Assignment # 02

Muhammad Shafeen
FAST University Peshawar
Department of Computer Science
Course: Operating System
Instructor: Saad Ahmad

Due Date: 25th October 2024
Time: 5:00 PM

Contents

1 Exec System Calls

The exec() family of functions replaces the current process image with a new process image. The functions described in this manual page are layered on top of execve(2). (See the manual page for execve(2) for further details about the replacement of the current process image.)

The initial argument for these functions is the name of a file that is to be executed.

The functions can be grouped based on the letters following the "exec" prefix.

1.1 execl

1.1.1 Purpose and Usage

Explanation as per manual of execv( ) The const char *arg and subsequent ellipses can be thought of as arg0, arg1, ..., argn. Together they describe a list of one or more pointers to null-terminated strings that represent the argument list available to the executed program. The first argument, by convention, should point to the filename associated with the file being executed. The list of arguments must be terminated by a null pointer, and, since these are variadic functions, this pointer must be cast (char *) NULL.

By contrast with the 'l' functions, the 'v' functions (below) specify the command-line arguments of the executed program as a vector

1.1.2 C Code Example

```c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main() {
    printf("Before execl: Listing directory contents\n");
    pid_t pid;
    pid=fork();
    if(pid==0)
{

    printf("Child with PID : %d  , running execl: \n",getpid());
    sleep(1);
    if (execl("/bin/ls", "sudo ls", "-l", "/media/shafeenyousafzai/
   OLDERDRIVE/Semester 5/Operating-System-Lab/Assignment # 2", (char *)
   NULL) == -1) {
        perror("execl failed");
        exit(EXIT_FAILURE);
    }

 }
 else{
 sleep(2);
 int x=0;
 int y=1;
 int sum=x+1;
 printf("Parent with PID : %d is printing the sum of %d and %d with answer
    : %d\n",getppid(),x,y,sum);

 pid_t pid2;
 pid2=fork();
 if(pid2==0){
```

```c
sleep(1);
printf("In the child code : " \n);
execl("
   return 0;
}
}
```

Listing 1: Using execl to execute `ls` command

1.1.3 Explanation

In this code, `execl` is used to execute the `ls -l` command. The first argument is the path to the executable, followed by the argument list terminated by `NULL`. If `execl` fails, an error message is printed.

1.1.4 Practical Scenario

`execl` can be used in a program that needs to execute another program without creating a new process. For example, a shell might use `execl` to run user commands.

1.2 execlp

1.2.1 Purpose and Usage

Explanation as per manual of execv( ) The const char *arg and subsequent ellipses can be thought of as arg0, arg1, ..., argn. Together they describe a list of one or more pointers to null-terminated strings that represent the argument list available to the executed program. The first argument, by convention, should point to the filename associated with the file being executed. The list of arguments must be terminated by a null pointer, and, since these are variadic functions, this pointer must be cast (char *) NULL.

By contrast with the 'l' functions, the 'v' functions (below) specify the command-line arguments of the executed program as a vector

1.2.2 C Code Example

```c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main() {
    printf("Opening TeXstudio with a custom environment\n");

    pid_t pid;
    pid=fork();
    if(pid==0){
    printf("This is the parent process with pid : %d \n", getppid());
    printf("Calculating division of two numbers : \n");
    int x = 10;
    float y =  2;
    float div=x/2;
    printf("Answer : %f \n",div);
    }
    else{
    sleep(2);
    printf("Running the command with display variable set to 1");
    if (execlp("/usr/bin/texstudio", "texstudio","USER=latex_user","PATH=/
   usr/bin:/bin","CUSTOM_VAR=HelloTeX","HOME=/home/latex_user","DISPLAY=:1
   ", (char *)NULL) == -1) {
        perror("execle failed");
        exit(EXIT_FAILURE);
    }

    printf("If you see this, execle failed\n");
    return 0;
}
}
```

Listing 2: Using execlp to execute `ls` command

1.2.3 Explanation

This C program demonstrates basic process creation and command execution using the fork() and execlp() system calls. When executed, the program first prints a message indicating it will open TeXstudio with a custom environment. It then creates a child process using fork(). In the child process (pid == 0), it incorrectly labels itself as the parent, performs a simple division of two numbers, and prints the result. Meanwhile, the parent process waits for two seconds, prints a message about running TeXstudio with specific environment variables, and attempts to launch the TeXstudio application with customized settings such as USER, PATH, CUSTOM_VAR, HOME, and DISPLAY. If the execlp() call fails, it prints an error message and exits. This program effectively illustrates how to manage child and parent processes and execute external applications with tailored environments.

1.2.4 Practical Scenario

`execlp` is useful when the exact location of the executable is not known, and you want the system to search for it in the directories listed in the `PATH` environment variable.

1.3 execle

1.3.1 Purpose and Usage

The `execle` system call is similar to `execl`, but it allows the caller to specify the environment for the new process.

1.3.2 C Code Example

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main() {
    printf("Opening TeXstudio with a custom environment\n");

    pid_t pid;
    char *custom_env[] = {
        "USER=latex_user",       // another user for latex
        "PATH=/usr/bin:/bin",    // the path where installed application
   can be found
        "CUSTOM_VAR=HelloTeX",   // custom variable
        "HOME=/home/latex_user", // Custom home directory
```

```c
        "DISPLAY=:0",              // Set the display to :1 so that the opened application is opened on the current screen
  NULL
    };
      char *custom_env2[] = {
        "USER=latex_user",         // another user for latex
        "PATH=/usr/bin:/bin",      // the path where installed application can be found
        "CUSTOM_VAR=HelloTeX",     // custom variable
        "HOME=/home/latex_user",  // Custom home directory
        "DISPLAY=:1",              // Set the display to :1 so that the opened application is opened on the current screen
  NULL
    };
    pid=fork();
    if(pid==0){
    // Replace the current process with the TeXstudio command
    if (execle("/usr/bin/texstudio", "texstudio", (char *)NULL, custom_env) == -1) {

        perror("execle failed for display variable 0");
        sleep(1);
        exit(EXIT_FAILURE);
    }
    }
    else{
    sleep(2);
    printf("Running the command with display variable set to 1");
    if (execle("/usr/bin/texstudio", "texstudio", (char *)NULL, custom_env2) == -1) {
        perror("execle failed");
        exit(EXIT_FAILURE);
    }

