CSE 344 -System Programming Homework #1 Report

Ozan GEÇKİN
1801042103

Objective:

You are expected to write an “advanced” file search program for POSIX compatible operating systems. Your program must be able to search for files satisfying the given criteria, and print out the results in the form of a nicely formatted tree.

Design Explanation:

My project uses recursive search to find a file.The program takes arguments from the commend line.These arguments must be according to the rule in the homework file.The program should definitely get a targetPath to search the entered file information.I get this with -w. Also, I will search in the path should be recursively (i.e. across all of its subtrees).

The search criteria can be any combination of the following (at least one of them must be employed):
• -f : filename (case insensitive), supporting the following regular expression: +
• -b : file size (in bytes)
• -t : file type (d: directory, s: socket, b: block device, c: character device f: regular file, p:
pipe, l: symbolic link)
• -p : permissions, as 9 characters (e.g. ‘rwxr-xr--’)
• -l: number of links

Example:
```
./main -w / home/ ozan /Desktop  -f lost+file  -b 100  -t f  - p rw-rw-r-- -l 1
./main   w / home/ ozan /Desktop  -f lost+file
..
.
```
It can be operated with any combination as in the examples above.
I didn't use static array in my project, I used char pointer. Since I use a char pointer, I don't know the length like static array, so I used strlen and counter.I used dynamic arrays with Malloc to keep paths and free () function to avoid memory leakage. I kept my command line arguments in a struct. I check the inputs I get using string library functions.I'm doing a signal check to check CTRL-C.I used mainde struct signaction for this check.I explained the functions I use in the rest of my report.

The algorithm I designed:

Since the application needs to parse al directories and subdirectories I went for a recursive method that list all files and folders in the given folder and process each of them to see if it fits the entered parameters. It then calls itself against each folder found. Before validating each file it's stat is taken For the regex I using two indexes, one that goes thru the file name been validated and one that goes thru the regex. If at at any position the values are different it return false. If the regex index finds a + sign then the file name index gets moved forward until a character different than the one before +. If at the end both index managed to get to the end if it's text the method returns true otherwise false. For the permissions we used the fact that each permissions on the text has an specific position and created an array with the masks used to validate the permission at the same position. We use and index that checks that if the permission at x position on the array and the mask at the same position against stats.st_mode is the same. If the permission is set to '-' then we check if it wasn't set. stats.st_mode is also used to check the file type of the file by using the IF_IS(File type) macros and the parameter entered. File size and number of links are validated comparing the values entered on the parameters vs the value on stat.st_size and stat_nmlink respectively. The validation is done by checking if the parameter was set and if the file fits the value. If the value doesn't pass the method return false, otherwise it goes to the next one. At the end it returns true because that means that the file passed all the tests it needed to pass. A struct was made to store all the the parameters entered and facilitate passing them to each method. In order to print the paths I keep a variable with the previous path. It tokenizes the path using '/' as the delimiter and print each token into it's line with - repeated 2 times per level of depth. In order to create the tree structure the previous path is stored and tokenized and the current path is on printed from the first token that it's different while keeping the depth even if not printed.

Program’s Functions

RepeatCount(…) : While entering file name in the program, it can be entered as "lost + file". Here is the auxiliary function to the function I use to control the "+". In case of finding a '+' this function checks all repeats.
The parameter is fileName, starting index (where '+' is located), length of fileName before ‘+.Return type returns int count.
RegexFilename(…): Editing filename and checking with filenames.It returns a boolean value according to the status.It scans until the end of one of the entered filename and current filenames.If it finds '+' in it, the helper calls a function (RepeatCount). It takes the current fileName and the user-entered fileName as parameters.

CheckDirectory(...) : Traverse directories recursively and check each file or dir entry against the parameters. It calls these functions outside of itself within ValidateEnry, PrinthPath. It takes the struct, which holds the Command line arguments, and the target pathi as parameters.If CNTRL-C is done, it has cntrl-c control inside. I used malloc inside the function because create a variable to store the full path since dir->d_name only contains the current dir name.Return type is void.

