Task 2: I/O Redirection

Objectives Completed:

- Executed command line arguments separately.
- Redirected the output of one process to another.
- Compared the output of two processes.

Pseudocode:

- Take input through command line arguments.
- Convert array command line arguments to an array of characters (single string).
- Split the string by ‘ p ’ or ‘ c ‘ and get two separate strings.

Part 1:

If the strings were separated by ‘p’, then do the following:

- Create two separate processes and run first string as a shell command while second process waits.
- When first process is completed take its output and send it to the second process as an input and the second process execute the second string as a shell command.

Part 2:

If the strings were separated by ‘c’, then do the following:

- Create two separate processes and execute both strings as shell commands.
- Send the output of both processes to the parent process.
- Compare the output and print True or False.

List of methods in the program:

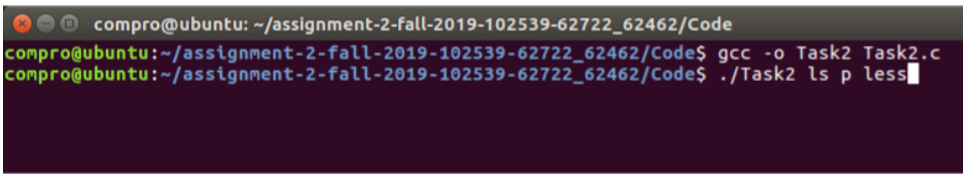
1. **char * parseCmdLineArgs(char * argv[]):** This method is used to convert the command line arguments to a an array of characters and adds spaces between them because the command line arguments are saved in a pointer of character arrays like a 2d array of characters.
2. **void split(char cmd[]):** This method is used to split a string into two separate strings. If space p space or space c space is found a global variable is set to 0 or 1 and the character p or c is replaced by ‘|’. Then the **strtok()** library function is

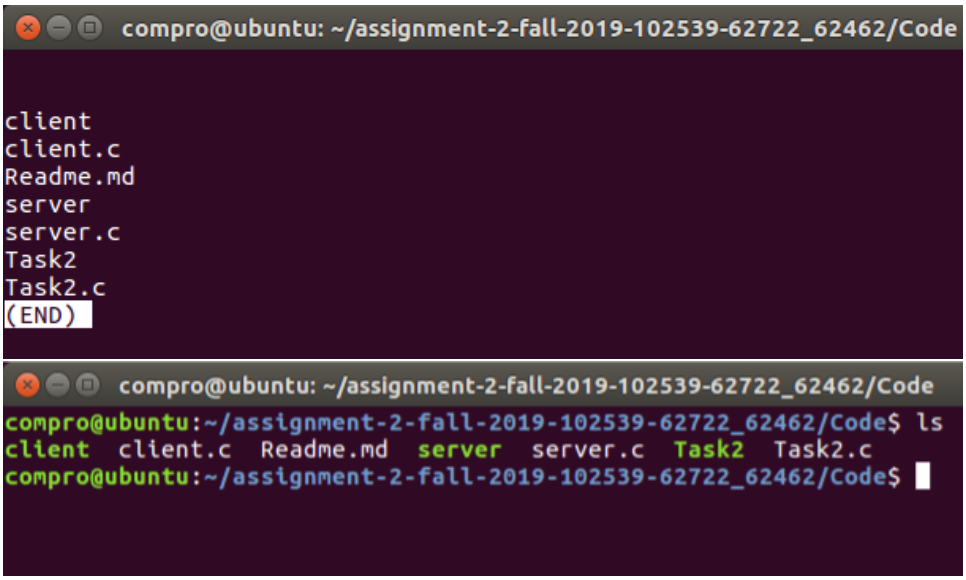
used to split the string into two strings and the strings are saved in global variables.

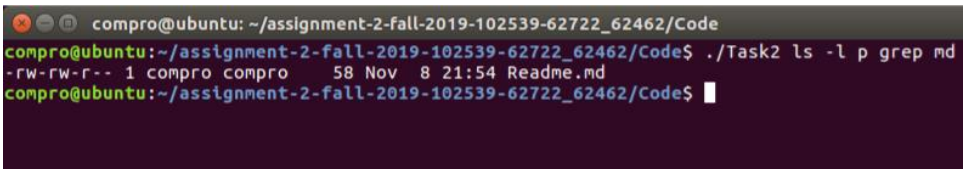
3. **int main():** The main method first checks if there are enough command line arguments to perform I/O redirection or comparison, so If **int argc** is less than 4 the program will exit with status 1.
   Then **char * parseCmdLineArgs(char * argv[])** method runs and the output is saved in an array and that array is used as the argument for **void split(char cmd[])** after that we create two child processes and the parent is main, both child process have a condition which will do only one of the following things: (a) redirect the output of first program to other child to be used as an input for the second program. (b) both childs execute their programs and redirect their output to the parent and the parent process save their output in an array and compares them. The conditions is decided by a global variable which was set by **split()** method.

