# CSE 3033 - OPERATING SYSTEMS
# Report of Programming Assignment # 2

**150117011 – 150116049**

**Buse Batman – Göksel Tokur**

This programming assignment is about writing a simple shell. And the assignment consists of these methods:

*At first we created two structs. First one is executed commands linked list, the other one is for background queue.

*Setup function reads the command line from the terminal and calculates the length. And checks for every character whether is it is a space or a tab character. Splits the argument by those characters and add it to args array. Also if there is any & character in the command line, it sets the value of background to 1.

*FindPath method gets the path environment variable, copies it into a string and splits that string by : character.

-For the path command, it prints the whole path lists; for the path + pathname command, adds that path to the path lists and for the path – pathname command, deletes the all occurences of that pathname from the path list.

-But we had a problem with those path commands. For every call of findPath method, getenv method calls again and again. Because of these adding and deletions operations don’t effect the getenv, these operations works only once when they are runned.

-This method also concatanate the command line argument to every path and check if there is an executable file like that by calling isFileExists method.

*IsFileExists method returns 1, if fopen method can open a file with that filename.

*SplitByAmpersandOrSemicolon method was for the bonus part, but we couldn’t implement it correctly.

*Push method adds a new command to the commands list with the calling arguments name. It is called when a new command line argument entered.

*ReverseDisplay method prints the command linked list from the end of the list. It is a recursive method.

*Pop method deletes the first added command from the linked list. It is runned when the size of the linked list is greater than 10.

*GetNth method gives the name of the command from the linked list with the given index value.

*GetLength method calculates the length of the linked list.

*EnqueueBackgroundQ method adds a new background process  to the end of the queue.

*DeleteByPid method deletes the process with the given pid from the queue.

*SignalHandler method is called when the correspoding signal (ctrl+z) occurs.

-It  kills the current foreground process if there is anyn. If there is not, it continues. Kill method sends a signal to that process id.

-If the signal  reports an error within the program it prints an error message.

*ChilPidHandler method uses WNOHANG, waitpid() to check if any zombie-children exist. If there is, its exit status returned. If not, either 0 returned (unterminated children exist) or -1 returned ERRNO is set to ESRCH (no child processes)

* Execute method takes the path of the command, the command array and background value of that command.

-It forks a child. Child executes the command with execv method.

-Parent makes operations by the background value. If the current process is foreground, it sets the isThereAnyForeGroundProcess and currentForegroundProcess values. And calls waitpid with childpid. Also initialize the signal child handler.

-If it is a background process (& character enetered to the and of the command), it adds that process to the background queue by calling enqueueBackgroundQ method and prints the information of that process.

*RemoveChar method removes the specific character from the string. We used this method to delete % character from the foreground process id.

*Main method is a bit crowded. At first we tried to make every operation in seperate methods but because of the pointers are so complicated, we couldn’t.

-At first it initializes the sigaction struct with signal handler and sa_restart flag.

-There is a while loop to execute command line arguments until control d signal entered.

-Within the loop at first setup function is called. For every loop we add the command to mergedArgs array (if the command is not history) to list for the history command. If the length of that list is greater than 10, we delete the first added command from the list with pop method.

-If the command is history, it prints the list with reverseDisplay method.

-If the command contains <, it uses the elements of the given filename to execute the command.  But after this command executed, program terminates.

-If the command contains >>, it prints the result of the executable to the given file. If the file does not exist, it creates a new one. If the file already exists it appends the result to the list. This part also has errors. After this command executed, program prints every command to the list and never returns.

-If the command contains 2>, it prints the error of the executable to the given file.

-If the command contains path, it goes to findPath method and do operations.

-If the command is exit and there are background processes it prints an error message. If there is no background processes, then it exits.

-If the command is fg with %pid, at first it deletes the % character and then deletes the element from the background queue with that pid.

-If the command is history with an index value, first it finds the command from the args list, copies the content to the newArgs array, deletes the space character at the end of the args string, adds null to the next index, finds the path of that command and executes the command. This one also has errors. It only executes the first string of the command. Because it deletes everything after the space character.

-If the command is anything else from the upper commands, it finds the path of the command and executes.