PrintUsage(…) : When wrong input is entered on the command line, it prints the correct input information to the screen.

ValidateEntry(…) : Checks if each parameter has the default value and if not . Checks if the path entered fulfill the requirements. If comparisons are true,return true. I call the CheckType, RegexFilename, CheckPermissions functions in it. Takes the struct containing fullPath, filename, and arguments as parameters.

PrintPath(…) : This function prints the paths. It keeps track of the previously printed one and only prints the differences. It has to copy the strings because strtok_r alters the original string and we need them as is afterwards. It tokenize the last used path and the current one and soon as it finds a difference it puts the token back by keeping a varible for the previous token and setting the separator to its original value. strtok always replace the separator after the current token for a '\0'.

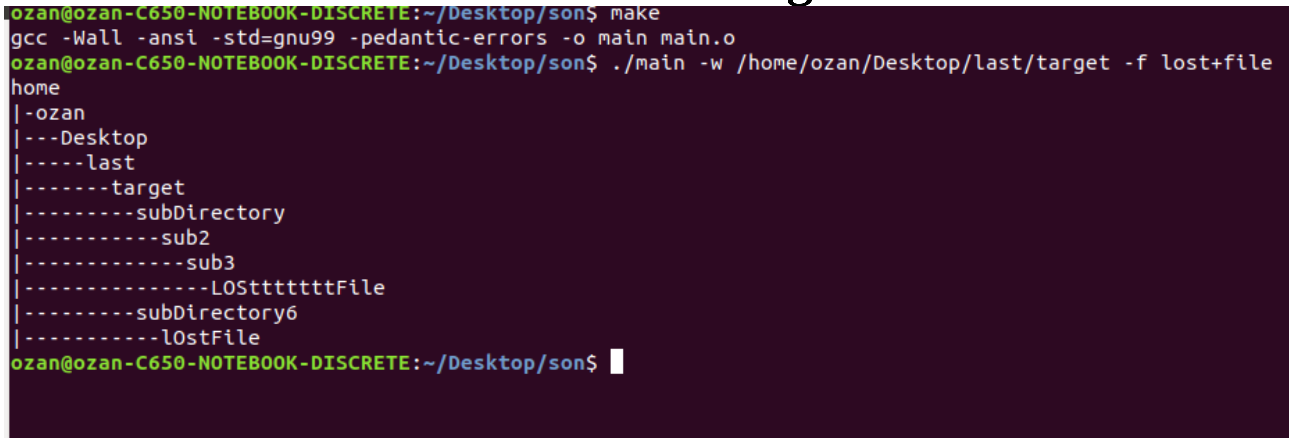
PrintSplitPath(…) : Print each toke on different line. Helps the PrintPath function.

CheckType(…): Validates the current file mode against the type entered on the parameters.
CheckPermissions(…): Check if the permission at each location fits against the one on the string. Return type boolean. Takes permissions as parameters.

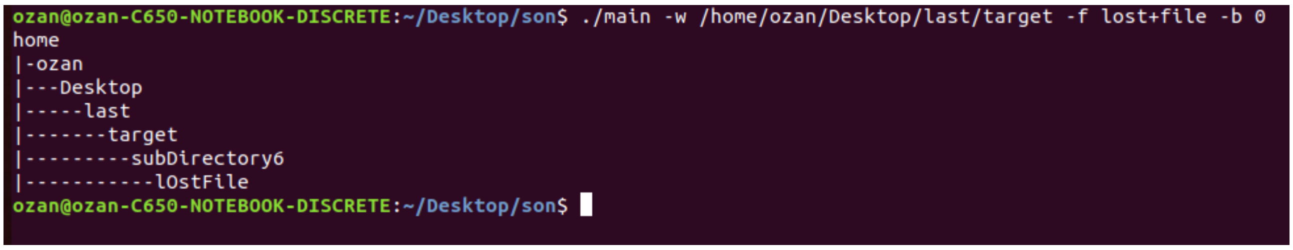
SignalHandler(…): Sets global var to true in order to know if ctrl was called.

