

MARCH 24, 2021

**GTU Department of Computer Engineering
CSE344 - Spring 2021
Homework 1 Report**

**Akif Kartal
171044098**

1 Problem Definition

The problem is to write an "advanced" file search program for POSIX compatible operating systems.

2 Solution

In order to write this program we need to **divide problem into modules**. These modules are following;

2.1 Argument Handling

In order to manage arguments in a proper way I used **argument struct** to keep given argument data as one block data. My argument struct is following.

```
1 typedef struct st
2 {
3     int wFlag;
4     int fFlag;
5     int bFlag;
6     int tFlag;
7     int pFlag;
8     int lFlag;
9     char *wArg;
10    char *fArg;
11    char *bArg;
12    char *tArg;
13    char *pArg;
14    char *lArg;
15    int isFound;
16    int count;
17
18 } args;
```

In this struct **count** represent number of optional arguments and **isFound** represent file is found or not. Also, in order to get argument's value **getopt()** library method was used.

2.2 Error Handling

In order to handle any error **stderr** used with **write system call** and **exit** system call was used. Following code shows an example of this;

```
1 void my_fprintf_with_stderr(const char *str){
2     ssize_t size = strlen(str);
3     if (size != write(STDERR_FILENO, str, size)) {
4         perror("write system call error!");
5         exit(EXIT_FAILURE);
6     }
7
8 }
```

Usage:

```
1 if (stat(path, &fileStat) == -1)
2 {
3     char *str = "Stat system call error!!\n";
4     my_fprintf_with_stderr(str);
5     exit(EXIT_FAILURE);
6 }
```

2.3 Advanced Search Algorithm

2.3.1 Regex Handling

The filename argument can contain more than one "+" character. To handle this situation in an easy way somehow we need to keep position data of these regexs. In order to do this I created my own **LinkedList** data structure to keep both position and previous letter information. My **LinkedList** node is following;

```
1 //regex information node
2 typedef struct node_s
3 {
4     int position;
5     char preChr;
6     struct node_s *next;
7 }
8 node_t;
```

2.3.2 Checking Given Parameters

- File name check was made by using regex linkedlist.
- File size, file type, file permissions and file links was checked by using **stat** calls.

2.3.3 Recursive Search Algorithm

TBA.

2.3.4 Drawing nicely formatted tree

If the searching file is found then directory tree will be drawn by using same recursive search algorithm with just a **minor** difference.

2.4 CTRL-C Handling

In order to give a message on CTRL-C interrupt, I used **sigaction** function from **signal.h** library.