1. Shell (shell.c):

- The program starts by displaying a prompt that shows the current working directory, username, and hostname.
- It enters into a loop where it waits for user input.
- The user's input is tokenized into a command and its arguments.
- Depending on the command, the program can execute various functionalities like the word command, date command, and dir command.
- Error handling Monitored.
- The shell continues to loop until the user enters "end".

2. Word Count Command (word.c):

- This command takes a filename as an argument and counts the number of words in the file.
- It uses the word_counter function to perform the word counting, considering spaces, tabs, and newlines as word separators.
- It can also handle options like n and d, which affect how the word count is displayed or calculated. Most difficult of this was n. The newline character were to be ignored so we used flags.

3. Directory Command (dir.c):

- This command handles directory creation, checking if a directory exists, and recursively removing directories.
- For -v I have taken the dir to be performing the same operation as -r But it just prints those. This also tells the status of current directory time to time.
- Depending on the provided arguments, it creates a new directory, checks if a directory exists, or removes directories.
- The deleteDirectoryRecursively function recursively deletes directories and their contents(r). We have to recursively delete a directory as if there exists something in it , it will cause problem.

4. Date Display Command (date.c):

- This command displays the last modified time of a file in readable format.
- #include <time.h> Main function
- It uses the stat function to retrieve file information and the localtime function to convert the timestamp into a readable format.
- It can handle options like d and R to modify the output format of the date.
- For -R it give time in RFC 5322 format in which day date month Year time ctd.

5. Makefile (Makefile):

- The provided Makefile contains rules for building the executable binaries for each command.
- The all target compiles all the commands (dir, date, shell, word).
- The clean target removes the compiled binaries.
- It uses -Wall and -Wexcept for better secure code.