Main(…) : I'm pars arguments with the getopt () function. And I'm checking.Setting up the signal handler to answer to CNTRL+C.

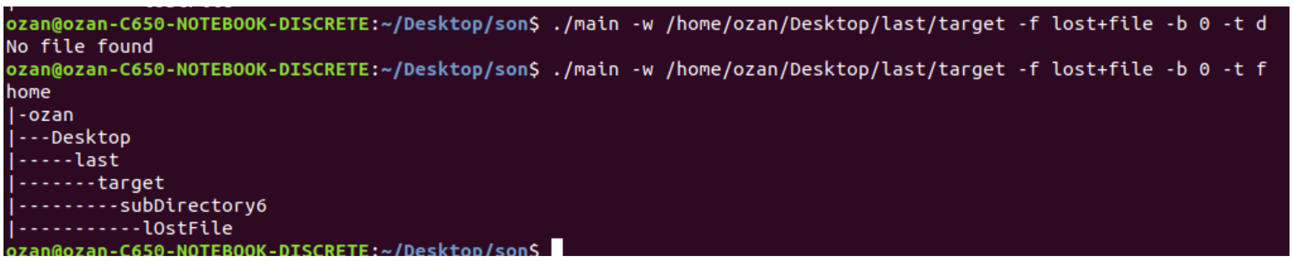
Some Screenshots that it is working:



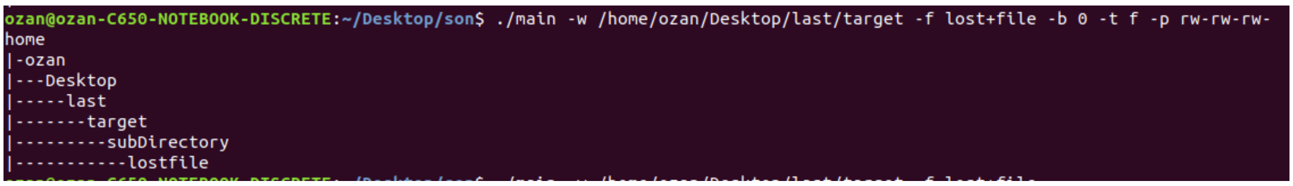
```
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ make
gcc -Wall -ansi -std=gnu99 -pedantic-errors -o main main.o
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop/last/target -f lost+file
home
|-ozan
|---Desktop
|-----last
|-------target
|---------subDirectory
|-----------sub2
|-------------sub3
|---------------LOStttttttFile
|---------subDirectory6
|-----------lOstFile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ 
```



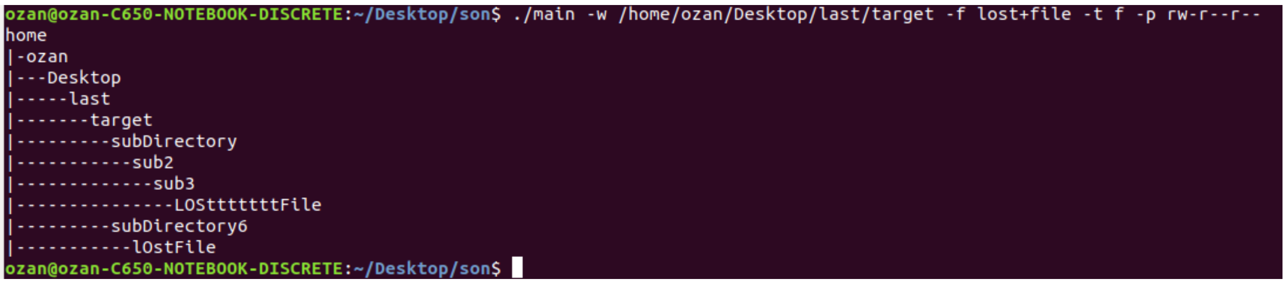
```
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop/last/target -f lost+file -b 0
home
|-ozan
|---Desktop
|-----last
|-------target
|---------subDirectory6
|-----------lOstFile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ 
```



