AUP LAB 2
Nikhil Gawande 111408013
Anupam Godse 111408016

Q1.1. "Using APIs, write the equivalent program for the following shell script.

```
mkdir junk
for i in 1 2 3 4 5 do
echo hello >junk/$i
done
ls –l junk
chmod –r junk
ls –l
chmod +r junk
ls –l junk
chmod –x junk
cd junk
chmod +x junk
cd junk
```

Write appropriate comments in the program to observe the execution output.

Program:

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#include<fcntl.h>
#include<ctype.h>
#include<dirent.h>
#include<string.h>
#include<time.h>
void list(char *path){
        struct stat fileStat;
        DIR *d;
        struct dirent *dir;
        printf("%s ",path);
        d = opendir(path);
        int flag = 1;
        if(d){
                while((dir = readdir(d)) != NULL){
                        flag = 0;
                        if(strcmp(dir->d_name,".") != 0 && strcmp(dir->d_name,"..") != 0){
                                char new_path[128];
                                strcpy(new_path,path);
                                strcat(new_path,"/");
                                strcat(new_path,dir->d_name);
                                printf("%s ",dir->d_name);
                                if(stat(new_path,&fileStat) >= 0){
```

```
                                printf((S_ISDIR(fileStat.st_mode)) ? "d" : "-");
                                printf((fileStat.st_mode & S_IRUSR) ? "r" : "-");
                                printf((fileStat.st_mode & S_IWUSR) ? "w" : "-");
                                printf((fileStat.st_mode & S_IXUSR) ? "x" : "-");
                                printf((fileStat.st_mode & S_IRGRP) ? "r" : "-");
                                printf((fileStat.st_mode & S_IWGRP) ? "w" : "-");
                                printf((fileStat.st_mode & S_IXGRP) ? "x" : "-");
                                printf((fileStat.st_mode & S_IROTH) ? "r" : "-");
                                printf((fileStat.st_mode & S_IWOTH) ? "w" : "-");
                                printf((fileStat.st_mode & S_IXOTH) ? "x" : "-");
                                printf(" %lu",fileStat.st_nlink);
                                printf(" %ld",fileStat.st_size);
                                printf(" %s",ctime(&fileStat.st_atime));
                        }
                }
        }
        if(flag == 1) {
                printf("cannot read directory : read failed\n");
        }
        closedir(d);
    }
    else {
        printf("cannot open directory : open failed\n");
    }
}
void makedir(char *path){
    struct stat st = {0};
    DIR *d;
    struct dirent *dir;
    int i;
    if(stat(path,&st) == -1){
        mkdir(path,0700);
        printf("directory created..\n\n");
    }
    else
        printf("Directory already exists..\n\n");
    d = opendir(path);
    if(d){
        for(i = 0; i < 5; i++){
            char *he = "hello";
            char new_path[128];
            char str[3];
            sprintf(str,"%d",i);
            strcpy(new_path,path);
            strcat(new_path,"/");
            strcat(new_path,str);
            int fd1 = open(new_path, O_RDWR | O_CREAT, 0777);
            write(fd1,he,5);
            close(fd1);
        }
        closedir(d);
    }
```

```
}
int main(int argc, char *argv[]){
	struct stat st = {0};
	DIR *d;
	struct dirent *dir;
	int i;
	char path[128];
	strcpy(path,argv[1]);
	strcat(path,"/junk");
	makedir(path);
	printf("listing files using list()\n");
	list(path);
	chmod(path, 0333);
	printf("\nread permission removed for /junk\n");
	printf("listing files using list()\n");
	list(path);
	chmod(path, 0777);
	printf("\nread permission added for /junk\n");
	printf("listing files using list()\n");
	list(path);
	chmod(path, 0666);
	printf("\nexecute permission removed for /junk\n");
	printf("trying to change directory to /junk\n");
	if(chdir(path) == 0) {
		printf("chdir successfull\n");
	}
	else {
		printf("chdir failed\n");
	}
	chmod(path, 0777);
	printf("\nexecute permission added for /junk\n");
	printf("trying to change directory to /junk\n");
	if(chdir(path) == 0) {
		printf("chdir successfull\n");
	}
	else {
		printf("chdir failed\n");
	}
	return 0;
}
```

Execution :

