LABORATORY 2
–
OPERATING SYSTEMS

RUFINO GARCIA SANCHEZ

Exercise 2

Task 1.

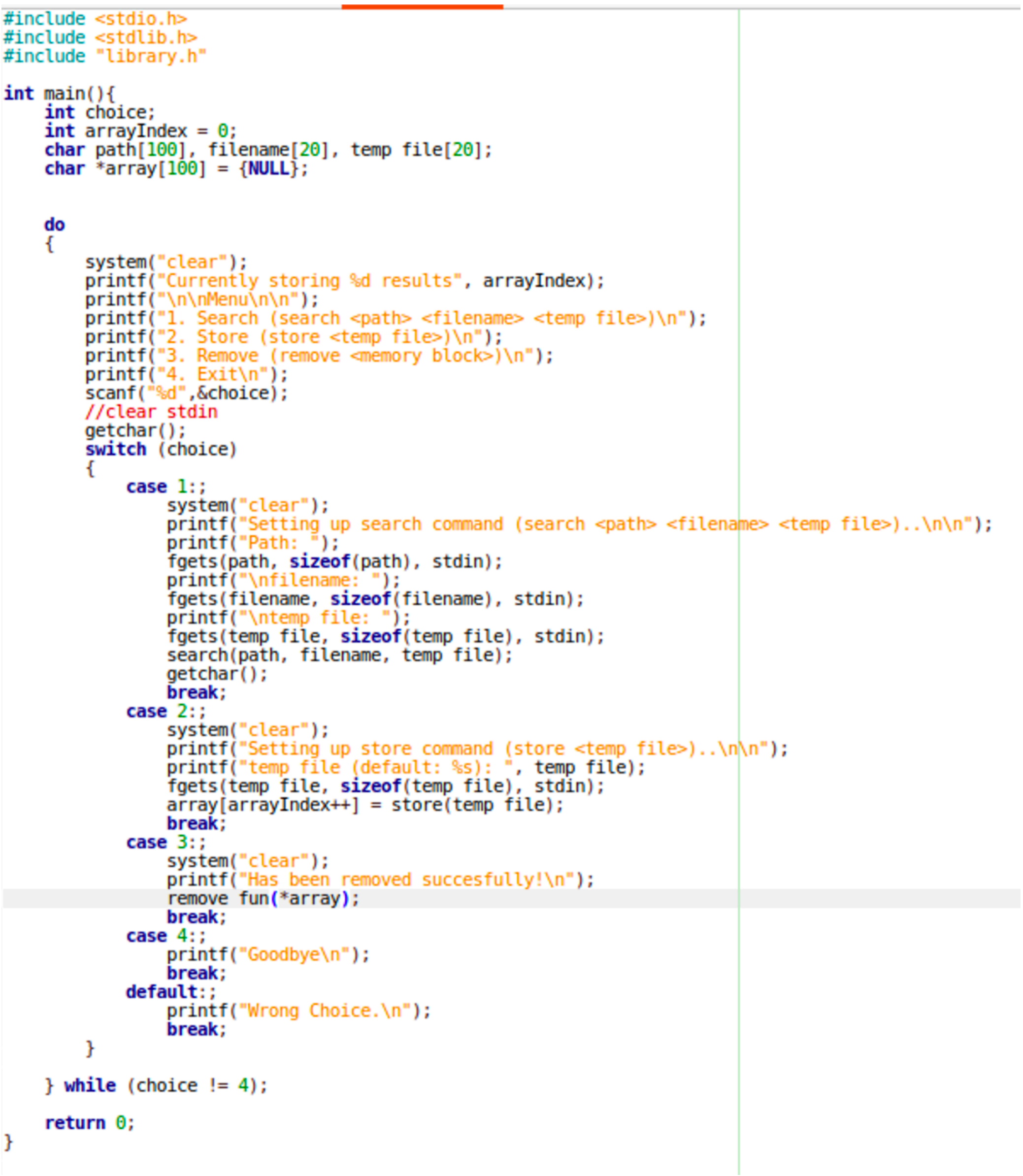
Write a library responsible for handling an *array* array with pointers to blocks containing the results of the execution of the find command. Following instructions are possible:

- **execute** search of the files with the name *filenames* starting from *root* directory, **store the output and diagnostic output in *temp* file.**

•**store** the content of the *temp* file in dynamicaly allocated block of memory, **add the pointer to this block in the *array* – *store temp***

•**remove (dealocate)** the block of memory accesed in the entry number in ***array* – *remove number*.**