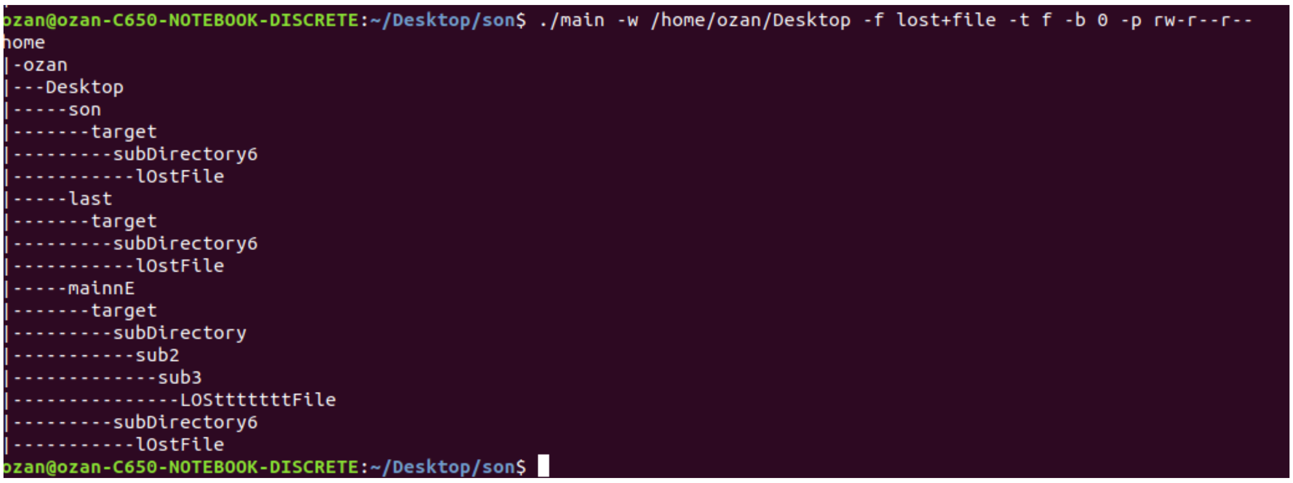
```
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop/last/target -f lost+file -b 0 -t d
No file found
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop/last/target -f lost+file -b 0 -t f
home
|-ozan
|---Desktop
|-----last
|-------target
|---------subDirectory6
|-----------lOstFile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ 
```



