



T.C.

MARMARA UNIVERSITY

FACULTY of ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

CSE3033 OPERATING SYSTEMS

TERM PROJECT #2

OSMAN MANTICI - 150117505

Part A

We define char array to determine alternative operations . Then by “PathList” method we created an array for path environments. Rest of the code will run in a while loop.

After printing shell name on terminal screen we use flush method to clear file pointer’s file buffer. The next step is forking a child process for input command and if forked the input arguments going to be controlled for execution.

The arguments will be controlled for input/output redirections and background process cases. After coping with child process we are looking for answer from parent process or background process and informing about background process and then we completed main function.As you can see the below I apply ls, pwd and ls &.

```
osmantici@OsmanMantici: ~/Masaüstü/Opsys2
Dosya Düzenle Görünüm Ara Uçbirim Yardım
osmantici@OsmanMantici:~/Masaüstü/Opsys2$ gcc abc.c -o abc.out
abc.c: In function 'main':
abc.c:718:35: warning: assignment to 'char' from 'void *' makes integer from pointer without a cast [-Wint-conversion]
    718 |         tempString[l] = NULL;
        |                        ^
abc.c:727:43: warning: assignment to 'char' from 'void *' makes integer from pointer without a cast [-Wint-conversion]
    727 |         tempString[l] = NULL;
        |                        ^
abc.c:732:38: warning: comparison between pointer and integer
    732 |         else if(temp2[k] == NULL){
        |                        ^~
osmantici@OsmanMantici:~/Masaüstü/Opsys2$ ./abc.out

MyShell: ls
abc.c          LinuxCommands-ShellProgramming.pdf 'Redirection & Pipes.pdf'
abc.out        processes          'redirect&pipes'
'CSE3033 - Lab - Week 7 - Section 1.mp4' Processes.pdf      threads
exec           Project2

MyShell: pwd
/home/osmantici/Masaüstü/Opsys2

MyShell: ls &

Process created with PID: 2409
Parent id is = 2367

MyShell: abc.c          LinuxCommands-ShellProgramming.pdf 'Redirection & Pipes.pdf'
abc.out        processes          'redirect&pipes'
'CSE3033 - Lab - Week 7 - Section 1.mp4' Processes.pdf      threads
exec           Project2
```

Part B

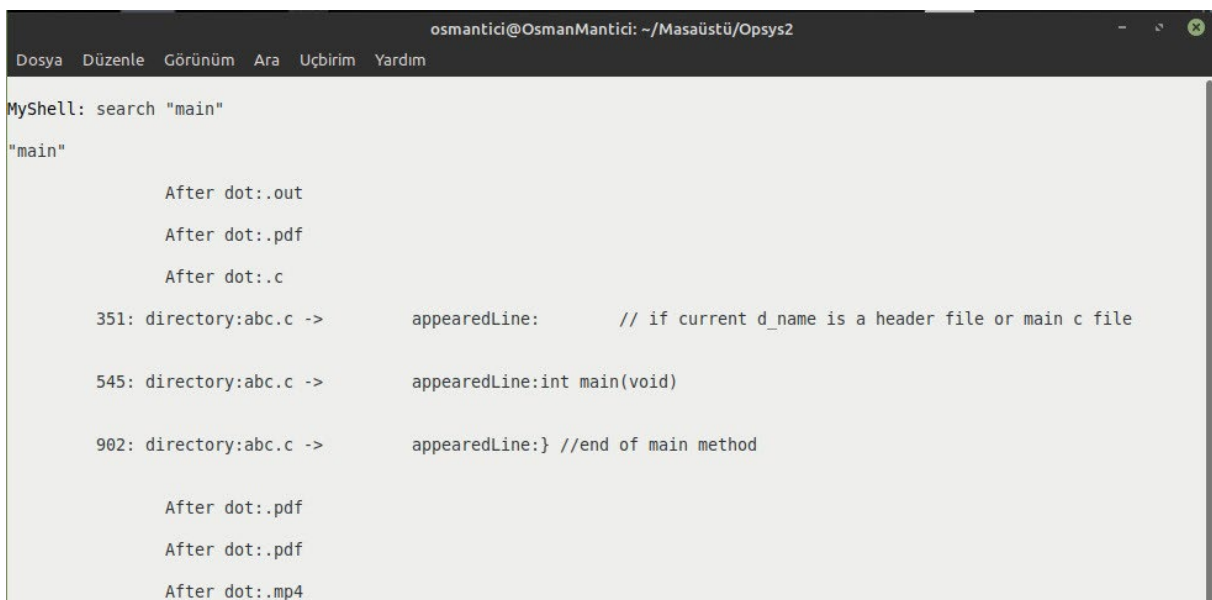
Search functionality. While our terminal running, if the entered command is search, main function calls “doSearch()” function with an input entered argument line.