    // This line will not be executed if execle is successful
    printf("If you see this, execle failed\n");
    return 0;
}
}
```

Listing 3: Using execle to execute `printenv` command with custom environment

1.3.3 Explanation

Using 'fork()' and 'execle()' to launch the TeXstudio application with custom environment variables. The program forks a new process. In the child process ('pid == 0'), it attempts to run TeXstudio with a custom environment ('custom_env') on display ':0'. In the **parent process**, it waits for two seconds and then attempts to launch TeXstudio with a different custom environment ('custom_env2') on display ':1'. If either 'execle()' call fails, it prints an error message. The line after 'execle()' won't execute if 'execle()' succeeds, as the process is replaced.

1.3.4 Practical Scenario

`execle` is useful when you need to set specific environment variables for the new process, such as customizing the `PATH` or other environment settings.

1.4 execv

1.4.1 Purpose and Usage

The `execv` system call replaces the current process image with a new process image specified by the given path. It takes an array of arguments instead of a variable number of arguments.

1.4.2 C Code Example

```c
// execvp_example.c
// execv_example.c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main() {
    printf("Running Execve to run texstudio app : ");

    char *args[] = { "texstudio", (char *)NULL,"USER=latex_user","PATH=/
   usr/bin:/bin","CUSTOM_VAR=HelloTeX","HOME=/home/latex_user","DISPLAY=:1
   " };


    pid_t pid;
  pid=fork();
    if(pid==0){
    // Replace the current process with the TeXstudio command
    if (execv("/usr/bin/texstudio", args) == -1) {
        perror("execv failed");
        exit(EXIT_FAILURE);
    }
    }
    else{
    sleep(3);
    printf("Running the command with display variable set to 1");
    if (execv("/usr/bin/texstudio", args) == -1) {
        perror("execv failed");
        exit(EXIT_FAILURE);
    }

    return 0;
}
}
```

Listing 4: Using execv to execute `ls` command

1.4.3 Explanation

Use of fork() and execv() to run the TeXstudio application in both the child and parent processes. After forking, the child process attempts to replace its process with TeXstudio using execv(), passing a set of arguments that includes environment variables like USER, PATH, and DISPLAY. If this execution fails, an error message is printed. In the parent process, after a brief sleep, it also attempts to run TeXstudio with the same arguments. If execv() fails in the parent process, an error message is printed, and the program exits.

1.4.4 Practical Scenario

`execv` is ideal when the arguments to the new program are already available in an array, such as when parsing command-line arguments or handling user input.

1.5 execve

1.5.1 Purpose and Usage

The `execve` system call is the underlying system call that all other `exec` functions eventually call. It allows the caller to specify both the argument list and the environment.

1.5.2 C Code Example

```c
// execve_example.c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main() {
    printf("Before execve: Displaying environment variables using '
   printenv'\n");


    char *args[] = { "texstudio", (char *)NULL,"USER=latex_user","PATH=/
   usr/bin:/bin","CUSTOM_VAR=HelloTeX","HOME=/home/latex_user","DISPLAY=:1
   " };
   char *custom_env2[] = {
        "USER=latex_user",
        "PATH=/usr/bin:/bin",
        "CUSTOM_VAR=HelloTeX",
        "HOME=/home/latex_user",
        "DISPLAY=:1",
  NULL
    };


    pid_t pid;
  pid=fork();
    if(pid==0){
    // Replace the current process with the TeXstudio command
    if (execve("/usr/bin/texstudio", args, custom_env2) == -1) {
        perror("execve failed");
        exit(EXIT_FAILURE);
    }
    }
    else{
    sleep(3);
    printf("Running the command with display variable set to 1");
    if (execve("/usr/bin/texstudio", args, custom_env2) == -1) {
        perror("execve failed");
        exit(EXIT_FAILURE);
    }

    return 0;
}
}
```

Listing 5: Using execve to execute `env` command with custom environment

1.5.3 Explanation

use of fork() and execve() to launch the TeXstudio application with custom environment variables Before Fork: It prints a message indicating the display of environment variables using printenv. In the child process (pid == 0), execve() is used to replace the current process with the TeXstudio application, passing custom environment variables (custom_env2). If execve() fails, an error message is printed, and the child process exits.

In the parent process, after a 3-second delay, it also attempts to run TeXstudio with the same execve() call and environment variables. If it fails, an error message is printed, and the program exits.

1.5.4 Practical Scenario

`execve` is useful when you need complete control over both the arguments and the environment of the new process, such as in low-level process management or when implementing shells.

1.6 execvp

1.6.1 Purpose and Usage

The `execvp` system call is similar to `execv`, but it uses the `PATH` environment variable to search for the executable if the specified path does not contain a slash.

1.6.2 C Code Example

```c
// execvp_example.c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main() {
    printf("using grep to find texstudio and execute it\n");

