Nithilam Subbaian
ECE-357: Computer Operating Systems
Prof. Hakner

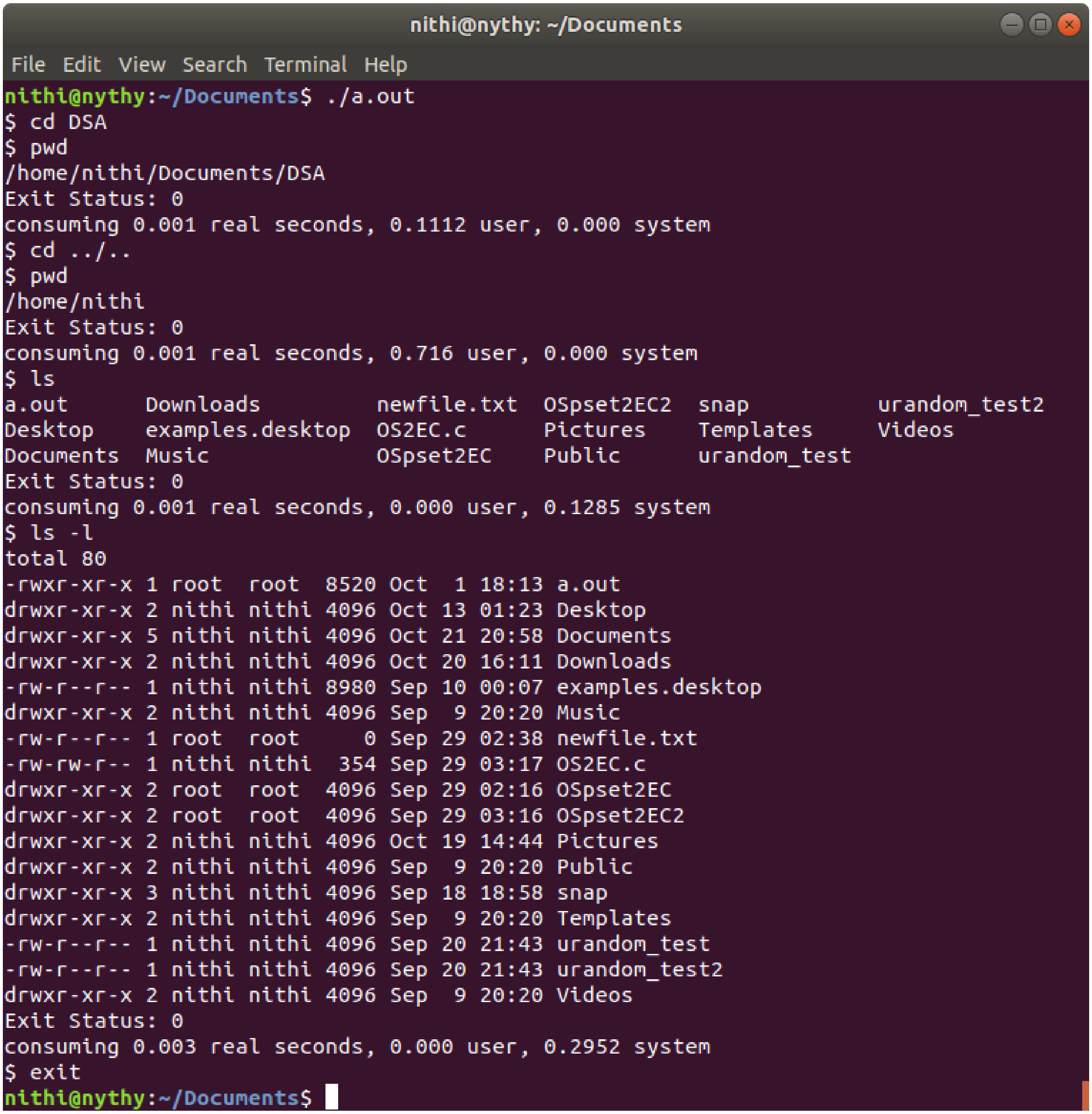
PSET 3: Problem 2 – Simple Shell Program

Source code:

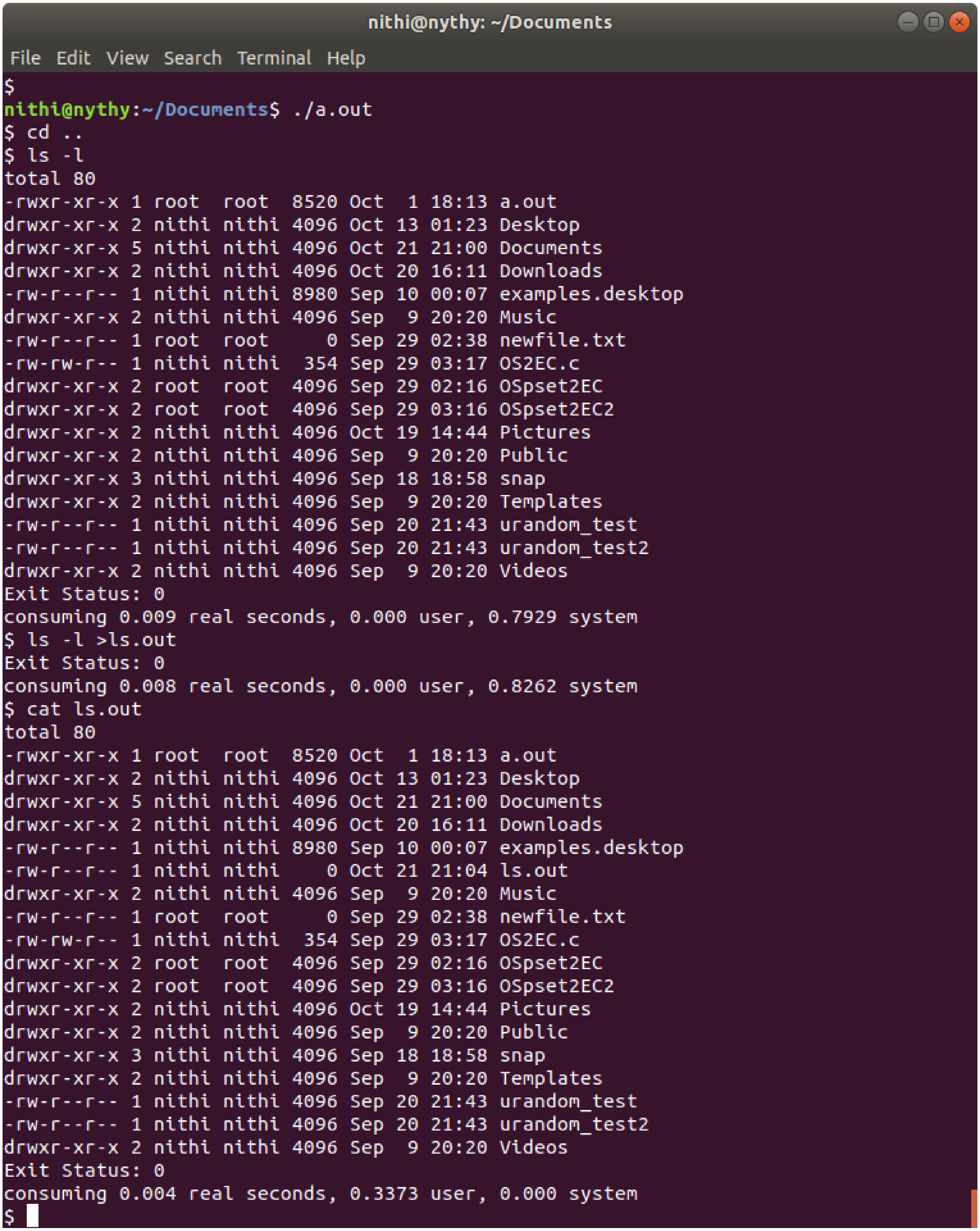
```
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <dirent.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/stat.h>
#include <sys/wait.h>
#include <sys/resource.h>

int status = -1;
char* command[BUFSIZ];
char* tokenelement[BUFSIZ];

int isIORedir(char* tokenelement){
	if (tokenelement[0] == '<') {
		return 1;
	} else if(tokenelement[0] == '>' && tokenelement[1] == '>') {
		return 4;
	} else if (tokenelement[0] == '>') {
		return 2;
	} else if(tokenelement[0] == '2' && tokenelement[1] =='>' && tokenelement[2] == '>') {
		return 5;
	} else if (tokenelement[0] == '2' && tokenelement[1] =='>') {
		return 3;
	} else{
		return 0;
	}
}


int opendupclose(int i, int offset, int flags, char* mode, int std_fd, char* std_stream){
	char* filename;
	int fd;
	filename = &tokenelement[i][offset];
```

```
    if ((fd = open(filename, flags, 0666))<0) {
        fprintf(stderr, "%d\n", flags);
        fprintf(stderr, "ERROR: Could not open file %s for %s: %s\n", filename, mode,
strerror(errno) );
        return -1;
    }
    if (dup2(fd, std_fd)<0) {
        fprintf(stderr, "ERROR: could not dup2 %s to %s: %s\n",filename, std_stream,
strerror(errno) );
        return -1;
    }
    if (close(fd)!=0) {
        fprintf(stderr, "ERROR: Could not close file '%s': %s\n", filename, strerror(errno) );