So in “doSearch()” function, we use second argument line variable (args[1]), because first array cell holds search command and second cell holds keyword which will be searched on current directory. But this keyword holds in this array cell with quotation marks like “keyword”. In order to get rid of these two quotation marks there is a for loop which skips the mark and copy keyword to another string variable. we will use after that these newly created and filled temp2 character array.

In order to search this keyword in files were placed in current directory, we create again “dirSearch” pointer for directory entries and opening that directory create a directory pointer

“dirPtr” but this time in addition those, we create another pointer named “filePtr” with “FILE” type. This filePtr allows us to read file contents.

At this point, because we are only looking for specific file extensions (.c, .C, .h, .H), we store file extensions in a character array by using “strchr()” function which returns first occurrence point of searched specific character. So in “afterDot” variable we keep first location of the file extension with a dot. For example, current file named like “currentFile.doc”, our afterDot variable holds beginning memory place of “.doc”. So if current file is a header file or is an executable c file, we open that file for just reading. Furthermore, we start to read line-by-line in to “buffer” variable with “fgets()” function. At this point, we inspect the buffer variable with “strstr()” function, it is very similar to strchr() but only difference, strstr() searches string in a string and returns first memory place of searched string. In order to keep what line it is stayed, we increase our “lineNumber” variable in the for loop. If there is an occurrence, we print on screen immediately.



```
osmantici@OsmanMantici: ~/Masaüstü/Opsys2
Dosya  Düzenle  Görünüm  Ara  Uçbirim  Yardım

MyShell: search "main"
"main"

    After dot:.out
    After dot:.pdf
    After dot:.c

351: directory:abc.c ->      appearedLine:      // if current d_name is a header file or main c file

545: directory:abc.c ->      appearedLine:int main(void)

902: directory:abc.c ->      appearedLine:} //end of main method

    After dot:.pdf
    After dot:.pdf
    After dot:.mp4
```

In bookmark functionality, we use a linked list structure in order to dynamically add, delete bookmarks.

We use “display()” function for “bookmark -l” option.

“insert_end()” for adding a new bookmark e.g. (bookmark “ps -a”).

“indexDisplay()” for executing a specific bookmark e.g. (bookmark -i 0).

“delete_pos()” for deleting a specific bookmark e.g. (bookmark -d 0), in addition this functionality we use “reconstruct()” function to update indexes of stored bookmarks.

After user entered bookmark keyword, we identify rest of the command after bookmark. The rest of command;

- If it starts with (“”) mark, we directly use just “insert_end()” function in order to add entered bookmark at the end of the bookmark linked list.

-- "insert_end()" function takes three input parameter and these are argument line (char *args[]), index value (int location), and argument line count (int argc). The argument count value argc is obtained from setup() function, so we change a little bit setup() function and it returns an integer value as argc. There is an important point now. Let's say given input like bookmark "ls -l | wc -l", in order to not loose these spaces between this command, we first copy second argument (args[1]) with using "strcpy()", after that concatenate a space tail of string and we continue to adding all argument line elements end of this string. So added bookmark content is stored now "content" value of relevant bookmark linked list structure and "info" value holds the index of the bookmark.