```
student@ac-computer-lab-pc:~/AUP-master/Lab2$
student@ac-computer-lab-pc:~/AUP-master/Lab2$ cc a2q1.c -o a
student@ac-computer-lab-pc:~/AUP-master/Lab2$ ./a .
Directory already exists..

listing files using list()
./junk 2 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017
3 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017
1 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017
0 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017
4 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017

read permission removed for /junk
listing files using list()
./junk cannot open directory : open failed

read permission added for /junk
listing files using list()
./junk 2 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017
3 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017
1 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017
0 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017
4 -rwxrwxr-x 1 5 Fri Oct 13 12:40:52 2017

execute permission removed for /junk
trying to change directory to /junk
chdir failed

execute permission added for /junk
trying to change directory to /junk
chdir successfull
student@ac-computer-lab-pc:~/AUP-master/Lab2$
```

2. A function realpath() resolves all symbolic links in path and returns the ultimate target. Write a program to list the ultimate target of the only filenames that are symbolic links in a directory. The program takes one optional argument, which is the name of a directory to be searched for the links. When no argument is specified, the search is conducted in the current working directory. Display appropriate error messages.

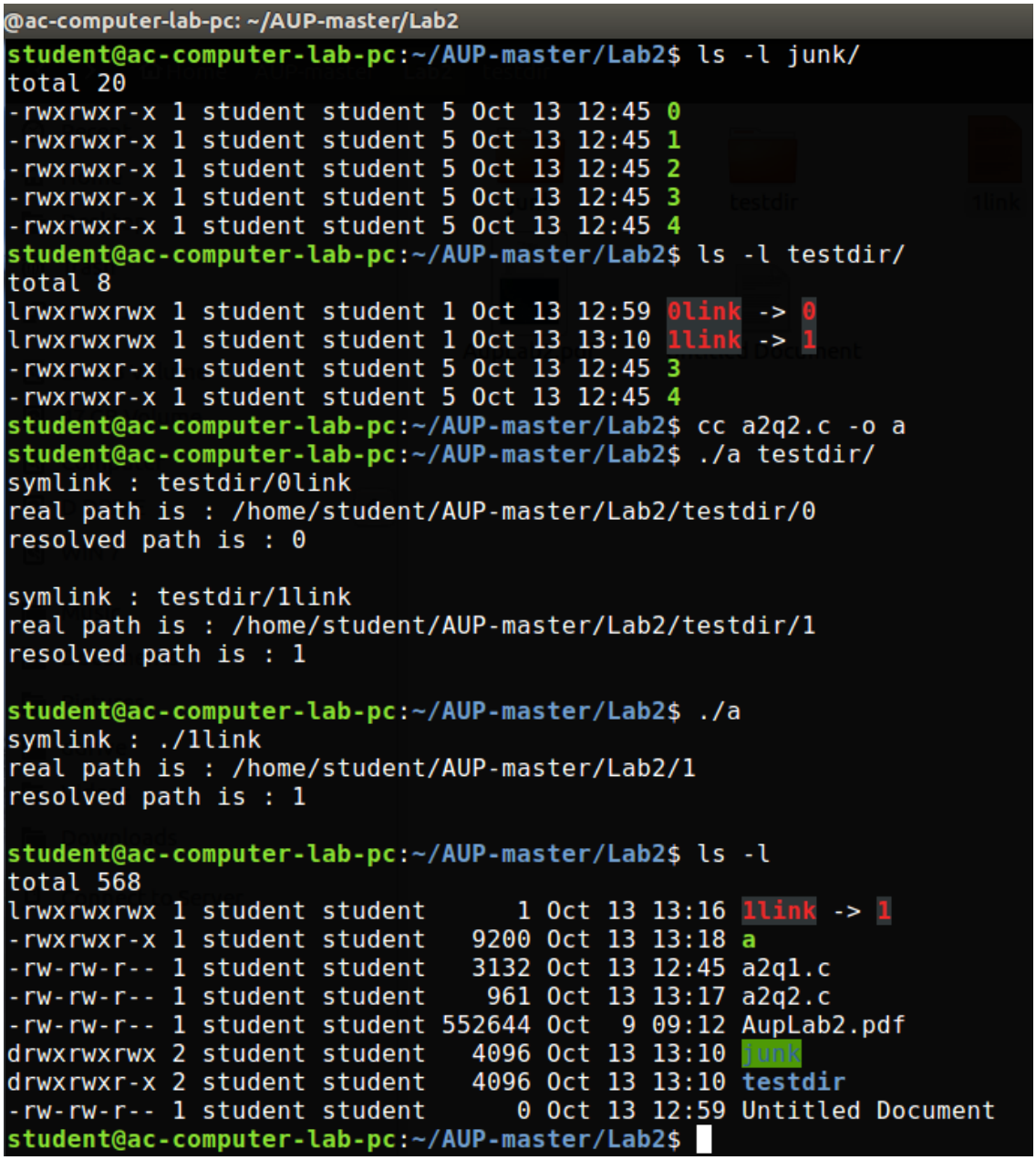
Program :