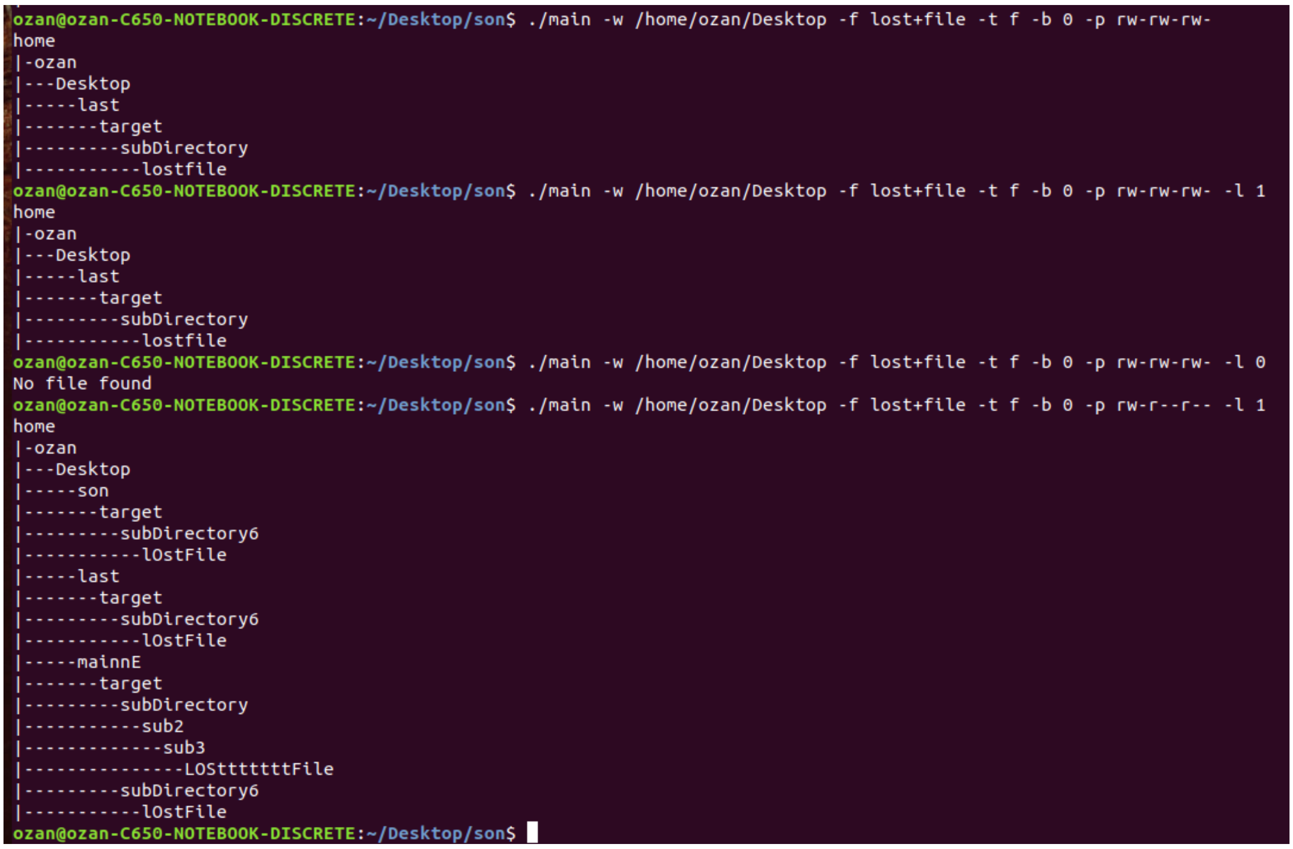
```
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop/last/target -f lost+file -b 0 -t f -p rw-rw-rw-
home
|-ozan
|---Desktop
|-----last
|-------target
|---------subDirectory
|-----------lostfile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop/last/target -f lost+file
```



```
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop/last/target -f lost+file -t f -p rw-r--r--
home
|-ozan
|---Desktop
|-----last
|-------target
|---------subDirectory
|-----------sub2
|-------------sub3
|---------------LOStttttttFile
|---------subDirectory6
|-----------lOstFile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$
```



```
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop -f lost+file -t f -b 0 -p rw-r--r--
home
|-ozan
|---Desktop
|-----son
|-------target
|---------subDirectory6
|-----------lOstFile
|-----last
|-------target
|---------subDirectory6
|-----------lOstFile
|-----mainnE
|-------target
|---------subDirectory
|-----------sub2
|-------------sub3
|---------------LOStttttttFile
|---------subDirectory6
|-----------lOstFile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$
```



```
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop -f lost+file -t f -b 0 -p rw-rw-rw-
home
|-ozan
|---Desktop
|-----last
|-------target
|---------subDirectory
|-----------lostfile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop -f lost+file -t f -b 0 -p rw-rw-rw- -l 1
home
|-ozan
|---Desktop
|-----last
|-------target
|---------subDirectory
|-----------lostfile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop -f lost+file -t f -b 0 -p rw-rw-rw- -l 0
No file found
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$ ./main -w /home/ozan/Desktop -f lost+file -t f -b 0 -p rw-r--r-- -l 1
home
|-ozan
|---Desktop
|-----son
|-------target
|---------subDirectory6
|-----------lOstFile
|-----last
|-------target
|---------subDirectory6
|-----------lOstFile
|-----mainnE
|-------target
|---------subDirectory
|-----------sub2
|-------------sub3
|---------------LOStttttttFile
|---------subDirectory6
|-----------lOstFile
ozan@ozan-C650-NOTEBOOK-DISCRETE:~/Desktop/son$
```