- If it starts with "-i" we take third argument line item into index variable and then we use that index for find the bookmark that placed on index variable. This finding bookmark content is done by the function "indexDisplay(index)" and this function returns the desired command to run. After finding content, we have to construct this content in as an input createChild() function, because createChild() will be execute this bookmark content. So then, returned string contains (") marks, we remove quotation marks from that string but actually removed version of that string is stored in another variable named "temp2". We copy the content of that temp2 string variable into argument line variable args[0], so while doing this copy operation we actually start constructing the input of crateChild(). At this point there is an integer value named "m" and it holds the number of spaces inside of bookmark content, but actually we want to find number of words in bookmark content. For example, if bookmark content like "ls -l | wc -l" there is 4 spaces and number of words equal to one incremented value of spaces so 5 words. We want this word count in order to initialize argument line with NULLs. After that, we separates temp2 variable space-by-space and each separated string "tempString", then this tempString assigned to the relevant place of the argument line(args[]).

--"indexDisplay()" function takes only "index" variable as input parameter. This function traverses the linked list and while traversing compares the entered "index" value and "info" value of list element and it stops if there is a match with index and info values. After that returns the string content of the bookmark.

- If it starts with "-d" we take third argument line item into index variable. After getting responsible index, we use it with the function parameter like "delete_pos(index)". Delete_pos() deletes given bookmark placed on the index. But we may consider two important point. First point is, after deleting an item from linked list we must reconstruct the index values of these linked list items. We do that work with function named "reConstruct()". For example, let's say if we have three item like 0, 1, 2 and then we delete index 1, linked list seems like 0,2. So the index of second item must be replaced with 1. The other important point is, decrementing of location variable. it is necessary for next element will be added to bookmark. So let's say after deleting an item from three element contained linked list we have two elements like indexes "0,1" and if we not decrement location variable newly added item will be had an index value with "3" instead "2".

--"delete_pos()" function means delete entered positon from bookmark list. The function takes only one input as parameter and this input is "index" value of entered argument line. The function goes to given index if it is existing and deletes that list element from bookmark list, but if it is not return without doing any deletion.

--"reConstruct()" function is not take any input, and it updates "info" values of bookmark linked list elements starting zero to end of linked list and incrementing one-by-one that "info" values.

- If it starts with "-l", we use just use "display()" function in order to show all elements of bookmark linked list.

--"display()" function is not take any input and start from beginning to end of linked list and just puts the information about relevant linked list element one-by-one.

--,"exit": This command will operate and terminate the myShell when the use enter the **exit** in the CMD.