    // Argument list for 'grep main execvp_example.c'
    char *args[] = { "texstudio", (char *)NULL,"USER=latex_user","PATH=/usr/bin:/bin","CUSTOM_VAR=HelloTeX","HOME=/home/latex_user","DISPLAY=:1"};


    if (execvp("grep", args) == -1) {
        perror("execvp failed");
        exit(EXIT_FAILURE);
    }
    return 0;
}
```

Listing 6: Using execvp to execute `grep` command

1.6.3 Explanation

Use execvp() to run the grep command, with arguments that seem intended for launching the TeXstudio application. However, the arguments provided (texstudio, USER=latex_user, etc.) do not align with how grep typically works. The first argument for grep should be the search pattern (e.g., "main"), followed by the file to search within (e.g., "execvp_example.c").

1.6.4 Practical Scenario

`execvp` is beneficial when executing commands that are expected to be found in the system's `PATH`, such as common utilities or user-installed programs.

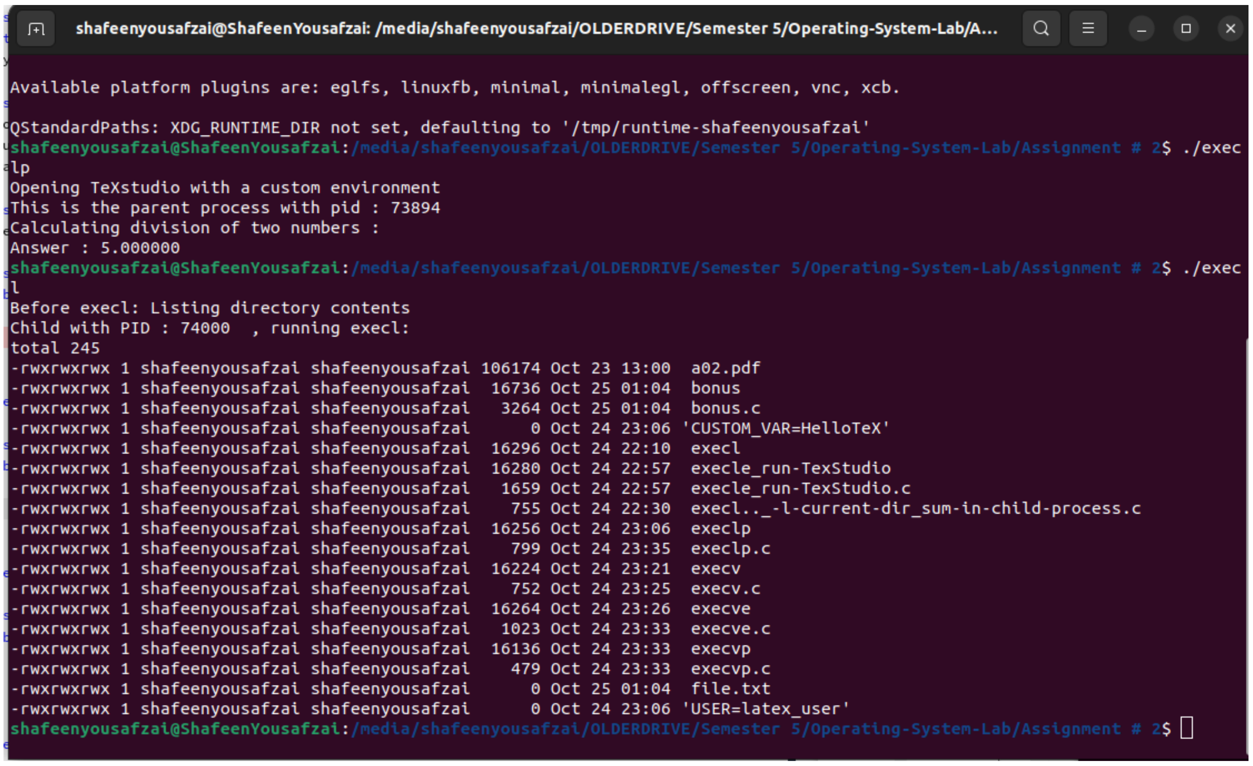
2 Code Documentation

For each `exec` system call, the corresponding C code files are provided in the submission. Each code file includes comments explaining the functionality and usage of the system call.

2.1 Screenshots and Terminal Output

Below are examples of the execution results for each `exec` system call.

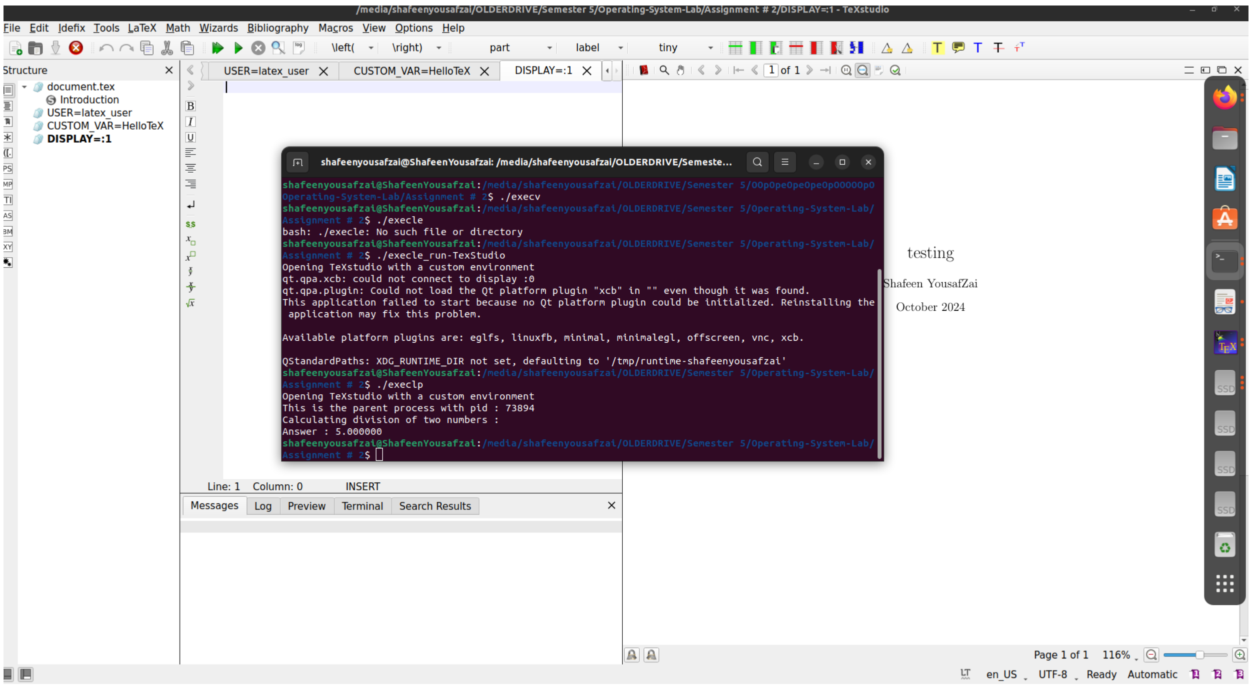
2.1.1 execl Execution



```
shafeenyousafzai@ShafeenYousafzai: /media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/A...

