**Sachin Chaudhary  
230701402**

**Ex No. 5:**

**System Calls Programming**

Aim:

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

Algorithm:

Start

Include the required header files: stdio.h and stdlib.h.

Variable Declaration

Declare an integer variable pid to hold the process ID.

Create a Process

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.

Print Statement Executed Twice

Print the statement:

THIS LINE EXECUTED TWICE

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

Check for Process Creation Failure

If pid == -1:

Print:

CHILD PROCESS NOT CREATED

Exit the program using exit(0).

Child Process Execution

If pid == 0 (child process):

Print:

Process ID of the child process using getpid().

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

Parent Process Execution

If pid > 0 (parent process):

Print:

Process ID of the parent process using getpid().

Parent's parent process ID using getppid().

Final Print Statement

Print the statement:

IT CAN BE EXECUTED TWICE

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

End

Program Code:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

int main() {

int pid;

// Create a process

pid = fork();

// Print this line for both parent and child processes

printf("THIS LINE EXECUTED TWICE\n");

// Check if fork() was successful

if (pid == -1) {

// Fork failed

printf("CHILD PROCESS NOT CREATED\n");

exit(0);

}

else if (pid == 0) {

// Child process

printf("Child Process ID: %d\n", getpid());

printf("Parent Process ID of Child: %d\n", getppid());

}

else {

// Parent process

printf("Parent Process ID: %d\n", getpid());

printf("Parent's Parent Process ID: %d\n", getppid());

}

// Final print statement for both processes

printf("IT CAN BE EXECUTED TWICE\n");

return 0;

}

Sample Output:

THIS LINE EXECUTED TWICE

Parent Process ID: 1234

Parent's Parent Process ID: 5678

IT CAN BE EXECUTED TWICE

THIS LINE EXECUTED TWICE

Child Process ID: 1235

Parent Process ID of Child: 1234

IT CAN BE EXECUTED TWICE

(Note: The process IDs (1234, 1235, 5678) will vary with each execution.)

Result:

The program demonstrates the creation of a child process using fork(). The fork() system call allows the parent and child processes to execute different parts of the program, with each process displaying its own process ID and the process ID of its parent. The output confirms that the program works correctly, executing the final print statement in both the parent and child processes.