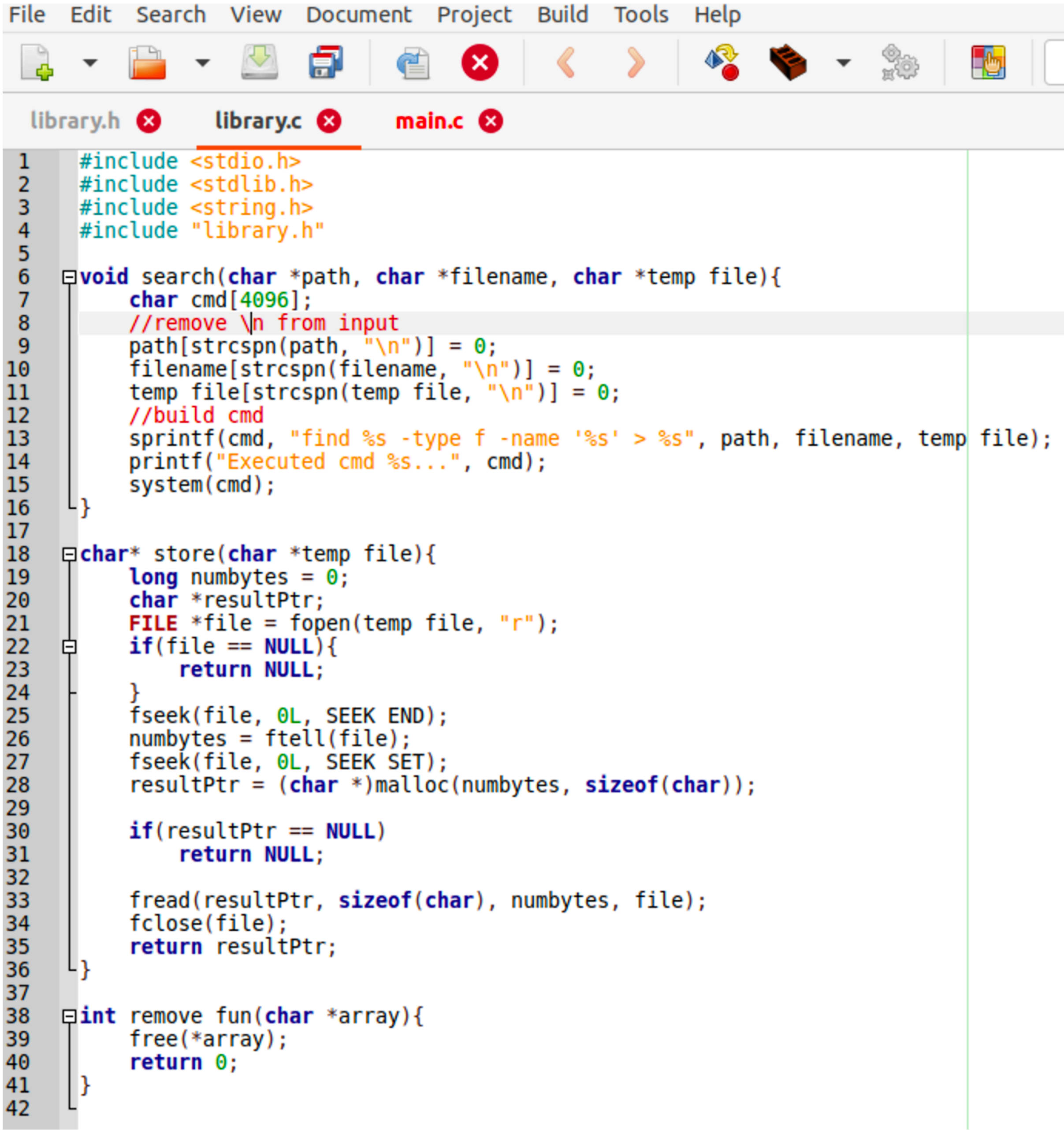
To this program I have create a menu where the user can choose this different option to do.



```
#include <stdio.h>
#include <stdlib.h>
#include "library.h"

int main(){
    int choice;
    int arrayIndex = 0;
    char path[100], filename[20], temp file[20];
    char *array[100] = {NULL};


    do
    {
        system("clear");
        printf("Currently storing %d results", arrayIndex);
        printf("\n\nMenu\n\n");
        printf("1. Search (search <path> <filename> <temp file>)\n");
        printf("2. Store (store <temp file>)\n");
        printf("3. Remove (remove <memory block>)\n");
        printf("4. Exit\n");
        scanf("%d",&choice);
        //clear stdin
        getchar();
        switch (choice)
        {
            case 1:;
                system("clear");
                printf("Setting up search command (search <path> <filename> <temp file>)..\n\n");
                printf("Path: ");
                fgets(path, sizeof(path), stdin);
                printf("\nfilename: ");
                fgets(filename, sizeof(filename), stdin);
                printf("\ntemp file: ");
                fgets(temp file, sizeof(temp file), stdin);
                search(path, filename, temp file);
                getchar();
                break;
            case 2:;
                system("clear");
                printf("Setting up store command (store <temp file>)..\n\n");
                printf("temp file (default: %s): ", temp file);
                fgets(temp file, sizeof(temp file), stdin);
                array[arrayIndex++] = store(temp file);
                break;
            case 3:;
                system("clear");
                printf("Has been removed succesfully!\n");
                remove fun(*array);
                break;
            case 4:;
                printf("Goodbye\n");
                break;
            default:;
                printf("Wrong Choice.\n");
                break;
        }

    } while (choice != 4);

    return 0;
}
```

As we can observe I have made a case that when the user introduces one number between 1..4 the program will execute different options.