Available platform plugins are: eglfs, linuxfb, minimal, minimalegl, offscreen, vnc, xcb.

QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/Assignment # 2$ ./exec
lp
Opening TeXstudio with a custom environment
This is the parent process with pid : 73894
Calculating division of two numbers :
Answer : 5.000000
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/Assignment # 2$ ./exec
l
Before execl: Listing directory contents
Child with PID : 74000  , running execl:
total 245
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai 106174 Oct 23 13:00  a02.pdf
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16736 Oct 25 01:04  bonus
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai   3264 Oct 25 01:04  bonus.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai      0 Oct 24 23:06 'CUSTOM_VAR=HelloTeX'
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16296 Oct 24 22:10  execl
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16280 Oct 24 22:57  execle_run-TexStudio
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai   1659 Oct 24 22:57  execle_run-TexStudio.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai    755 Oct 24 22:30  execl.._-l-current-dir_sum-in-child-process.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16256 Oct 24 23:06  execlp
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai    799 Oct 24 23:35  execlp.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16224 Oct 24 23:21  execv
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai    752 Oct 24 23:25  execv.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16264 Oct 24 23:26  execve
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai   1023 Oct 24 23:33  execve.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16136 Oct 24 23:33  execvp
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai    479 Oct 24 23:33  execvp.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai      0 Oct 25 01:04  file.txt
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai      0 Oct 24 23:06 'USER=latex_user'
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/Assignment # 2$ 
```

Figure 1: Terminal Output for execl

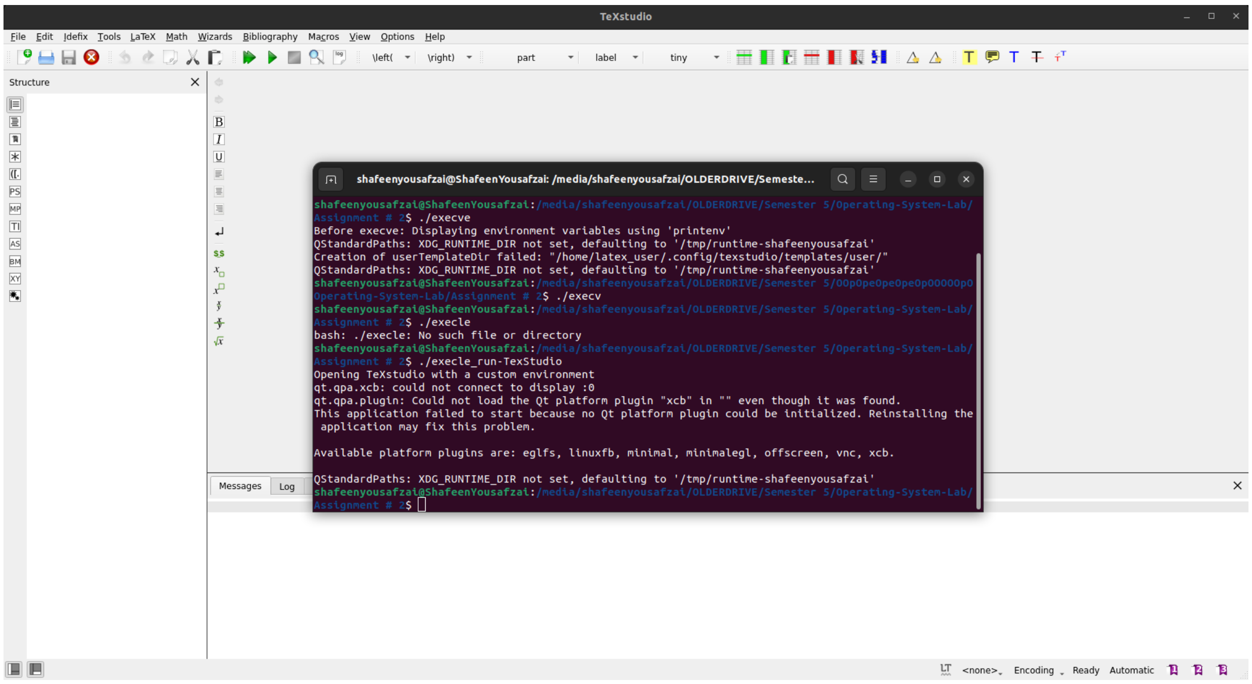
2.1.2 execlp Execution



```
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/OOpOpeOpeOpeOpOOOOOpO
Operating-System-Lab/Assignment # 2$ ./execv
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execle
bash: ./execle: No such file or directory
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execle_run-TexStudio
Opening TeXstudio with a custom environment
qt.qpa.xcb: could not connect to display :0
qt.qpa.plugin: Could not load the Qt platform plugin "xcb" in "" even though it was found.
This application failed to start because no Qt platform plugin could be initialized. Reinstalling the
 application may fix this problem.

Available platform plugins are: eglfs, linuxfb, minimal, minimalegl, offscreen, vnc, xcb.

QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execlp
Opening TeXstudio with a custom environment
This is the parent process with pid : 73894
Calculating division of two numbers :
Answer : 5.000000
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$
```

Figure 2: Terminal Output for execlp

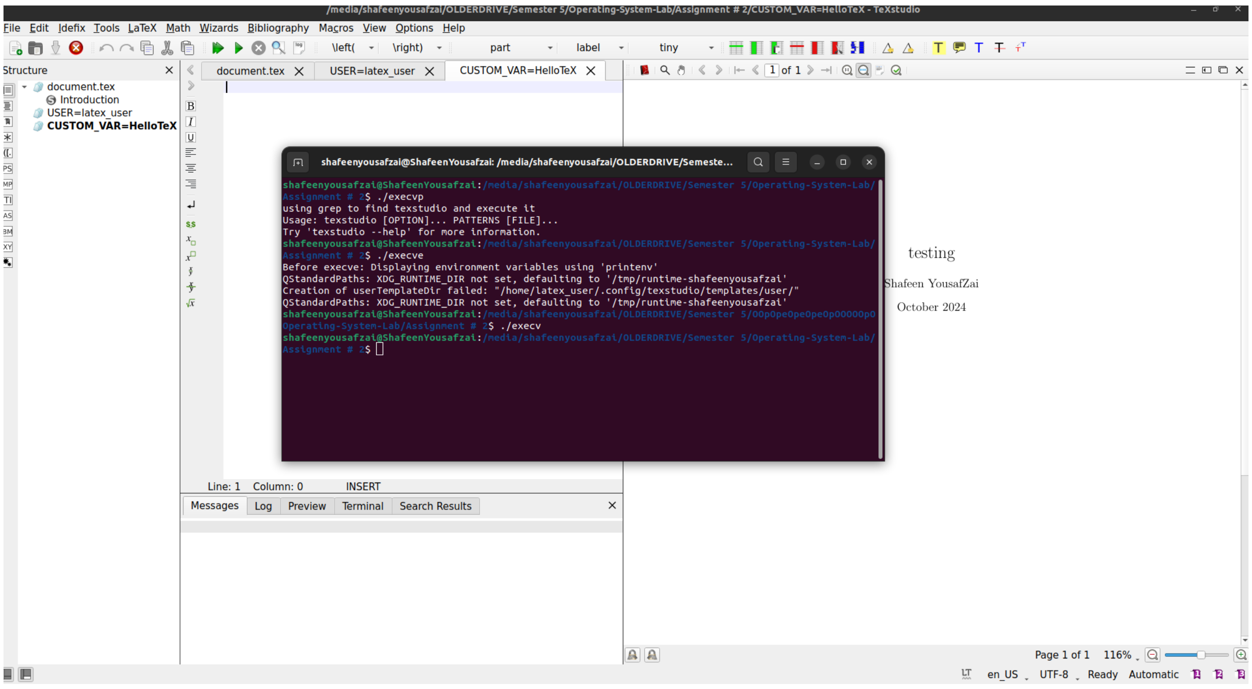
2.1.3 execle Execution



```
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execve
Before execve: Displaying environment variables using 'printenv'
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
Creation of userTemplateDir failed: "/home/latex_user/.config/texstudio/templates/user/"
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/OOpOpeOpeOpeOpOOOOOpO
Operating-System-Lab/Assignment # 2$ ./execv
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execle
bash: ./execle: No such file or directory
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execle_run-TexStudio
Opening TeXstudio with a custom environment
qt.qpa.xcb: could not connect to display :0
qt.qpa.plugin: Could not load the Qt platform plugin "xcb" in "" even though it was found.
This application failed to start because no Qt platform plugin could be initialized. Reinstalling the
 application may fix this problem.

Available platform plugins are: eglfs, linuxfb, minimal, minimalegl, offscreen, vnc, xcb.

QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$
```

Figure 3: Terminal Output for execle

2.1.4 execv Execution

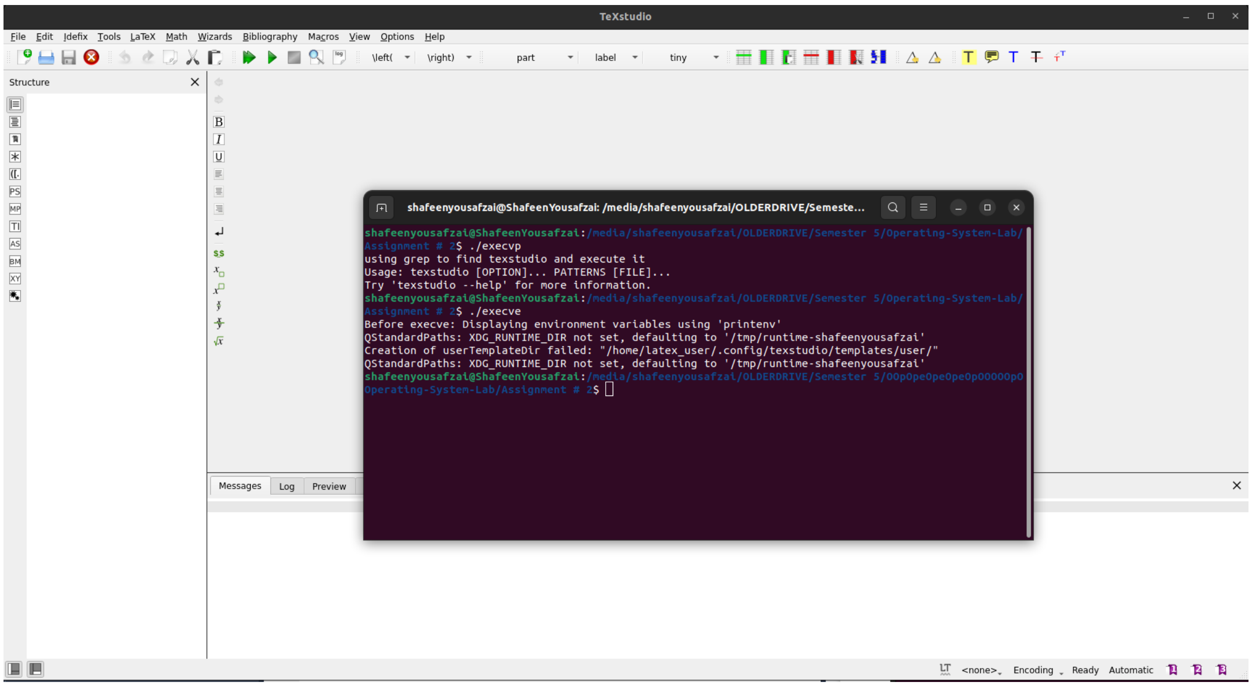


