char *argv[]){
        int r, n;
        char dname[16], ftmp[16], fname[16], linkpar[64];
        struct stat st;
        mode_t mode;
        DIR *dir;
        struct dirent *ent;

        if(argc > 1)
                strcpy(dname, argv[1]);
        else
                strcpy(dname, ".");

        strcpy(fname, dname);
        strcat(fname, "/");
```

```
	dir = opendir(dname);
	if(!dir){
		printf("Error opening directory\n");
		return 0;
	}
	while((ent = readdir(dir)) != NULL){
		if(strcmp(ent->d_name, ".") != 0 && strcmp(ent->d_name, "..") != 0){
			strcpy(ftmp, fname);
			strcat(ftmp, ent->d_name);
			lstat(ftmp, &st);

			if(S_ISLNK(st.st_mode)){
				printf("%s : ", ftmp);
				while(S_ISLNK(st.st_mode)){

					r = readlink(ftmp, linkpar, 63);
					if(r == -1){
						printf("Link broken\n");
						break;
					}
					linkpar[r] = '\0';

					strcpy(ftmp, linkpar);
					lstat(ftmp, &st);
				}
				printf("%s\n", ftmp);
			}
		}
	}
}
```

problem 3:

yes it is possible by enabling the sticky bit of the directory.
A Sticky bit is a permission bit that is set on a file or a directory that lets only the owner of the file/directory or the root user to delete or rename the file. No other user is given privileges to delete the file created by some other user.