Now we are going to see the code of the library. In this one I have made **3 functions**, one for find, one for allocate in memory the content of the file in a pointer of char, and another one to deallocate the array with the pointer to the content of temp.



```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "library.h"

void search(char *path, char *filename, char *temp file){
    char cmd[4096];
    //remove \n from input
    path[strcspn(path, "\n")] = 0;
    filename[strcspn(filename, "\n")] = 0;
    temp file[strcspn(temp file, "\n")] = 0;
    //build cmd
    sprintf(cmd, "find %s -type f -name '%s' > %s", path, filename, temp file);
    printf("Executed cmd %s...", cmd);
    system(cmd);
}

char* store(char *temp file){
    long numbytes = 0;
    char *resultPtr;
    FILE *file = fopen(temp file, "r");
    if(file == NULL){
        return NULL;
    }
    fseek(file, 0L, SEEK END);
    numbytes = ftell(file);
    fseek(file, 0L, SEEK SET);
    resultPtr = (char *)malloc(numbytes, sizeof(char));

    if(resultPtr == NULL)
        return NULL;

    fread(resultPtr, sizeof(char), numbytes, file);
    fclose(file);
    return resultPtr;
}

int remove fun(char *array){
    free(*array);
    return 0;
}
```

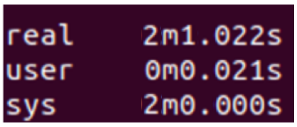
Task 2

Write a program which uses library from (**Task 1**) to store data obtained by several execution of the *search*, *store*, and *remove* commands... Calculate times using *times ()* function of each operation and store it in the *report.txt* file.

Provide comments to the obtained results of tests.

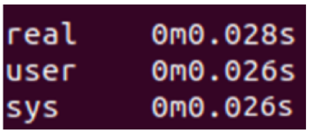
The **times ()** using the different functions of the library are:

For the **search function** -->



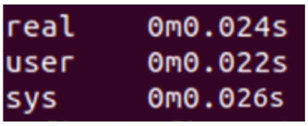
```
real	2m1.022s
user	0m0.021s
sys	2m0.000s
```

For the **store function** -->



```
real	0m0.028s
user	0m0.026s
sys	0m0.026s
```

For the **remove function** -->



```
real	0m0.024s
user	0m0.022s
sys	0m0.026s
```

Task 3

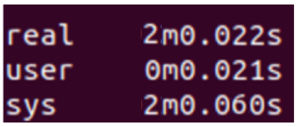
Build versions of program from Task 2 using different kind of libraries – ***static***, ***shared*** **and** ***dynamic***.

Now we are going to use dynamic and shared libraries, first we have generated and now we are going to watch the times () between these libraries, and we are going to compare it.

With the **static library** is in the task 2 so we proceed to **show with the shared and dynamic.**

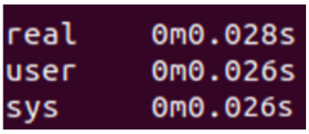
Shared library:

For the **search function** -->



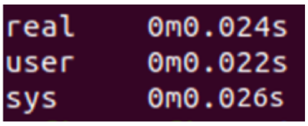
```
real	2m0.022s
user	0m0.021s
sys	2m0.060s
```

For the **store function** -->



```
real	0m0.028s
user	0m0.026s
sys	0m0.026s
```

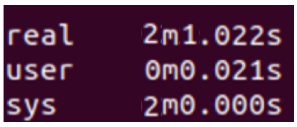
For the **remove function** -->



```
real    0m0.024s
user    0m0.022s
sys     0m0.026s
```

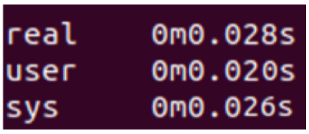
Dynamic library:

For the **search function** -->



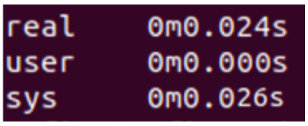
```
real    2m1.022s
user    0m0.021s
sys     2m0.000s
```

For the **store function** -->



```
real    0m0.028s
user    0m0.020s
sys     0m0.026s
```

For the **remove function** -->



```
real    0m0.024s
user    0m0.000s
sys     0m0.026s
```