        return -1;
    }
}


int parameterchange(){
    char* std_stream;
    int std_fd, offset, flags;
    int result = 0;

    for(int i = 0; tokenelement[i] != NULL; i++) {
        switch (isIORedir(tokenelement[i])) {
        case 1:
            result = opendupclose(i, 1,O_RDONLY, "reading", 0, "stdin");
            break;
        case 2:
            result = opendupclose(i, 1,O_RDWR|O_TRUNC|O_CREAT, "writing", 1, "stdout");
            break;
        case 3:
            result = opendupclose(i, 2,O_RDWR|O_TRUNC|O_CREAT, "writing", 2, "stderr");
            break;
        case 4:
            result = opendupclose(i, 2,O_RDWR|O_APPEND|O_CREAT, "writing", 1, "stdout");
            break;
        case 5:
            result = opendupclose(i, 3,O_RDWR|O_APPEND|O_CREAT, "writing", 2, "stderr");
            break;
        default:
            return -1;
        }
    }
    return result;
```

```
}

int lineParse(char* line, FILE* input){
    char* d = " \t\n";
    char *token = strtok(line, d);
    int nonIOcount = 0;
    int IOredirectioncount = 0;
    struct rusage ru;
    struct timeval start, end;

    while(token != NULL) {
        if (isIORedir(token) == 0) {
            command[nonIOcount++] = token;
        }else if ( isIORedir(token) != 0) {
            tokenelement[IOredirectioncount++] = token;
        }
        token = strtok(NULL, d);
    }
    if (strcmp(command[0], "cd")==0) {
        if(command[1] == NULL) {
            fprintf(stderr, "ERROR: Could not change directory because no path was specified\n");
