ACTIVITY NO. 1

Process Creation and Management in Operating Systems using C++	
Course Code: CPE022A	**Program:** Computer Engineering
Course Title: Operating Systems	**Date Performed:08/05/2025**
Section: CPE31S4	**Date Submitted:08/05/2025**
Name: Bautista, Jhon Hendricks T.	**Instructor: Mrs. Maria Rizette Sayo**

1. Objective(s)

- Learn how operating systems create and manage processes.
- Compare different process creation methods fork() in Windows.
- Implement basic process control operations.

2. Intended Learning Outcomes (ILOs)

After this activity, the student should be able to:

- Write C++ programs that create and manage processes.
- Explain how fork() and CreateProcess() work.
- Demonstrate proper process synchronization and cleanup

3. Discussion

Key Concepts:

This lab activity focuses exclusively on Windows process management using the Windows API, highlighting:

1. Process creation via CreateProcess()
2. Parent-child process relationships
3. Process synchronization with WaitForSingleObject()
4. Proper handle management with CloseHandle()

Technical Implementation:

1. Process Creation

- CreateProcess() is the fundamental Windows API for spawning new processes
- Unlike Unix's fork()/exec() model, Windows creates new processes from scratch
- Requires explicit specification of:
  - Executable path/command line
  - Security attributes
  - Environment and working directory
  - Startup configuration

2. Process Synchronization

- WaitForSingleObject() blocks the parent until child process termination
- More flexible than Unix wait() as it can:
    - Specify timeout periods
    - Wait on various object types (not just processes)
    - Be used with other synchronization mechanisms

3. Resource Management

- Strict handle management is required:
    - Process handles
    - Thread handles
- Failure to close handles leads to resource leaks
- Windows maintains reference counts for handles

4. Materials and Equipment

Personal Computer with C++ IDE
Recommended IDE:

- CLion (must use TIP email to download)
- DevC++ (use the embarcadero fork or configure to C++17)

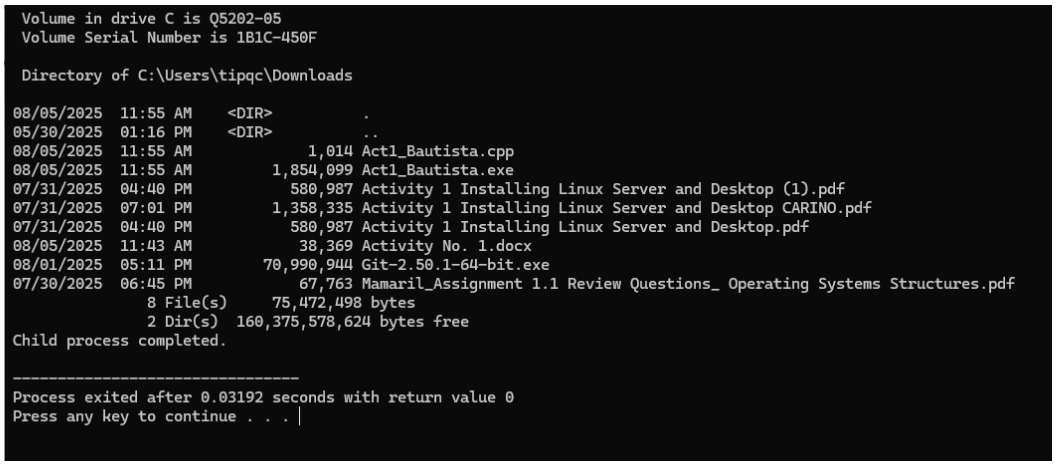
5. Procedure

Setting Up the Development Environment

1. Open Visual Studio (or any C++ IDE with Windows SDK)
2. Create a new C++ Console Application project
3. Ensure Windows.h header is available
4. Copy the source codes in Dev C++ or any IDE and save it to your Gdrive as Activity-1.cpp

