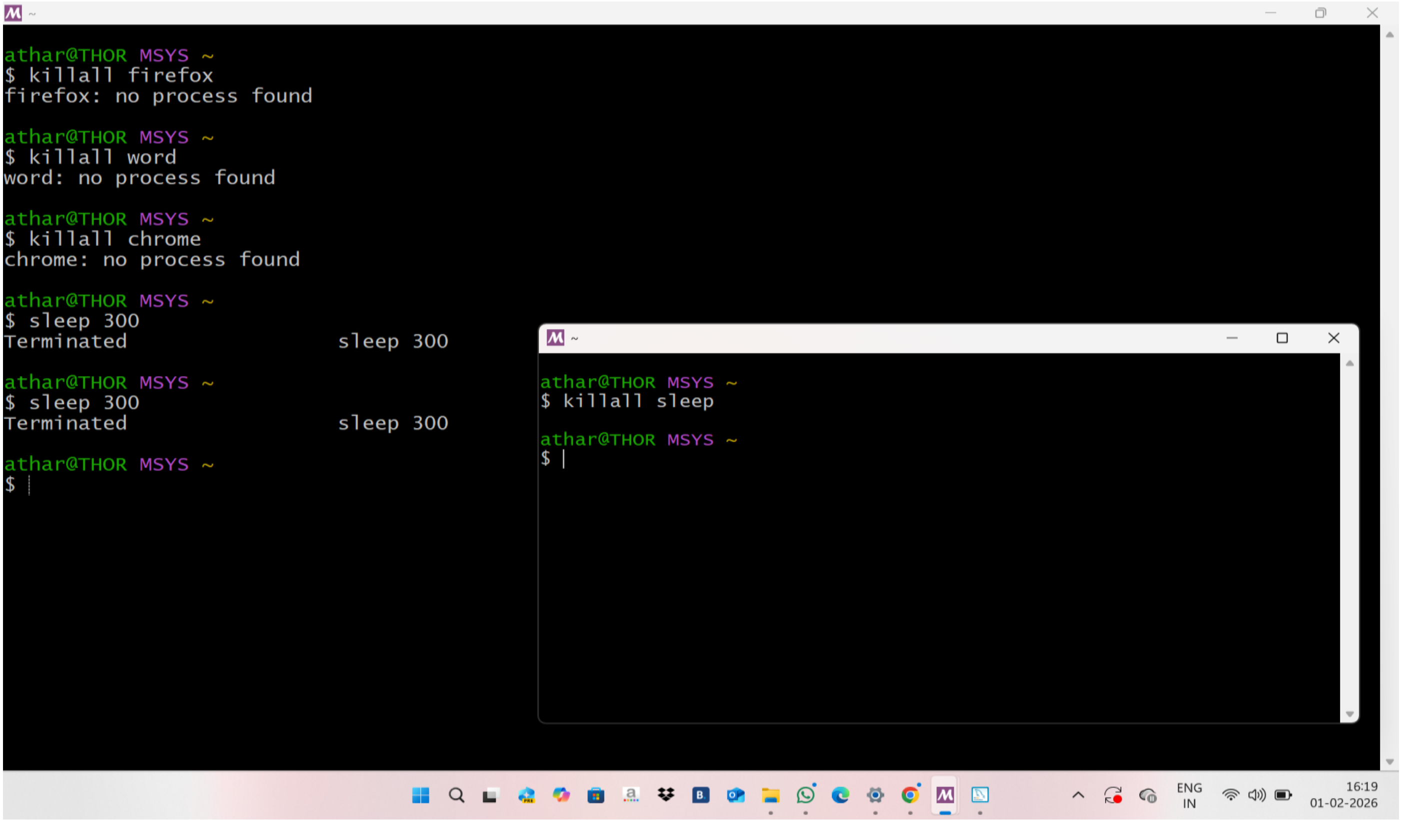
1. LINUX operating system. Which commands will he use to complete the given task with the help of the following operation?

