Experiment:1-Create a new process by invoking the appropriate system call. Get the process identifier of the currently running process and its respective parent using system calls and display the same using a C program.

Aim:

The aim of this C program is to create a new process using the fork() system call and then obtain and display the Process ID (PID) of both the parent process and the child process using system calls like getpid() and getppid().

Procedure:

1. **Fork a New Process**: The fork() system call is used to create a new child process. This system call creates a child process that is a copy of the parent.
2. **Get Process IDs**:
   - Use the getpid() system call to get the PID of the currently running process.
   - Use the getppid() system call to get the PID of the parent process.
3. **Display the PIDs**: Display the PID of the parent and child process to understand the relationship between the processes.

C Program:

```
#include <stdio.h>

#include <unistd.h> // for getpid(), getppid(), fork()


int main() {

  pid_t pid;


  // Create a new process using fork

  pid = fork();


  if (pid < 0) {

    // Fork failed

    printf("Fork failed\n");

    return 1;

  } else if (pid == 0) {

    // Child process

    printf("Child Process:\n");

    printf("PID of Child: %d\n", getpid());
```

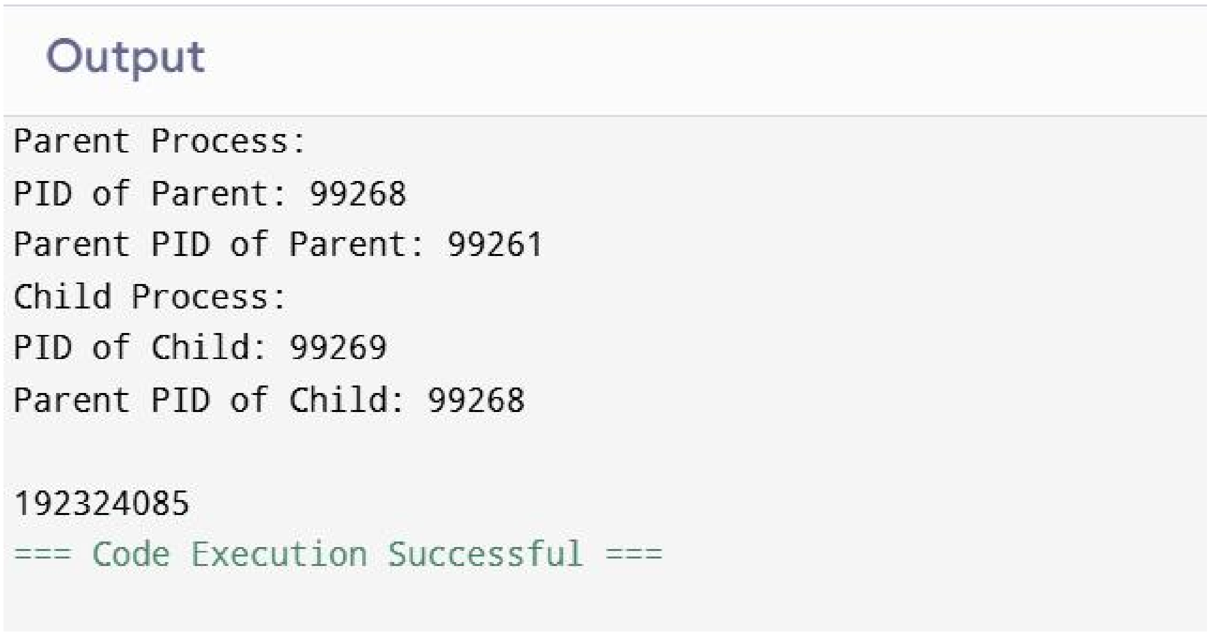
```
        printf("Parent PID of Child: %d\n", getppid());
    } else {
        // Parent process
        printf("Parent Process:\n");
        printf("PID of Parent: %d\n", getpid());
        printf("Parent PID of Parent: %d\n", getppid());
    }

    return 0;
}
```

Explanation:

1. **fork()**: This function creates a new process by duplicating the calling process. The new process (child) gets a PID different from the parent. If fork() returns 0, it indicates that the process is the child process. Otherwise, it returns the PID of the child to the parent.
2. **getpid()**: This system call returns the PID of the calling process.
3. **getppid()**: This system call returns the PID of the parent process of the calling process.

Output:

The output depends on whether the process is executed by the parent or the child. Below is an example of possible output when the program is executed.



```
Parent Process:
PID of Parent: 99268
Parent PID of Parent: 99261
Child Process:
PID of Child: 99269
Parent PID of Child: 99268

192324085
=== Code Execution Successful ===
```