```
#include<unistd.h>
#include<stdio.h>
#include<sys/stat.h>
#include<sys/types.h>
#include<fcntl.h>
#include<ctype.h>
#include<dirent.h>
#include<string.h>
#include<stdlib.h>
int main(int argc, char *argv[]){
        struct stat fileStat;
```

```
        DIR *d;
        struct dirent *dir;
        char path[128];
        char buff[128];
        if(argc > 1)
                strcpy(path,argv[1]);
        else
                strcpy(path,"./");
        d = opendir(path);
        if(d){
                while((dir = readdir(d)) != NULL){
                        if(strcmp(dir->d_name,".") != 0 && strcmp(dir->d_name,"..") != 0){
                                char new_path[128];
                                strcpy(new_path, path);
                                //strcat(new_path, "/");
                                strcat(new_path, dir->d_name);
                                lstat(new_path, &fileStat);

                                if(S_ISLNK(fileStat.st_mode)){
                                        printf("symlink : %s\n", new_path);
                                        realpath(new_path, buff);
                                        printf("real path is : %s\n", buff);
                                        int temp = readlink(new_path, buff, 127);
                                        buff[temp] = '\0';
                                        printf("resolved path is : %s\n\n", buff);
                                }
                        }
                }
                closedir(d);
        }
        return 0;
}
```

Execution :



3. Create a shared directory for usage with a purpose that any user (not super user) can create new files in this directory, but only the owner can delete his own files and everyone else can read all files?

```
alice@anupam-Inspiron-7548:~$ ls
examples.desktop
alice@anupam-Inspiron-7548:~$ mkdir testdir
alice@anupam-Inspiron-7548:~$ ls -l testdir/
total 0
alice@anupam-Inspiron-7548:~$ ls -l
total 16
-rw-r--r-- 1 alice alice 8980 Oct 13 13:24 examples.desktop
drwxrwxr-x 2 alice alice 4096 Oct 13 13:29 testdir
alice@anupam-Inspiron-7548:~$ chmod 1777 testdir/
alice@anupam-Inspiron-7548:~$ ls -l
total 16
-rw-r--r-- 1 alice alice 8980 Oct 13 13:24 examples.desktop
drwxrwxrwt 2 alice alice 4096 Oct 13 13:29 testdir
alice@anupam-Inspiron-7548:~$ nano testdir/alicefile
```

```
alice@anupam-Inspiron-7548:~$ cat testdir/alicefile
Search your computer
alice@anupam-Inspiron-7548:~$ su - bob
Password:
bob@anupam-Inspiron-7548:~$ cat /home/alice/testdir/alicefile
hi my name is alice.
bob@anupam-Inspiron-7548:~$ rm /home/alice/testdir/alicefile
rm: remove write-protected regular file '/home/alice/testdir/alicefile'? yes
rm: cannot remove '/home/alice/testdir/alicefile': Operation not permitted
bob@anupam-Inspiron-7548:~$ su - alice
Password:
alice@anupam-Inspiron-7548:~$ rm testdir/alicefile
alice@anupam-Inspiron-7548:~$ ls -l
total 16
-rw-r--r-- 1 alice alice 8980 Oct 13 13:24 examples.desktop
drwxrwxrwt 2 alice alice 4096 Oct 13 13:32 testdir
alice@anupam-Inspiron-7548:~$ ls -l testdir/
total 0
alice@anupam-Inspiron-7548:~$
```