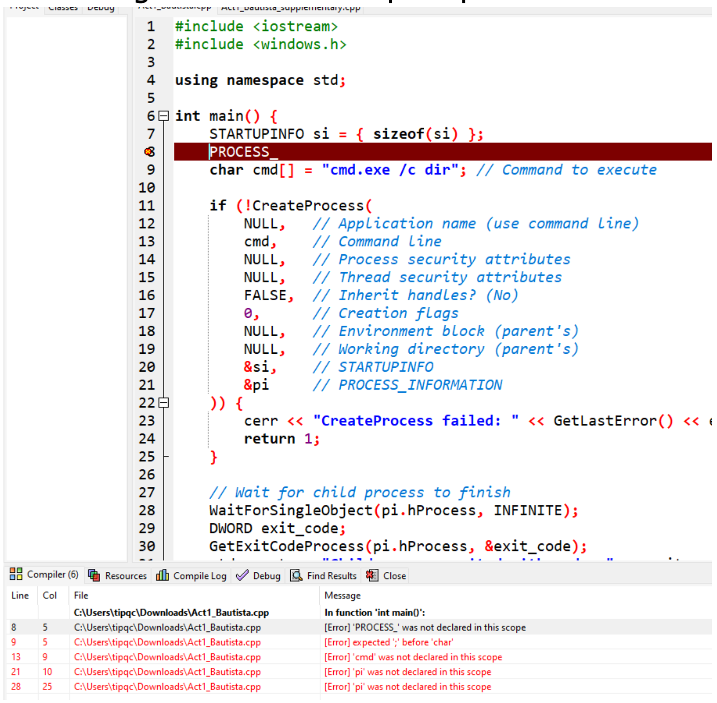
```
#include <iostream>
#include <windows.h>

using namespace std;

int main() {
    STARTUPINFO si = { sizeof(si) };
    PROCESS_INFORMATION pi;
    char cmd[] = "cmd.exe /c dir"; // Command to execute

    if (!CreateProcess(
        NULL,   // Application name (use command line)
        cmd,    // Command line
        NULL,   // Process security attributes
        NULL,   // Thread security attributes
        FALSE,  // Inherit handles? (No)
        0,      // Creation flags
        NULL,   // Environment block (parent's)
        NULL,   // Working directory (parent's)
        &si,    // STARTUPINFO
        &pi     // PROCESS_INFORMATION
    )) {
        cerr << "CreateProcess failed: " << GetLastError() << endl;
        return 1;
    }

    // Wait for child process to finish
    WaitForSingleObject(pi.hProcess, INFINITE);

    // Clean up
    CloseHandle(pi.hProcess);
    CloseHandle(pi.hThread);

    cout << "Child process completed." << endl;
    return 0;
}
```

6. Output



```
 Volume in drive C is Q5202-05
 Volume Serial Number is 1B1C-450F

 Directory of C:\Users\tipqc\Downloads

08/05/2025  11:55 AM    <DIR>          .
05/30/2025  01:16 PM    <DIR>          ..
08/05/2025  11:55 AM             1,014 Act1_Bautista.cpp
08/05/2025  11:55 AM         1,854,099 Act1_Bautista.exe
07/31/2025  04:40 PM           580,987 Activity 1 Installing Linux Server and Desktop (1).pdf
07/31/2025  07:01 PM         1,358,335 Activity 1 Installing Linux Server and Desktop CARINO.pdf
07/31/2025  04:40 PM           580,987 Activity 1 Installing Linux Server and Desktop.pdf
08/05/2025  11:43 AM            38,369 Activity No. 1.docx
08/01/2025  05:11 PM        70,990,944 Git-2.50.1-64-bit.exe
07/30/2025  06:45 PM            67,763 Mamaril_Assignment 1.1 Review Questions_ Operating Systems Structures.pdf
               8 File(s)     75,472,498 bytes
               2 Dir(s)  160,375,578,624 bytes free
Child process completed.

--------------------------------
Process exited after 0.03192 seconds with return value 0
Press any key to continue . . .
```

Figure1. Command prompt window



```
1   #include <iostream>
2   #include <windows.h>
3
4   using namespace std;
5
6   int main() {
7       STARTUPINFO si = { sizeof(si) };
8       PROCESS_
9       char cmd[] = "cmd.exe /c dir"; // Command to execute
10
11      if (!CreateProcess(
12          NULL,   // Application name (use command line)
13          cmd,    // Command line
14          NULL,   // Process security attributes
15          NULL,   // Thread security attributes
16          FALSE,  // Inherit handles? (No)
17          0,      // Creation flags
18          NULL,   // Environment block (parent's)
19          NULL,   // Working directory (parent's)
20          &si,    // STARTUPINFO
21          &pi     // PROCESS_INFORMATION
22      )) {
23          cerr << "CreateProcess failed: " << GetLastError() <<
24          return 1;
25      }
26
27      // Wait for child process to finish
28      WaitForSingleObject(pi.hProcess, INFINITE);
29      DWORD exit_code;
30      GetExitCodeProcess(pi.hProcess, &exit_code);
```

Line	Col	File	Message
		C:\Users\tipqc\Downloads\Act1_Bautista.cpp	In function 'int main()':
8	5	C:\Users\tipqc\Downloads\Act1_Bautista.cpp	[Error] 'PROCESS_' was not declared in this scope
9	5	C:\Users\tipqc\Downloads\Act1_Bautista.cpp	[Error] expected ';' before 'char'
13	9	C:\Users\tipqc\Downloads\Act1_Bautista.cpp	[Error] 'cmd' was not declared in this scope
21	10	C:\Users\tipqc\Downloads\Act1_Bautista.cpp	[Error] 'pi' was not declared in this scope
28	25	C:\Users\tipqc\Downloads\Act1_Bautista.cpp	[Error] 'pi' was not declared in this scope