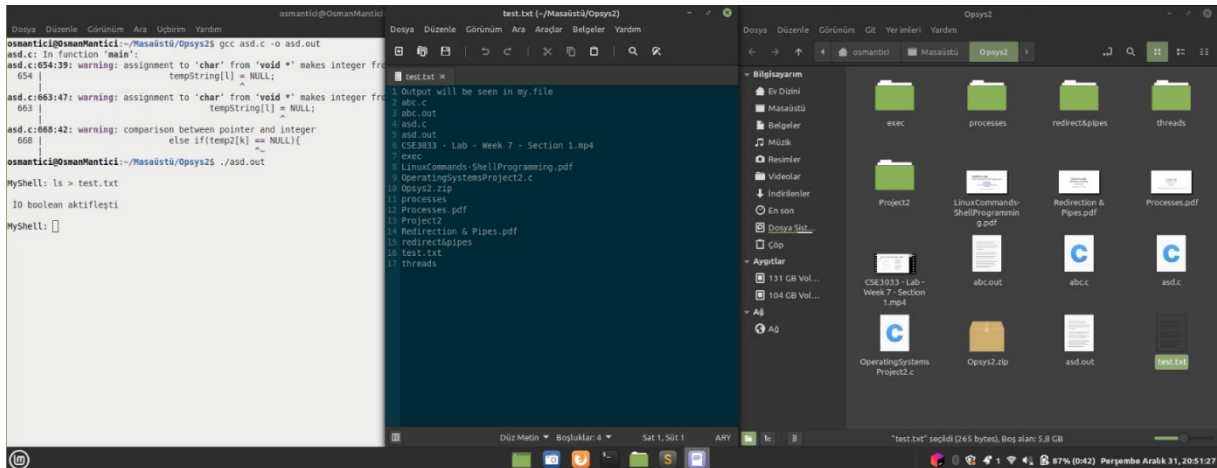
```
osmantici@OsmanMantici: ~/Masaüstü/Opsys2
Dosya Düzenle Görünüm Ara Uçbirim Yardım
osmantici@OsmanMantici:~/Masaüstü/Opsys2$ gcc abc.c -o abc.out
abc.c: In function 'main':
abc.c:715:35: warning: assignment to 'char' from 'void *' makes integer from pointer without a cast [-Wint-conversion]
715 |         tempString[l] = NULL;
    |         ^
abc.c:724:43: warning: assignment to 'char' from 'void *' makes integer from pointer without a cast [-Wint-conversion]
724 |         tempString[l] = NULL;
    |         ^
abc.c:729:38: warning: comparison between pointer and integer
729 |         else if(temp2[k] == NULL){
    |                        ^~
osmantici@OsmanMantici:~/Masaüstü/Opsys2$ ./abc.out
MyShell: bookmark "ps -a"
MyShell: bookmark "ls -l | wc -l"
MyShell: bookmark -l
location: 0 content:"ps -a"
location: 1 content:"ls -l | wc -l"
MyShell: bookmark -i 0
PID TTY TIME CMD
3187 pts/0 00:00:00 abc.out
3189 pts/0 00:00:00 abc.out
3192 pts/0 00:00:00 abc.out
3194 pts/0 00:00:00 abc.out
3195 pts/0 00:00:00 ps
MyShell: bookmark -d 0
location: 1 content:"ls -l | wc -l"
location: 0 content:"ls -l | wc -l" The deleted element is:0 its content is:"ps -a"
MyShell: bookmark -l
location: 0 content:"ls -l | wc -l"
MyShell: exit
Logging out...
Terminated
osmantici@OsmanMantici:~/Masaüstü/Opsys2$ gcc abc.c -o abc.out
abc.c: In function 'main':
abc.c:715:35: warning: assignment to 'char' from 'void *' makes integer from pointer without a cast [-Wint-conversion]
715 |         tempString[l] = NULL;
    |         ^
abc.c:724:43: warning: assignment to 'char' from 'void *' makes integer from pointer without a cast [-Wint-conversion]
724 |         tempString[l] = NULL;
    |         ^
abc.c:729:38: warning: comparison between pointer and integer
729 |         else if(temp2[k] == NULL){
    |                        ^~
osmantici@OsmanMantici:~/Masaüstü/Opsys2$ ./abc.out
MyShell: ^Z
[1]+ Stopped ./abc.out
osmantici@OsmanMantici:~/Masaüstü/Opsys2$
```

Part C

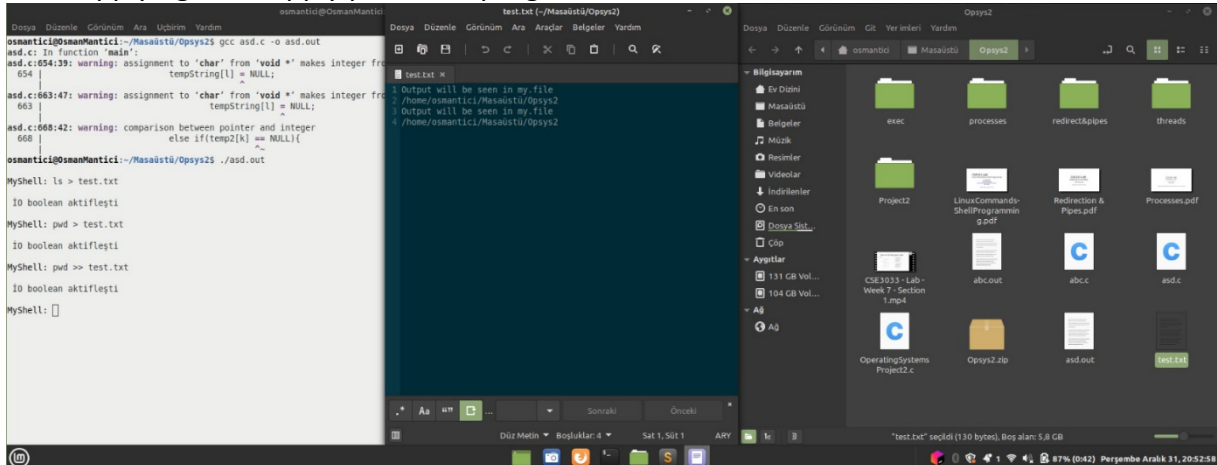
The shell supports I/O-redirection on either or both *stdin* and/or *stdout* and it can include arguments as well. We create boolean for checking is there input output operation. After detect we control the check the sign for applying right method.

