1. How Code is Organized:

Firstly, I managed to divide the code into modules, each module handles a part in the process of reading, writing and executing command.

a- Main:

This is the main module that is responsible for calling the correct mode from two modes: interactive and batch mode, and fires its function.

b- fileHandler:

This module is responsible for handling reading commands from file in the batch mode, the handler will return an array of strings each element contains the lines of the file in each string and it will return array contains NULL, if we couldn't find or read the file.

c- Interactive:

This module contains a while loop that run infinitely, until CTRL-D or CTRL-C are triggered, this module takes each command and send it to another module to process the command and execute it.

d- Batch:

This is the module which is responsible for batch mode as it calls the fileHandler module to read the file and get the commands array to execute and process them sequentially.

e- singleCommandHandler:

this module gets single command and process it by splitting it to arguments and commands and execute it.

f- Vector:

This is a data structure module, that is responsible to get the string and save it in the vector, it helps the user to get a copy from what is saved in the vector.

g- Basic:

This module is responsible for the commands that are executed by the (execv) function, as it has one function that call the (execv) function over multiple paths, that saved in the PATH environmental variable saved in my shell, and prints error if all paths in the PATH variable make the (execv) returns -1 or stop if one of the (execv) functions is executed successfully.

h- Special:

This module is responsible for the commands that aren’t executed by the (execv) function, like cd, export, echo, ...etc. It contains only one function that take the argument list and check if they belong to one of those function, if it belongs, then it returns 0 and set the valid function which is a pointer to a function to the correct function to be executed.

i- Executer:

This module is responsible for executing the proper function for the sent command, It has one function, that is responsible for forking, and creating a child to execute the function, that is sent to the execute as a parameter.

j- Environment:

Contains the environmental variables that is used by this my shell, also contains a vector of key value pair that contains the variable as a key and its value as the value of that key, it also contains a get and insert functions to get and add new or modify existing variable.

k- History:

Responsible for writing the commands that is got by the user in batch mode or in interactive mode.

l- Logger:

Used for appending messages that the program writes during the execution.

m-Stringparsing:

This is responsible for parsing the whole line of the command and parse it, by dividing it into arguments and remove the variable used in the command by its' values, and return a state data structure that describe the state of the command sent to it.

n- commandState:

This is just a header file that contains a data structure, that save the state of the command, and describe it by the following attributes:

I. arg_list: the argument list of the command.
II. Background: flag that tells if the command will run in the back ground or not.
III. Sz: the size of the arg_list
IV. Mask: it’s number used for command syntax checking.
V. forcedBasic: it’s a flag to tell that if it command must ran as basic command, must be ran by the (execv) function.

2- Main Functions:

a. Main: main function to launch the program it run in two modes according to the number of the arguments that it has taken.

b. Parse_command: used to parse the command it takes and return the commandState data structure that contains the state of the command sent to function.

c. Basic: function that run the commands which are the executed by (execv) function in the child process.

d. Special: used to determine which special function will run according to the state of the command.

e. Execute: this is one of the most important functions in the shell, as it’s responsible for running and executing the command after parsing it.

f. Interactive: used to run the interactive mode.

g. Batch: used to run the batch mode.

h. Push_back: it’s a function in the vector data structure used to push new elements at the end of the vector.

i. copyOfArray: used to get a copy from the current elements in the vector.

j. Insert_variable: used to add new or modify old environmental variable.

k. Get_variable: used to get the environmental variable that saved in the table, or it return an empty string if it doesn’t exist.

l. readFile: used to read the file of commands in the batch mode.

m. appendToLogger: used to write in the logger file.

n. appendToHistory: used to write in the history file.

3- How to compile Code:

a- You can compile the code by, firstly running the makeFile by writing make in the linux shell, to compile the code and create the object file.

b- To run in the interactive mode, you should to run the program only by writing with no other arguments, e.g. ./output

c- To run in the batch mode, you should run the program by writing the name of the program then the file you want to read from. E.g. ./output file