            return -1;
        } else if(chdir(command[1])<0) {
            fprintf(stderr, "ERROR: Could not change directory to %s: %s\n", command[1],
strerror(errno));
            return -1;
        }
    }else if(strcmp(command[0], "exit")==0) {
        if (command[1] == NULL) {
            _exit(status);
        }else{
            _exit(atoi(command[1]));
        }
    } else if ( command[0][0] != '#') {
        int pid;
        gettimeofday(&start, NULL);

        switch(pid = fork()) {

        case 0:
            if (parameterchange()<0) {
                fprintf(stderr, "ERROR: Could not redirect IO and therefore command could not be
executed\n");
                _exit(-1);
            }
            if(input!=stdin) {
                fclose(input);
            }
            if (execvp(command[0], command)<0) {
                fprintf(stderr, "ERROR: Could not execute command '%s':%s\n", command[0],
strerror(errno));
                _exit(-1);
            }

        case -1:
            fprintf(stderr, "ERROR: Could not succesfully fork: %s\n", strerror(errno));
            break;

        default:
            if (wait3(&status, 0, &ru) < 0) {
                fprintf(stderr, "ERROR: Could not get information on child process: %s\n",
strerror(errno) );
            } else{
                gettimeofday(&end, NULL);
                double elapsed = (end.tv_sec - start.tv_sec) +
                        ((end.tv_usec - start.tv_usec)/1000000.0);