File operation 1

As you can see in the below first file operation writes the standard output of **myprog** to the file **file.out**. **file.out** is created if it does not exist and truncated if it does. We write the command **ls** and it create **test.txt** and write inside it.



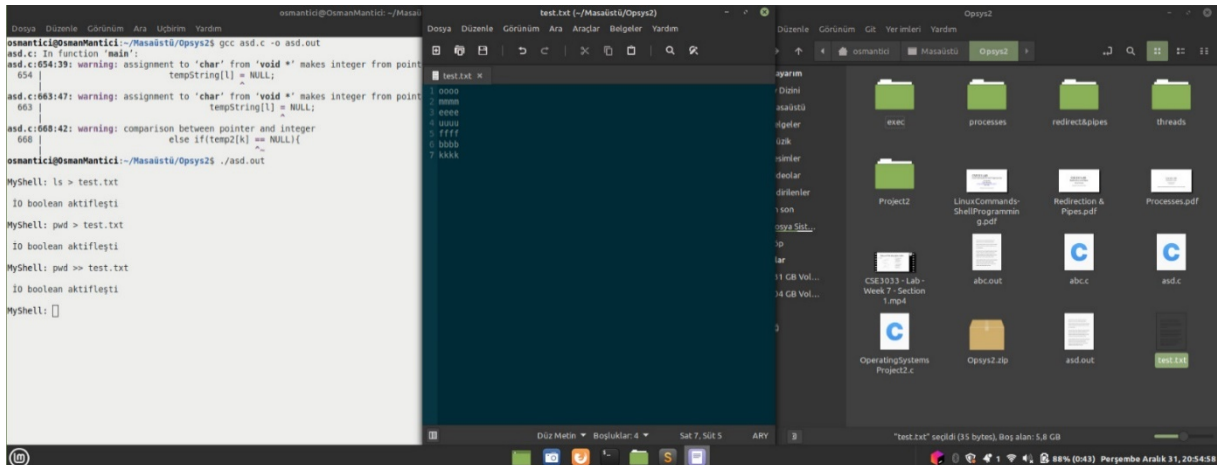
After applying **ls** we apply **pwd** then program write on the file after delete **ls** conclusions.



File operation 2

In this operation, appends the standard output of **myprog** to the file **file.out**. **file.out** is created if it does not exist and appended to if it does so as you can see in the upside we write **pwd** a program writes to file but it does not delete it write on it.

