Part 1:

Text

Description automatically generated

Part 2:

I didn’t know which way to implement part 2, so I did both ways: first using simple-ls and tryShell, second using simple-ls and simple-cat.

Simple-lsShell.c Terminal Log:

Text

Description automatically generated

A picture containing text

Description automatically generated

**Simple-lsShell.c Listing:**

/\*

\* simple-ls.c

\* Extremely low-power ls clone.

\* ./simple-ls .

\*/

#include <sys/types.h>

#include <dirent.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <fcntl.h>

#define MAX 1024

**void** execute(**char** \*file) {

pid\_t pid;

**int** status;

**char** \*cmd = "cat";

**char** \*arg[3];

arg[0] = "cat";

arg[1] = file;

arg[2] = **NULL**;

**if**((pid = fork()) < 0) {

printf("%s", "error on forking");

exit(1);

}

**if** (pid == 0) {

**if**(execvp(cmd, arg) < 0) {

printf("%s", "error\n");

exit(1);

}

} **else** {

**while**(wait(&status) != pid);

}

}

**int** main(**int** argc, **char** \*\*argv) {

DIR \*dp;

**struct** dirent \*dirp;

**if** (argc != 2) {

fprintf(stderr, "usage: %s dir\_name\n", argv[0]);

exit(1);

}

// argv[1] == name of the directory to list

// opendir returns a pointer to a DIR structure

// dp stores that pointer

**if** ((dp = opendir(argv[1])) == **NULL**) {

fprintf(stderr, "can't open '%s'\n", argv[1]);

exit(1);

}

// passes dp into readdir

// loops to read each directory entry

// readdir returns a pointer to dirent structure

// dirp stores that pointer

**while** ((dirp = readdir(dp)) != **NULL** ){

printf("%s\n", dirp->d\_name);

**if**(strstr(dirp->d\_name, ".c") != **NULL**) {

// printf("\*\*\*\*\*\*\*\* START OF FILE: %s \*\*\*\*\*\*\*\*", dirp->d\_name);

execute(dirp->d\_name);

printf("\*\*\*\*\*\*\*\* END OF FILE: %s \*\*\*\*\*\*\*\*\n\n", dirp->d\_name);

}

}

closedir(dp);

**return**(0);

}

Still Part 2:

simple-lsCat.c Terminal Log:

Text

Description automatically generated

A picture containing graphical user interface

Description automatically generated

**simple-lsCat.c Listing:**

/\*

\* simple-ls.c

\* Extremely low-power ls clone.

\* ./simple-ls .

\*/

#include <sys/types.h>

#include <dirent.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <fcntl.h>

#define BUFFSIZE 32768

**void** catfile(**int** fd) {

**int** n;

**char** buf[BUFFSIZE];

**while** ((n = read(fd, buf, BUFSIZ)) > 0) {

**if** (write(STDOUT\_FILENO, buf, n) != n) {

fprintf(stderr, "write error\n");

exit(1);

}

}

}

**int** main(**int** argc, **char** \*\*argv) {

DIR \*dp;

**struct** dirent \*dirp;

**int** fd;

**if** (argc != 2) {

fprintf(stderr, "usage: %s dir\_name\n", argv[0]);

exit(1);

}

// argv[1] == name of the directory to list

// opendir returns a pointer to a DIR structure

// dp stores that pointer

**if** ((dp = opendir(argv[1])) == **NULL**) {

fprintf(stderr, "can't open '%s'\n", argv[1]);

exit(1);

}

// passes dp into readdir

// loops to read each directory entry

// readdir returns a pointer to dirent structure

// dirp stores that pointer

**while** ((dirp = readdir(dp)) != **NULL** ){

printf("%s\n", dirp->d\_name);

**if**(strstr(dirp->d\_name, ".c") != **NULL**) {

// printf("\*\*\*\*\*\*\*\* START OF FILE: %s \*\*\*\*\*\*\*\*", dirp->d\_name);

fd = open(dirp->d\_name, O\_RDONLY);

catfile(fd);

printf("\*\*\*\*\*\*\*\* END OF FILE: %s \*\*\*\*\*\*\*\*\n\n", dirp->d\_name);

}

}

closedir(dp);

**return**(0);

}