                fprintf(stderr, "Exit Status: %i\n",WEXITSTATUS(status));
                fprintf(stderr, "consuming %.3f real seconds, %ld.%.3ld user, %ld.%.3ld system\n",
                    elapsed, ru.ru_utime.tv_sec, ru.ru_utime.tv_usec,
                    ru.ru_stime.tv_sec, ru.ru_stime.tv_usec);
            }
            break;
        }
    }

    for (int i = 0; i < nonIOcount; i++) {
        command[i] = NULL;
    }
    for (int j = 0; j< IOredirectioncount; j++) {
        tokenelement[j]=NULL;
    }

    return 0;
}


int main(int argc, char** argv){
    FILE *input;
    size_t n = 0;
    int alive = 1;
    if(argc == 1) {
        input = stdin;
    } else {
        if ((input = fopen(argv[1], "r") )<0) {
            fprintf(stderr,"ERROR: Could not open file %s: %s", argv[1], strerror(errno));
            return -1;
        }
    }
    while(alive) {
        printf("$ ");
        char* linebuffer = NULL;
        if (getline(&linebuffer, &n, input) != -1) {
            if(strcmp(linebuffer, "\n") == 0) {
                continue;
            }
            lineParse(linebuffer, input);
        } else if (feof(input) == 0) {
            fprintf(stderr, "ERROR: Could not read command from stdin: %s\n", strerror(errno));
        } else{
            break;
        }
    }
    printf("\n");
    _exit(status);
    return 0;
}
```



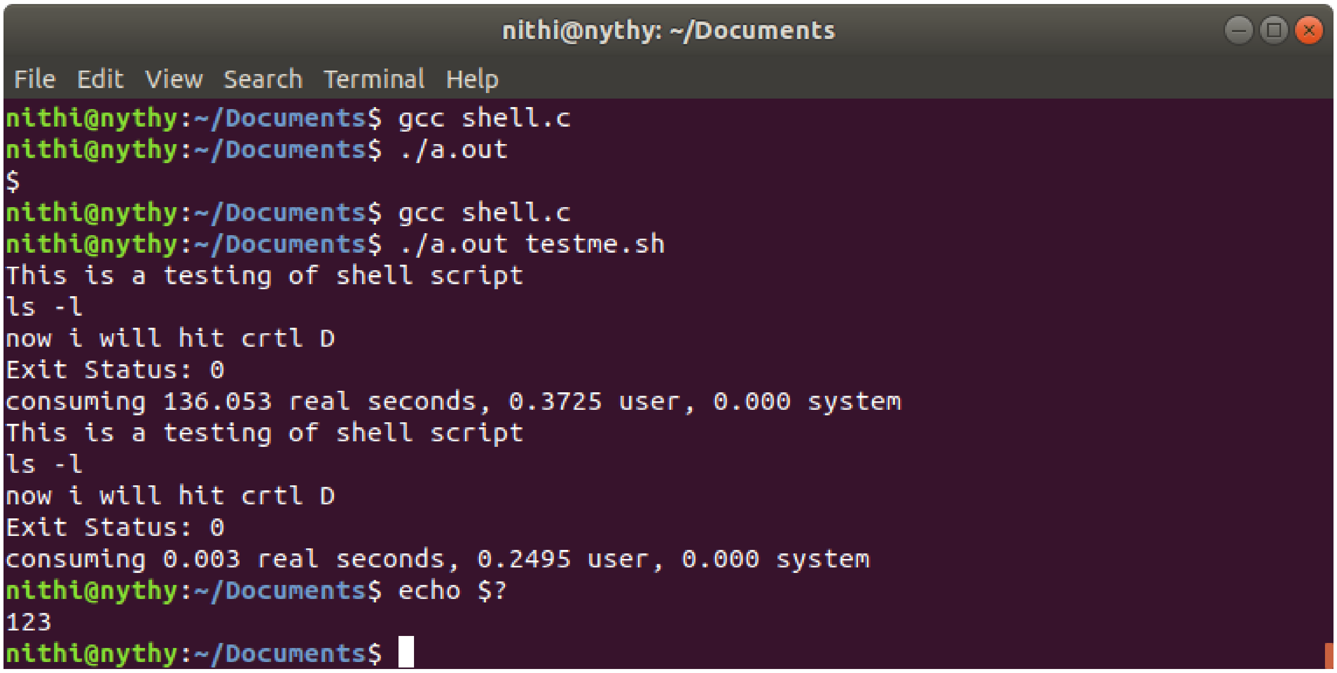
```
nithi@nythy:~/Documents$ ./a.out
$ cd DSA
$ pwd
/home/nithi/Documents/DSA
Exit Status: 0
consuming 0.001 real seconds, 0.1112 user, 0.000 system