```
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execvp
using grep to find texstudio and execute it
Usage: texstudio [OPTION]... PATTERNS [FILE]...
Try 'texstudio --help' for more information.
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execve
Before execve: Displaying environment variables using 'printenv'
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
Creation of userTemplateDir failed: "/home/latex_user/.config/texstudio/templates/user/"
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/OOpOpeOpeOpeOpOOOOOpO
Operating-System-Lab/Assignment # 2$ ./execv
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ 
```

/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/Assignment # 2/CUSTOM_VAR=HelloTeX - TeXstudio
Structure
document.tex
Introduction
USER=latex_user
CUSTOM_VAR=HelloTeX
USER=latex_user
CUSTOM_VAR=HelloTeX
testing
Shafeen YousafZai
October 2024
Line: 1 Column: 0 INSERT
Messages Log Preview Terminal Search Results
Page 1 of 1 116%
en_US UTF-8 Ready Automatic

Figure 4: Terminal Output for execv

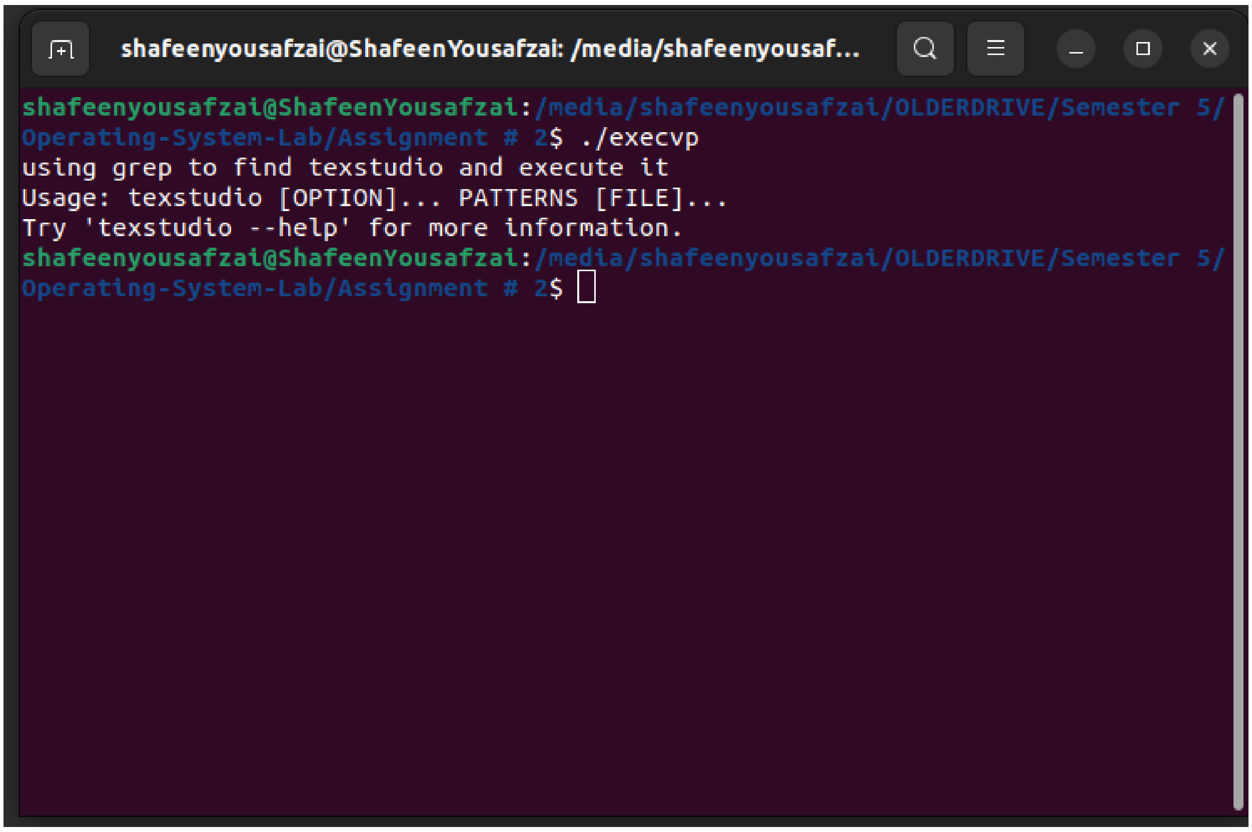
2.1.5 execve Execution



```
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execvp
using grep to find texstudio and execute it
Usage: texstudio [OPTION]... PATTERNS [FILE]...
Try 'texstudio --help' for more information.
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/
Assignment # 2$ ./execve
Before execve: Displaying environment variables using 'printenv'
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
Creation of userTemplateDir failed: "/home/latex_user/.config/texstudio/templates/user/"
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-shafeenyousafzai'
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/OOpOpeOpeOpeOpOOOOOpO
Operating-System-Lab/Assignment # 2$ 
```

Figure 5: Terminal Output for execve

2.1.6 execvp Execution