Sample Execution:

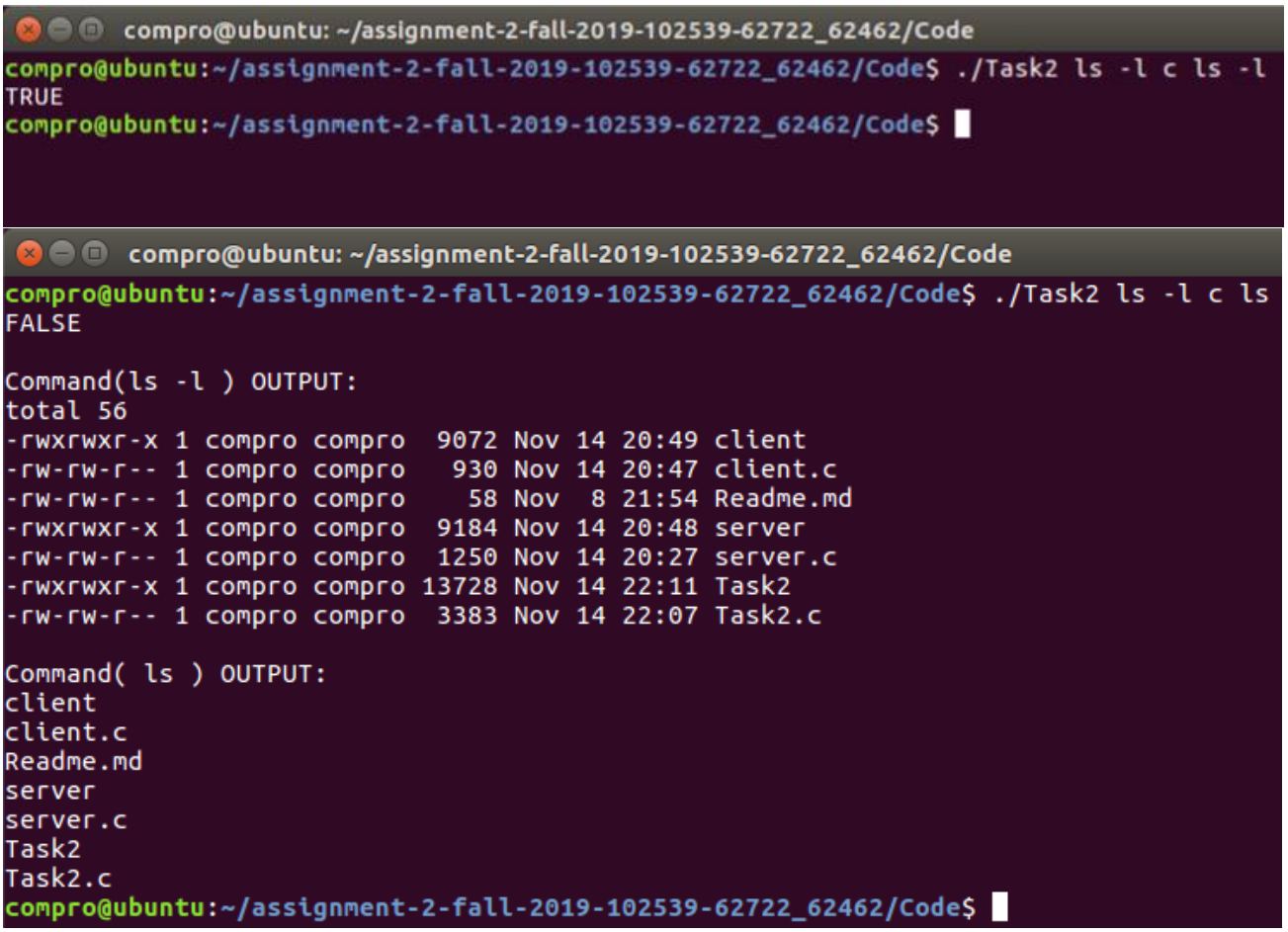
I/O Redirection:






compro@ubuntu: ~/assignment-2-fall-2019-102539-62722_62462/Code
compro@ubuntu:~/assignment-2-fall-2019-102539-62722_62462/Code$ ./Task2 ls -l p grep md
-rw-rw-r-- 1 compro compro 58 Nov 8 21:54 Readme.md
compro@ubuntu:~/assignment-2-fall-2019-102539-62722_62462/Code$

Comparison:


compro@ubuntu: ~/assignment-2-fall-2019-102539-62722_62462/Code
compro@ubuntu:~/assignment-2-fall-2019-102539-62722_62462/Code$ ./Task2 ls -l c ls -l
TRUE
compro@ubuntu:~/assignment-2-fall-2019-102539-62722_62462/Code$
compro@ubuntu: ~/assignment-2-fall-2019-102539-62722_62462/Code
compro@ubuntu:~/assignment-2-fall-2019-102539-62722_62462/Code$ ./Task2 ls -l c ls
FALSE
Command(ls -l) OUTPUT:
total 56
-rwxrwxr-x 1 compro compro 9072 Nov 14 20:49 client
-rw-rw-r-- 1 compro compro 930 Nov 14 20:47 client.c
-rw-rw-r-- 1 compro compro 58 Nov 8 21:54 Readme.md
-rwxrwxr-x 1 compro compro 9184 Nov 14 20:48 server
-rw-rw-r-- 1 compro compro 1250 Nov 14 20:27 server.c
-rwxrwxr-x 1 compro compro 13728 Nov 14 22:11 Task2
-rw-rw-r-- 1 compro compro 3383 Nov 14 22:07 Task2.c
Command(ls) OUTPUT:
client
client.c
Readme.md
server
server.c
Task2
Task2.c
compro@ubuntu:~/assignment-2-fall-2019-102539-62722_62462/Code$