$ cd ../..
$ pwd
/home/nithi
Exit Status: 0
consuming 0.001 real seconds, 0.716 user, 0.000 system
$ ls
a.out      Downloads         newfile.txt  OSpset2EC2  snap         urandom_test2
Desktop    examples.desktop  OS2EC.c      Pictures    Templates    Videos
Documents  Music             OSpset2EC    Public      urandom_test
Exit Status: 0
consuming 0.001 real seconds, 0.000 user, 0.1285 system
$ ls -l
total 80
-rwxr-xr-x 1 root  root  8520 Oct  1 18:13 a.out
drwxr-xr-x 2 nithi nithi 4096 Oct 13 01:23 Desktop
drwxr-xr-x 5 nithi nithi 4096 Oct 21 20:58 Documents
drwxr-xr-x 2 nithi nithi 4096 Oct 20 16:11 Downloads
-rw-r--r-- 1 nithi nithi 8980 Sep 10 00:07 examples.desktop
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Music
-rw-r--r-- 1 root  root     0 Sep 29 02:38 newfile.txt
-rw-rw-r-- 1 nithi nithi  354 Sep 29 03:17 OS2EC.c
drwxr-xr-x 2 root  root  4096 Sep 29 02:16 OSpset2EC
drwxr-xr-x 2 root  root  4096 Sep 29 03:16 OSpset2EC2
drwxr-xr-x 2 nithi nithi 4096 Oct 19 14:44 Pictures
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Public
drwxr-xr-x 3 nithi nithi 4096 Sep 18 18:58 snap
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Templates
-rw-r--r-- 1 nithi nithi 4096 Sep 20 21:43 urandom_test
-rw-r--r-- 1 nithi nithi 4096 Sep 20 21:43 urandom_test2
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Videos
Exit Status: 0
consuming 0.003 real seconds, 0.000 user, 0.2952 system
$ exit
nithi@nythy:~/Documents$ 
```



```
$
nithi@nythy:~/Documents$ ./a.out
$ cd ..