```
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Assignment # 2$ ./execvp
using grep to find texstudio and execute it
Usage: texstudio [OPTION]... PATTERNS [FILE]...
Try 'texstudio --help' for more information.
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Assignment # 2$ 
```

Figure 6: Terminal Output for execvp

3 Bonus Task (Optional)

3.1 Mini-Shell Implementation

3.1.1 C Code

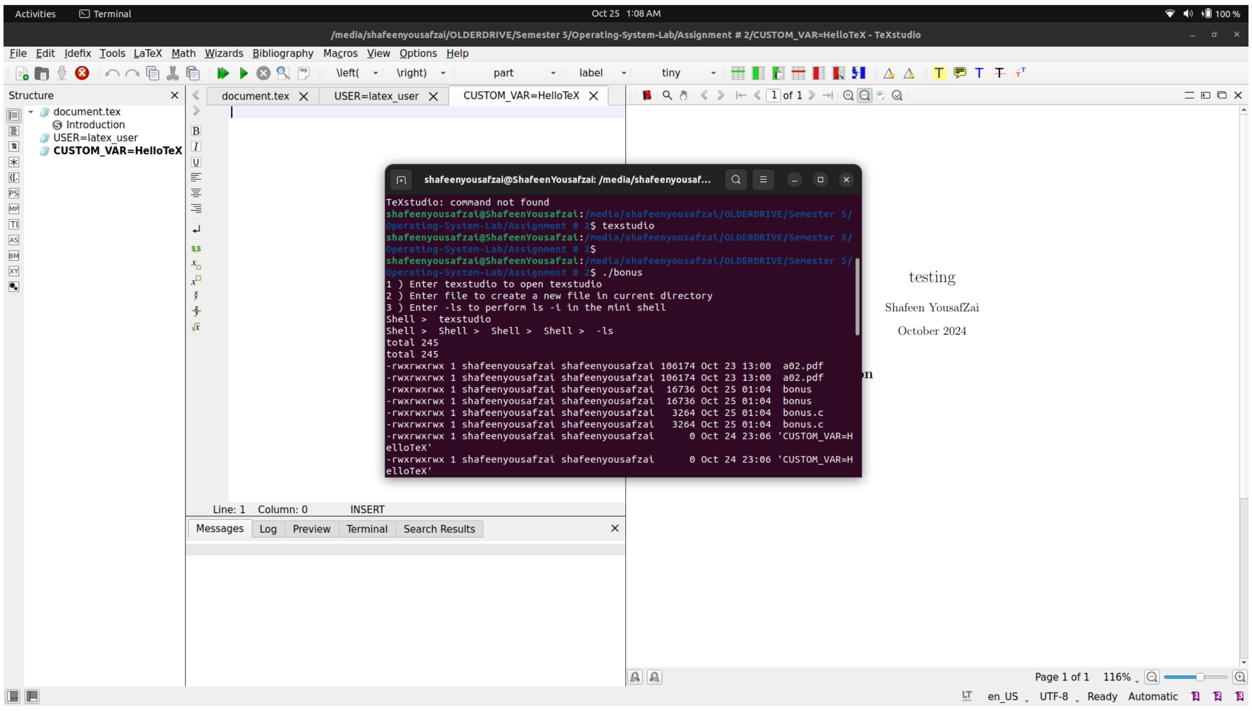
```c
// simple_mini_shell.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/wait.h>

#define MAX_INPUT_SIZE 1024    // Maximum length of the input line
#define MAX_ARG_COUNT 64       // Maximum number of arguments

int main() {
    char input[MAX_INPUT_SIZE];
    char *args[MAX_ARG_COUNT];
    pid_t pid;
    int status;
    printf("1 ) Enter ""texstudio"" to open texstudio\n");
    printf("2 ) Enter ""file"" to create a new file in current directory\n
");
    printf("3 ) Enter ""-ls"" to perform ls -i in the mini shell\n");
    do{

        // 1. Read: Display prompt and get user input ; using Sir.Muhammad
 Nouman's given logic of read, eval , execute , loop.
        printf("Shell >  ");
        fflush(stdout);

        if (fgets(input, sizeof(input), stdin) == NULL) {
            printf("\n");
            break;
        }

        // Remove newline character from input
        input[strcspn(input, "\n")] = '\0';

        // Skip empty input
        if (strlen(input) == 0) {
            continue;
        }

        // 2. Eval: Parse the input into command and arguments
        int arg_index = 0;
        args[arg_index] = strtok(input, " ");
        while (args[arg_index] != NULL && arg_index < MAX_ARG_COUNT - 1) {
            args[++arg_index] = strtok(NULL, " ");
        }
        args[arg_index] = NULL;

        // Handle built-in 'exit' command
        if (strcmp(args[0], "exit") == 0) {
            printf("Exiting mini-shell.\n");
            break;
        }
        pid_t pid;
        pid=fork();
        if(pid==0){
        if (strcmp(args[0], "texstudio") == 0) {
            char *args[] = { "texstudio", (char *)NULL,"USER=latex_user","
 PATH=/usr/bin:/bin","CUSTOM_VAR=HelloTeX","HOME=/home/latex_user","
 DISPLAY=:1" };
            execv("/usr/bin/texstudio", args);
            break;
        }}
        sleep(2);

        pid_t pid2;
        pid2=fork();
                if(pid2==0){
        if (strcmp(args[0], "-ls") == 0) {
            execl("/bin/ls", "sudo ls", "-l", "/media/shafeenyousafzai/
 OLDERDRIVE/Semester 5/Operating-System-Lab/Assignment # 2", (char *)
 NULL);
            break;
        }
        }
        sleep(2);

 pid_t pid3;
        pid3=fork();
```

```
                if(pid3==0){
        if (strcmp(args[0], "file") == 0) {
            execl("/bin/touch", "touch", "/media/shafeenyousafzai/
    OLDERDRIVE/Semester 5/Operating-System-Lab/Assignment # 2/file.txt", (
    char *)NULL);
            break;
        }
        }
        sleep(2);
        // 3. Execute the command
        pid = fork();

        if (pid < 0) {
            perror("fork failed");
            continue;
        }

        if (pid == 0) {
            //  Execute the command
            if (execvp(args[0], args) == -1) {
                perror("exec failed");
            }
            exit(EXIT_FAILURE);
        } else {
            // Wait for the child to finish
            do {
                if (waitpid(pid, &status, WUNTRACED) == -1) {
                    perror("waitpid failed");
                    break;
                }
            } while (!WIFEXITED(status) && !WIFSIGNALED(status));
        }
    }while(1);

