# CSE3033 Operating Systems Project#2 Report

Terminal


**Students**

Emir Büçkün

Alperen Koruyucu

Melih Apaydın

**Instructor**

Assoc. Prof. Dr. Ali Haydar Özer


**Date**

19 December 2022

# Overview

In this project, we tried to implement our own terminal which runs on a Linux distribution. The program created with C language. This program has its own unique operations as well as other ordinary operations. All errors printed in stderr. Our program supports these features:

- Executing commands which are already loaded into our system as foreground and background process.
- Executing some extra commands which are not already loaded into our system like “history”, “^Z” and “exit”.
- Using I/O redirections:
  - “<“ - getting input from a file.
  - “>“ - printing output to a file.
  - “>>“ - printing output to a file (append).
  - “2>“ - printing errors to a file.
  - “2>>“ - printing errors to a file (append).

# Part A

In this first part, we need a shell that supports to run programs which are already loaded into our system like “ls”, “clear”, “gedit” and etc. In our shell, this is handled by creating a new child by using “fork()” function and running corresponding program by using “execv()” function. Firstly, path of that program is found by using “findPathOf” function. This function takes two parameters. These are “path” and “exe”. If the path of the program is determined, then path is loaded into “path” variable. Then this path is given to the “execv()” function.

If user enters “&” sign at the end, then the program should be run in the background. If background variable is one, our child is run in the background

using “waitpid” function. This function is called at each new child creation because of signal handler.

## Part B

Now, we need some extra command supports which are not loaded into our system. These commands are “history”, “^Z” and “exit”.

The purpose of the “history” command is to list the lastly executed ten commands. They are printed on the screen when this command is used.

The purpose of the “^Z” command is to stop the currently running foreground process and its all children. If there is no any foreground process, then the signal does nothing.

The purpose of the “exit” command is to shut down our shell. If there is any background processes, it will print a message about that and the shell will not be turned off until there is no any background process.

## Part C

We also need I/O redirections support for our shell. We have five different symbol to make I/O redirections. These are “>”, “>>”, “2>”, “2>>” and “<“. If all of the symbols are placed correctly with correct number of arguments in the correct order, then operation type, input file name and output file name is assigned to the corresponding variables. After that, shell cuts the command and take just program name with arguments. Then shell runs the program same as Part A. Before running “execv” function in the child part, shell checks if there will be any redirections or not. Redirections flags are used to handle that. According to the different situations, “inputRedirection” and “outputRedirection” functions are called.