Figure 2. Console Showing if there's an error

7. Observation

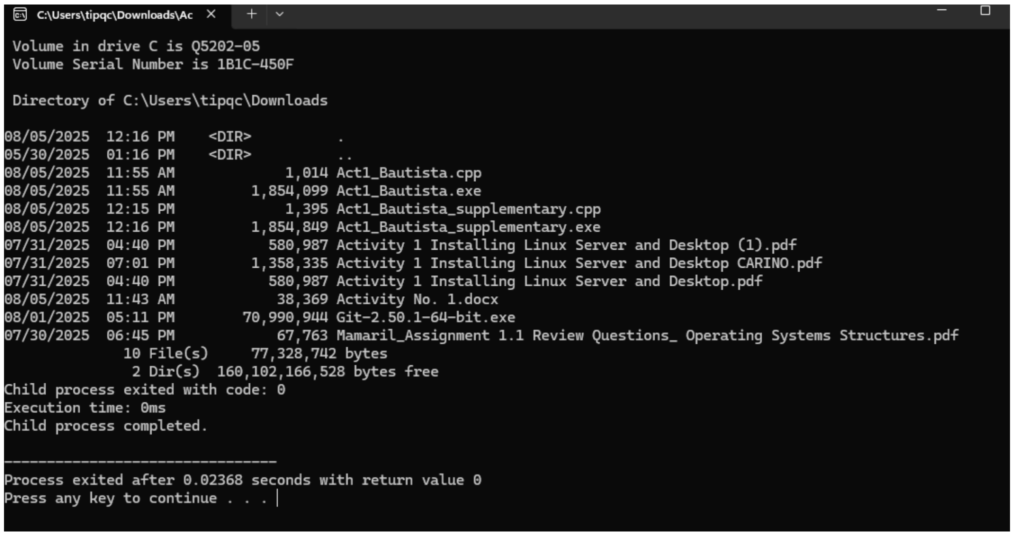
Based on the output of the program, it is mainly about accessing a log of the directory being used by the program. This is possible due to the use of a windows command which accesses the command line to output the child process. The process is the displaying of the directory through the command line which is dictated by "cmd.exe /c dir". If ever the program fails to create the process, it will output "CreateProcess failed:".

8. Supplementary Activity

1. Copy the source codes above and add these snippets of codes after WaitForSingleObject()

```
// Add after WaitForSingleObject
DWORD exit_code;
GetExitCodeProcess(pi.hProcess, &exit_code);
std::cout << "Child process exited with code: " << exit_code << std::endl;

// Execution time measurement
DWORD start_time = GetTickCount();
WaitForSingleObject(pi.hProcess, INFINITE);
DWORD duration = GetTickCount() - start_time;
std::cout << "Execution time: " << duration << "ms" << std::endl;
```

2. Save your program as Activity-1-supplementary.cpp in your Gdrive.
3. What did you observe?



```
C:\Users\tipqc\Downloads\Ac

 Volume in drive C is Q5202-05
 Volume Serial Number is 1B1C-450F

 Directory of C:\Users\tipqc\Downloads

08/05/2025  12:16 PM    <DIR>          .
05/30/2025  01:16 PM    <DIR>          ..
08/05/2025  11:55 AM             1,014 Act1_Bautista.cpp
08/05/2025  11:55 AM         1,854,099 Act1_Bautista.exe
08/05/2025  12:15 PM             1,395 Act1_Bautista_supplementary.cpp
08/05/2025  12:16 PM         1,854,849 Act1_Bautista_supplementary.exe
07/31/2025  04:40 PM           580,987 Activity 1 Installing Linux Server and Desktop (1).pdf
07/31/2025  07:01 PM         1,358,335 Activity 1 Installing Linux Server and Desktop CARINO.pdf
07/31/2025  04:40 PM           580,987 Activity 1 Installing Linux Server and Desktop.pdf
08/05/2025  11:43 AM            38,369 Activity No. 1.docx
08/01/2025  05:11 PM        70,990,944 Git-2.50.1-64-bit.exe
07/30/2025  06:45 PM            67,763 Mamaril_Assignment 1.1 Review Questions_ Operating Systems Structures.pdf
              10 File(s)     77,328,742 bytes
               2 Dir(s)  160,102,166,528 bytes free
Child process exited with code: 0
Execution time: 0ms
Child process completed.

--------------------------------
Process exited after 0.02368 seconds with return value 0
Press any key to continue . . .
```

Figure 3. Revised Program

After running the program with the added snippet of code there are additional information on the output command line. After properly accessing the files in the directory, before clearing the handles the program outputs the exited code and the execution time. The execution time was extracted using the GetTickCount function which is a windows function for elapsed time in running a program. Then for the extraction of the exit code, this was possible by using the GetExitCodeProcess being added in the code.

8. Conclusion
After accomplishing the activity I was able to learn that in C++ we are able to implement a Windows API in order to create Windows Processes. Based on the procedure there are three steps for running a windows process. First is the Process creation where we define different specifications. Second is the Process Synchronization for proper timeout periods. Last is the resource management which is about closing handles. In the supplementary I was able to perform these steps by creating a C++ program that was able to create a process for accessing and displaying files in a directory. For me, I had a challenge in understanding the program first since it is my first time programming for windows processes. I think I need to learn more about the different functions involved with Windows and their respective outputs in the Command Line.
9. Assessment Rubric