$ ls -l
total 80
-rwxr-xr-x 1 root  root  8520 Oct  1 18:13 a.out
drwxr-xr-x 2 nithi nithi 4096 Oct 13 01:23 Desktop
drwxr-xr-x 5 nithi nithi 4096 Oct 21 21:00 Documents
drwxr-xr-x 2 nithi nithi 4096 Oct 20 16:11 Downloads
-rw-r--r-- 1 nithi nithi 8980 Sep 10 00:07 examples.desktop
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Music
-rw-r--r-- 1 root  root     0 Sep 29 02:38 newfile.txt
-rw-rw-r-- 1 nithi nithi  354 Sep 29 03:17 OS2EC.c
drwxr-xr-x 2 root  root  4096 Sep 29 02:16 OSpset2EC
drwxr-xr-x 2 root  root  4096 Sep 29 03:16 OSpset2EC2
drwxr-xr-x 2 nithi nithi 4096 Oct 19 14:44 Pictures
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Public
drwxr-xr-x 3 nithi nithi 4096 Sep 18 18:58 snap
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Templates
-rw-r--r-- 1 nithi nithi 4096 Sep 20 21:43 urandom_test
-rw-r--r-- 1 nithi nithi 4096 Sep 20 21:43 urandom_test2
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Videos
Exit Status: 0
consuming 0.009 real seconds, 0.000 user, 0.7929 system
$ ls -l >ls.out
Exit Status: 0
consuming 0.008 real seconds, 0.000 user, 0.8262 system
$ cat ls.out
total 80
-rwxr-xr-x 1 root  root  8520 Oct  1 18:13 a.out
drwxr-xr-x 2 nithi nithi 4096 Oct 13 01:23 Desktop
drwxr-xr-x 5 nithi nithi 4096 Oct 21 21:00 Documents
drwxr-xr-x 2 nithi nithi 4096 Oct 20 16:11 Downloads
-rw-r--r-- 1 nithi nithi 8980 Sep 10 00:07 examples.desktop
-rw-r--r-- 1 nithi nithi    0 Oct 21 21:04 ls.out
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Music
-rw-r--r-- 1 root  root     0 Sep 29 02:38 newfile.txt
-rw-rw-r-- 1 nithi nithi  354 Sep 29 03:17 OS2EC.c
drwxr-xr-x 2 root  root  4096 Sep 29 02:16 OSpset2EC
drwxr-xr-x 2 root  root  4096 Sep 29 03:16 OSpset2EC2
drwxr-xr-x 2 nithi nithi 4096 Oct 19 14:44 Pictures
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Public
drwxr-xr-x 3 nithi nithi 4096 Sep 18 18:58 snap
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Templates
-rw-r--r-- 1 nithi nithi 4096 Sep 20 21:43 urandom_test
-rw-r--r-- 1 nithi nithi 4096 Sep 20 21:43 urandom_test2
drwxr-xr-x 2 nithi nithi 4096 Sep  9 20:20 Videos
Exit Status: 0
consuming 0.004 real seconds, 0.3373 user, 0.000 system
$
```

Source code of Shell Script used to test this feature:

```
#!/home/nithi/Documents/a.out
#This is an example of a shell script that your shell must execute correctly
#notice that lines starting with a # sign are ignored as comments!
#let’s say this here file is called testme.sh.  you created it with say
#vi testme.sh ; chmod +x testme.sh
#you invoked it with
#./testme.sh
cat >cat.out
#at this point, type some lines at the keyboard, then create an EOF (Ctrl-D)
#your shell invoked the system cat command with output redirected to cat.out
cat cat.out
#you better see the lines that you just typed!
exit 123
#after your shell script exits, type echo $? from the UNIX system shell
#the value should be 123.  Since your shell just exited, the following
#bogus command should never be seen
```



```
nithi@nythy:~/Documents$ gcc shell.c
nithi@nythy:~/Documents$ ./a.out
$
nithi@nythy:~/Documents$ gcc shell.c
nithi@nythy:~/Documents$ ./a.out testme.sh
This is a testing of shell script
ls -l
now i will hit crtl D
Exit Status: 0
consuming 136.053 real seconds, 0.3725 user, 0.000 system
This is a testing of shell script
ls -l
now i will hit crtl D
Exit Status: 0
consuming 0.003 real seconds, 0.2495 user, 0.000 system
nithi@nythy:~/Documents$ echo $?
123
nithi@nythy:~/Documents$
```