    return 0;
}
```

Listing 7: Mini-Shell using execvp

3.1.2 Explanation

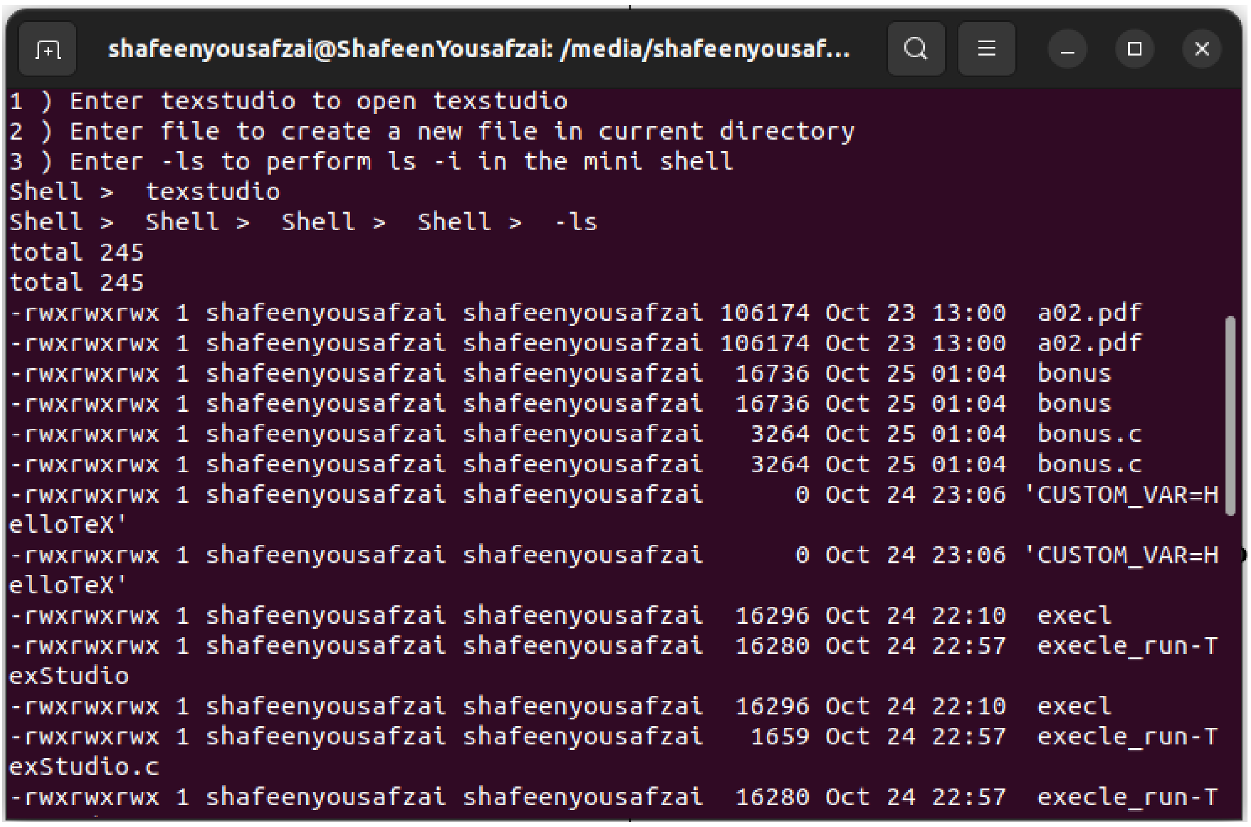
Approach : Read , eval , print , loop . This mini-shell continuously prompts the user for input, parses the command and its arguments, and uses `execvp` ,`execv` ,`execl` to execute the command. The shell supports basic commands and allows the user to exit by typing also allows user to open a latex editor texstudio by typing and also find ls -i and create a file`exit`.


/media/shafeenyousafzai/OLDERDRIVE/Semester 5/Operating-System-Lab/Assignment # 2/CUSTOM_VAR=HelloTeX - TeXstudio
document.tex
Introduction
USER=latex_user
CUSTOM_VAR=HelloTeX
testing
Shafeen YousafZai
October 2024

```
TeXstudio: command not found
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Assignment # 2$ texstudio
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Assignment # 2$
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Assignment # 2$ ./bonus
1 ) Enter texstudio to open texstudio
2 ) Enter file to create a new file in current directory
3 ) Enter -ls to perform ls -i in the mini shell
Shell >  texstudio
Shell >  Shell >  Shell >  Shell >  -ls
total 245
total 245
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai 106174 Oct 23 13:00  a02.pdf
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai 106174 Oct 23 13:00  a02.pdf
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16736 Oct 25 01:04  bonus
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16736 Oct 25 01:04  bonus
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai   3264 Oct 25 01:04  bonus.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai   3264 Oct 25 01:04  bonus.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai      0 Oct 24 23:06 'CUSTOM_VAR=H
elloTeX'
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai      0 Oct 24 23:06 'CUSTOM_VAR=H
elloTeX'
```

Figure 7: Mini-Shell Execution

3.1.3 Screenshot



```
1 ) Enter texstudio to open texstudio
2 ) Enter file to create a new file in current directory
3 ) Enter -ls to perform ls -i in the mini shell
Shell >  texstudio
Shell >  Shell >  Shell >  Shell >  -ls
total 245
total 245
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai 106174 Oct 23 13:00  a02.pdf
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai 106174 Oct 23 13:00  a02.pdf
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16736 Oct 25 01:04  bonus
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16736 Oct 25 01:04  bonus
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai   3264 Oct 25 01:04  bonus.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai   3264 Oct 25 01:04  bonus.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai      0 Oct 24 23:06 'CUSTOM_VAR=H
elloTeX'
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai      0 Oct 24 23:06 'CUSTOM_VAR=H
elloTeX'
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16296 Oct 24 22:10  execl
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16280 Oct 24 22:57  execle_run-T
exStudio
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16296 Oct 24 22:10  execl
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai   1659 Oct 24 22:57  execle_run-T
exStudio.c
-rwxrwxrwx 1 shafeenyousafzai shafeenyousafzai  16280 Oct 24 22:57  execle_run-T
```

Figure 8: Mini-Shell Execution