Ex. No.: 5 **System Calls Programming**

Date:18.02.25

Aim:

To experiment system calls using fork(), execlp() and pid() functions.

Algorithm:

1. Start

o Include the required header files (stdio.h and stdlib.h).

2. Variable Declaration

o Declare an integer variable pid to hold the process ID.

3. Create a Process

o Call the fork() function to create a new process. Store the return value in the pid variable:

 If fork() returns:

 -1: Forking failed (child process not created).

 0: Process is the child process.

 Positive integer: Process is the parent process.

4. Print Statement Executed Twice

o Print the statement:

scss

Copy code

THIS LINE EXECUTED TWICE

(This line is executed by both parent and child processes after fork()).

5. Check for Process Creation Failure

o If pid == -1:

 Print:

Copy code

CHILD PROCESS NOT CREATED

 Exit the program using exit(0).

6. Child Process Execution

o If pid == 0 (child process):

 Print:

 Process ID of the child process using getpid().

 Parent process ID of the child process using getppid().

7. Parent Process Execution

o If pid > 0 (parent process):

 Print:

 Process ID of the parent process using getpid().

 Parent's parent process ID using getppid().

8. Final Print Statement

o Print the statement:

objectivec

Copy code

IT CAN BE EXECUTED TWICE

(This line is executed by both parent and child processes).

9. End

Program:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include<sys/wait.h>
int main() {
    int pid = fork();

    if (pid < 0) {
```

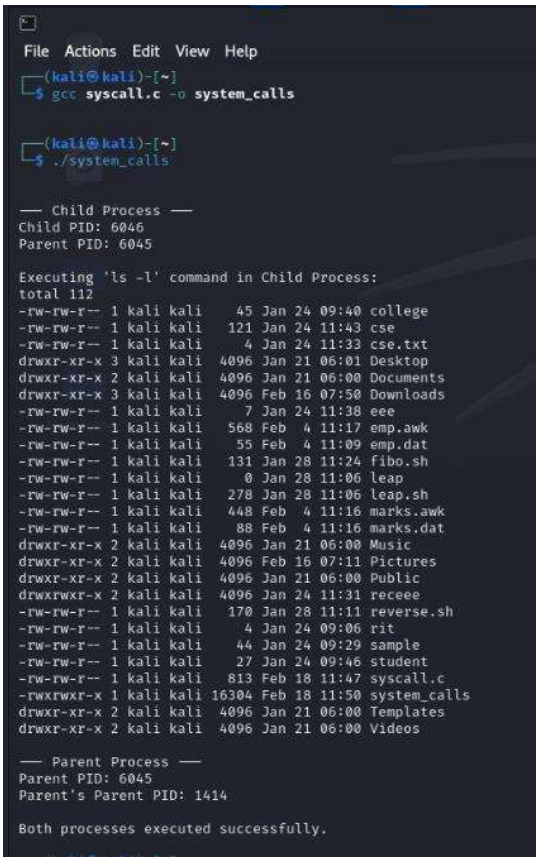
```
        printf("CHILD PROCESS NOT CREATED\n");
        exit(0);
    }
    else if (pid == 0) {
        printf("\n--- Child Process ---\n");
        printf("Child PID: %d\n", getpid());
        printf("Parent PID: %d\n", getppid());

        printf("\nExecuting 'ls -l' command in Child Process:\n");
        execlp("ls", "ls", "-l", NULL);

        printf("execlp() failed!\n");
        exit(1);
    }
    else {
        wait(NULL);
        printf("\n--- Parent Process ---\n");
        printf("Parent PID: %d\n", getpid());
        printf("Parent's Parent PID: %d\n", getppid());

        printf("\nBoth processes executed successfully.\n");
    }
    return 0;
}
```

Output:



Result:

Thus, the program using system calls was successfully executed.