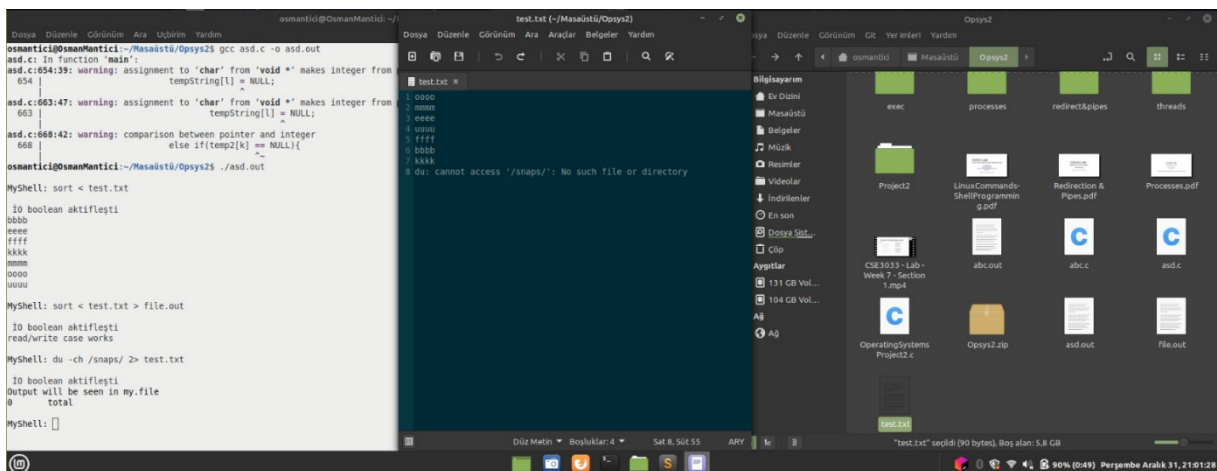
File operation 3



In this operation uses the contents of the file **file.in** as the standard input to program myprog so we apply sort command after creating test.txt program sort and print to console as we expect it.

File operation 4

In this operation Writes the standard error of **myprog** to the file **file.out**. So we use special command `du -ch /snaps/ 2> test.txt` to get error message then as we expect we get error message.

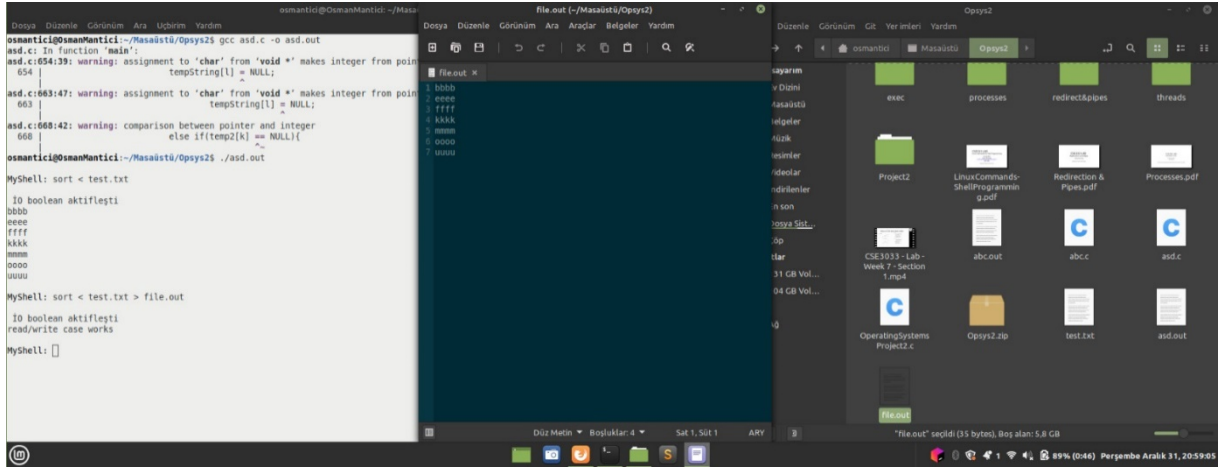


Program write to file.out but it does not delete the content of it.

File operation 5

Executes the command **myprog** which will read input from **file.in** and stdout of

the command is directed to the file **file.out** the only difference between file operation 3 and file operation 5 is in this operation we do not write to console we write to the output.txt as you can see below.



The screenshot shows a terminal window with the following content:

```
osman@osmanMantici: ~/MasaiStu$ gcc asd.c -o asd.out
asd.c: In function 'main':
asd.c:654:39: warning: assignment to 'char' from 'void*' makes integer from pointer without a cast
654 |         tempString[l] = NULL;
      |         ^
asd.c:663:47: warning: assignment to 'char' from 'void*' makes integer from pointer without a cast
663 |         tempString[l] = NULL;
      |         ^
asd.c:668:42: warning: comparison between pointer and integer
668 |         else if(temp2[k] == NULL){
      |              ^
osman@osmanMantici:~/MasaiStu$ ./asd.out
MyShell: sort < test.txt
10 boolean aktifleşt
bbbb
nece
ffff
kkkk
oooo
uuuu
MyShell: sort < test.txt > file.out
10 boolean aktifleşt
read/write case works
MyShell: 
```

The terminal output shows the execution of a C program that sorts the contents of 'test.txt' and writes the result to 'file.out'. The output of the program is displayed in the terminal window, showing the sorted contents of 'test.txt' and the status of the read/write operation.