Assignement -4

1. **Basic Process Creation Objective: Understanding process creation using fork() in C.**

**Task:**

**● Write a C program that creates a child process using fork().**

**● Print the process IDs (PID) of both parent and child.**

**● Use getpid() and getppid() to display process details.**

**Expected Output:**

**● Display parent and child PIDs.**

**● Identify whether execution happens in the child or parent process.**

#include <stdio.h>

#include <windows.h>

int main()

{

    STARTUPINFO si;

    PROCESS\_INFORMATION pi;

    ZeroMemory(&si, sizeof(si));

    si.cb = sizeof(si);

    ZeroMemory(&pi, sizeof(pi));

    if (!CreateProcess(

            NULL,

            "cmd /c echo Child Process Running",

            NULL,

            NULL,

            FALSE,

            0,

            NULL,

            NULL,

            &si,

            &pi

            )){

        printf("Process creation failed. Error: %d\n", GetLastError());

        return 1;

    }

    printf("Parent Process:\n");

    printf("PID: %lu\n", GetCurrentProcessId());

    printf("Child Process Created (PID: %lu)\n", pi.dwProcessId);

    WaitForSingleObject(pi.hProcess, INFINITE);

    CloseHandle(pi.hProcess);

    CloseHandle(pi.hThread);

    return 0;

}

**Output :**

=thread-group-added,id="i1"

GNU gdb (GDB) 7.6.1

Copyright (C) 2013 Free Software Foundation, Inc.

License GPLv3+: GNU GPL version 3 or later <http:*//gnu.org/licenses/gpl.html>*

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law.  Type "show copying"

and "show warranty" for details.

This GDB was configured as "mingw32".

For bug reporting instructions, please see:

<http:*//www.gnu.org/software/gdb/bugs/>.*

Warning: Debuggee TargetArchitecture not detected, assuming x86\_64.

=cmd-param-changed,param="pagination",value="off"

=cmd-param-changed,param="args",value="2>CON 1>CON <CON"

[New Thread 68308.0xcbec]

[New Thread 68308.0xddfc]

Loaded 'C:\Windows\SysWOW64\kernel32.dll'. Symbols loaded.

Loaded 'C:\Windows\SysWOW64\KernelBase.dll'. Symbols loaded.

Loaded 'C:\Windows\SysWOW64\apphelp.dll'. Symbols loaded.

Loaded 'C:\Windows\SysWOW64\msvcrt.dll'. Symbols loaded.

[New Thread 68308.0xa054]

Child Process Running

Parent Process:

PID: 68308

Child Process Created (PID: 62164)

The program 'D:\OScript\Assignment4\pr\_1.exe' has exited with code 0 (0x00000000).

[Running] cd "d:\OScript\Assignment4\" && gcc pr\_1.c -o pr\_1 && "d:\OScript\Assignment4\"pr\_1

Child Process Running

Parent Process:

PID: 24500

Child Process Created (PID: 49416)

[Done] exited with code=0 in 3.093 seconds

1. **Process Synchronization Using wait() Objective: Understanding how a parent waits for a child process to finish execution.**

**Task:**

**● Modify the previous program to make the parent wait for the child to complete execution using wait() .**

**● Print messages before and after the child process exits.**

**Expected Output:**

**● The parent should wait for the child to finish before printing its final message.**

#include <stdio.h>

#include <windows.h>

int main()

{

    STARTUPINFO si;

    PROCESS\_INFORMATION pi;

    ZeroMemory(&si, sizeof(si));

    si.cb = sizeof(si);

    ZeroMemory(&pi, sizeof(pi));

    printf("Parent Process (PID: %lu) is creating a child process...\n", GetCurrentProcessId());

    if (!CreateProcess(

            NULL,

            "cmd /c echo Child Process Running",

            NULL,

            NULL,

            FALSE,

            0,

            NULL,

            NULL,

            &si,

            &pi

        )) {

        printf("Process creation failed. Error: %d\n", GetLastError());

        return 1;

    }

    printf("Child Process Created (PID: %lu)\n", pi.dwProcessId);

    printf("Parent Process is waiting for the child to complete...\n");

    WaitForSingleObject(pi.hProcess, INFINITE);

    printf("Child Process has finished execution.\n");

    printf("Parent Process (PID: %lu) resumes execution.\n", GetCurrentProcessId());

    CloseHandle(pi.hProcess);

    CloseHandle(pi.hThread);

    return 0;

}

**Output :**

PS D:\OScript>  & 'c:\Users\Khushi\.vscode\extensions\ms-vscode.cpptools-1.23.6-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-k35pw3y1.kcb' '--stdout=Microsoft-MIEngine-Out-w5kj5mew.nfu' '--stderr=Microsoft-MIEngine-Error-hf1qobia.1lw' '--pid=Microsoft-MIEngine-Pid-reqtjsqk.qp0' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

[Running] cd "d:\OScript\Assignment4\" && gcc pr\_2.c -o pr\_2 && "d:\OScript\Assignment4\"pr\_2

Child Process Running

Parent Process (PID: 61520) is creating a child process...

Child Process Created (PID: 14316)

Parent Process is waiting for the child to complete...

Child Process has finished execution.

Parent Process (PID: 61520) resumes execution.

[Done] exited with code=0 in 3.23 seconds

1. **Process Execution using exec()**

**Objective:**

**● Executing a new program within a child process.**

**Task:**

**● Create a C program where the child process replaces itself with another program (e.g., /bin/ls) using execlp().**

**● The parent process waits for the child to complete execution.**

**Expected Output:**

**● The child process successfully replaces itself with the ls command output.**