

Ex. No.: 5

System Calls Programming

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

Algorithm:

1. **Start**
 - Include the required header files (stdio.h and stdlib.h).
2. **Variable Declaration**
 - Declare an integer variable pid to hold the process ID.
3. **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.
4. **Print Statement Executed Twice**
 - 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**
 - If pid == -1:
 - Print:

```
Copy code
CHILD PROCESS NOT CREATED
```
 - Exit the program using exit(0).
6. **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().
7. **Parent Process Execution**
 - 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**
 - 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>

int main ()

{

int pid;

pid=fork();

printf("\nTHIS LINE IS EXECUTED TWICE");

if (pid==-1)

{

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

exit(0);

}

if (pid==0)

{

printf("\n I AM CHILD PROCESS AND MY ID IS %d \n", getpid());

printf("\n I AM CHILD PARENT AND MY ID IS %d \n", getppid());

}

else

{

printf("\n I AM PARENT PROCESS AND MY ID IS %d \n", getpid());
```

```
printf("\n I AM PARENT PROCESS AND MY ID IS %d \n", getppid());  
  
}  
  
printf("\n IT CAN BE EXECUTED TWICE");  
  
printf("\n");
```

Output:

```
[student@localhost ~]$ ./a.out

THIS LINE EXECUTED TWICE
I AM PARENT PROCESS AND MY ID IS: 1724

THE PARENTS PARENT PROCESS ID IS: 1578

IT CAN BE EXECUTED TWICE

THIS LINE EXECUTED TWICE
I AM THE CHILD PROCESS AND MY ID IS 1725

THE CHILD PARENT PROCESS ID IS 1724

IT CAN BE EXECUTED TWICE
[student@localhost ~]$ █
```

Result:

The program was executed and got the output.