●Kill processes by name

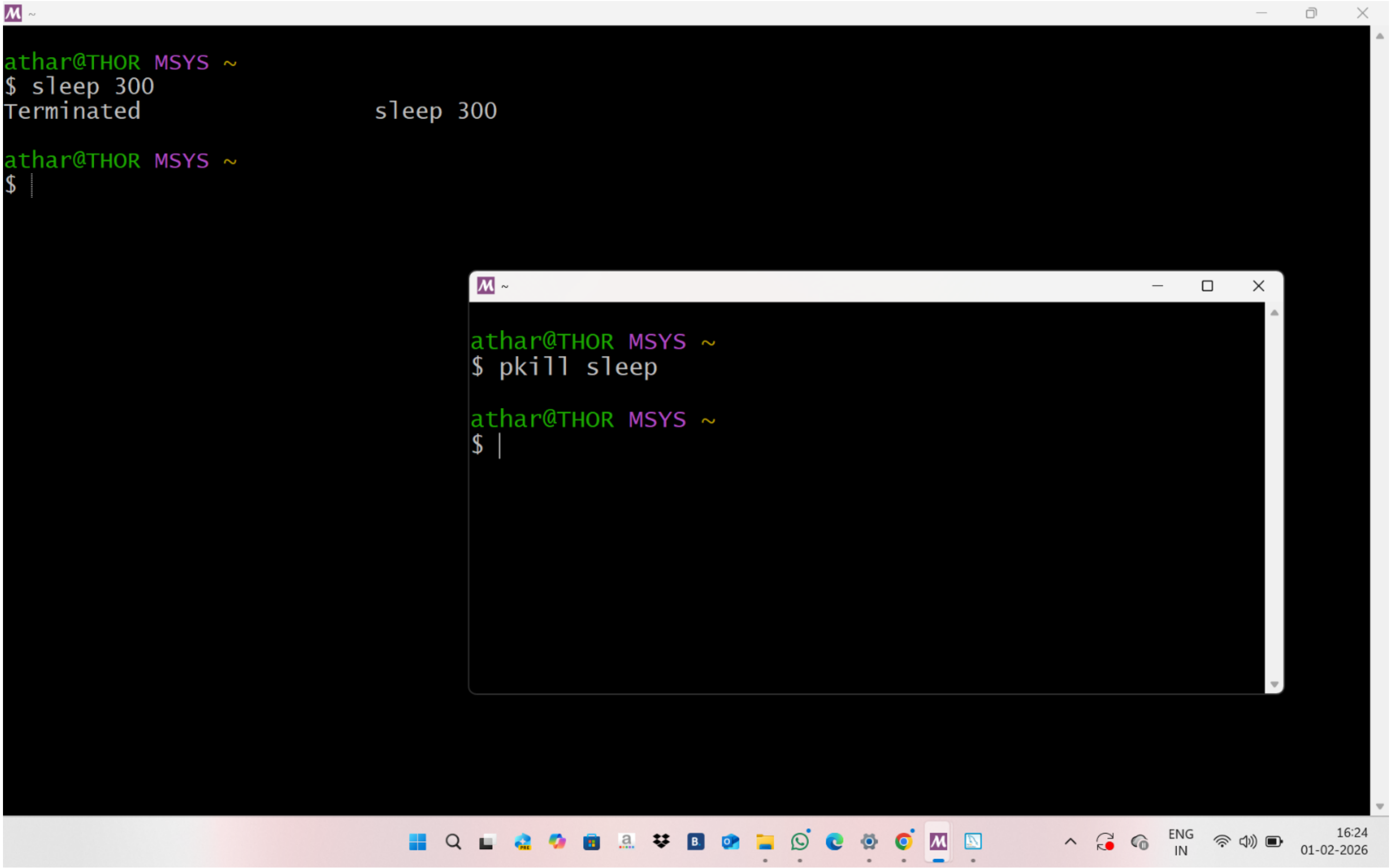
●Kill a process based on the process name

●Kill a single process at a time with the given process ID



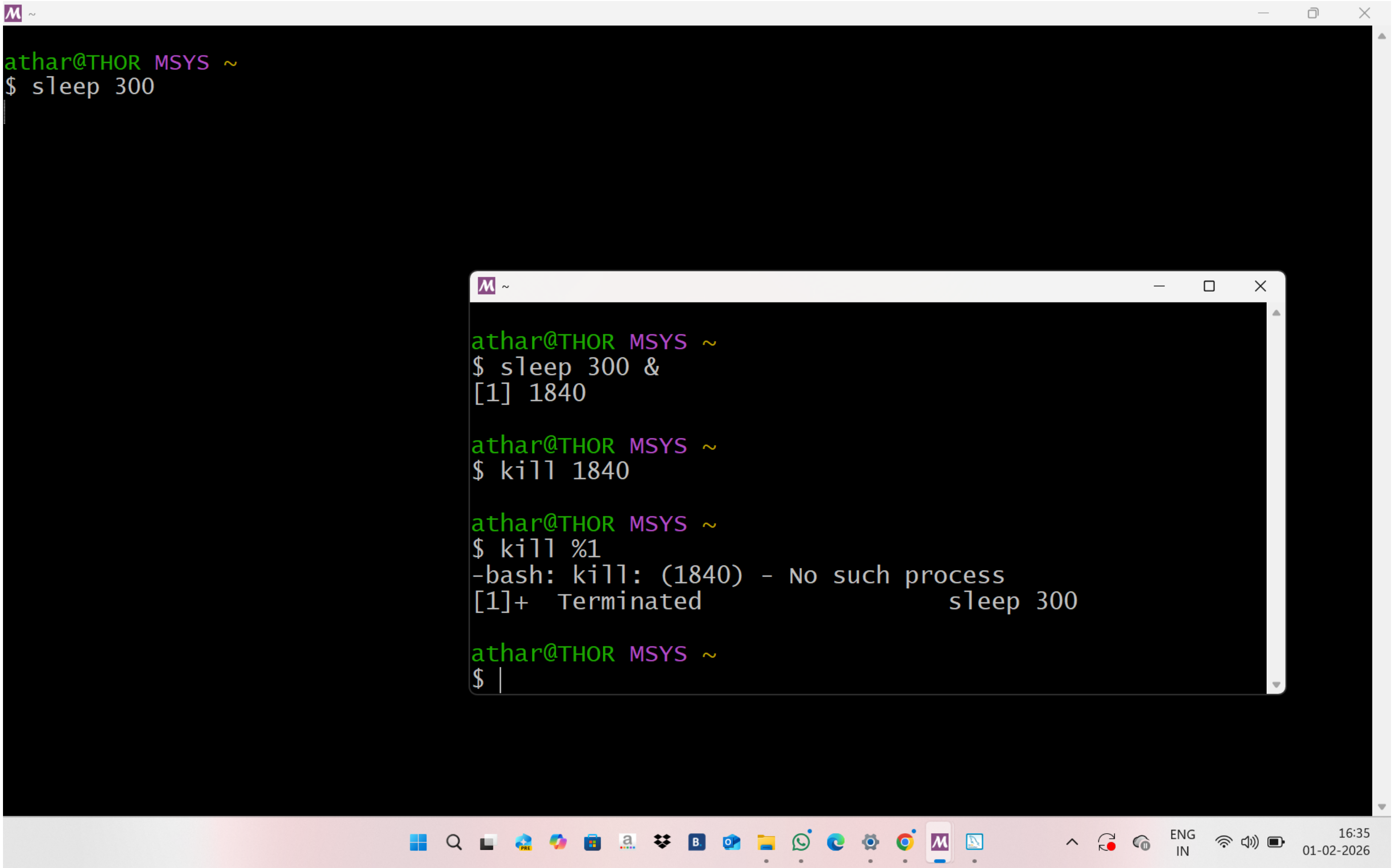
```
athar@THOR MSYS ~
$ killall firefox
firefox: no process found

athar@THOR MSYS ~
$ killall word
word: no process found

athar@THOR MSYS ~
$ killall chrome
chrome: no process found

athar@THOR MSYS ~
$ sleep 300
Terminated                    sleep 300

athar@THOR MSYS ~
$ sleep 300
Terminated                    sleep 300

athar@THOR MSYS ~
$
```

```
athar@THOR MSYS ~
$ killall sleep

athar@THOR MSYS ~
$ |
```



```
athar@THOR MSYS ~
$ sleep 300
Terminated                    sleep 300

athar@THOR MSYS ~
$
```

```
athar@THOR MSYS ~
$ pkill sleep

athar@THOR MSYS ~
$ |
```



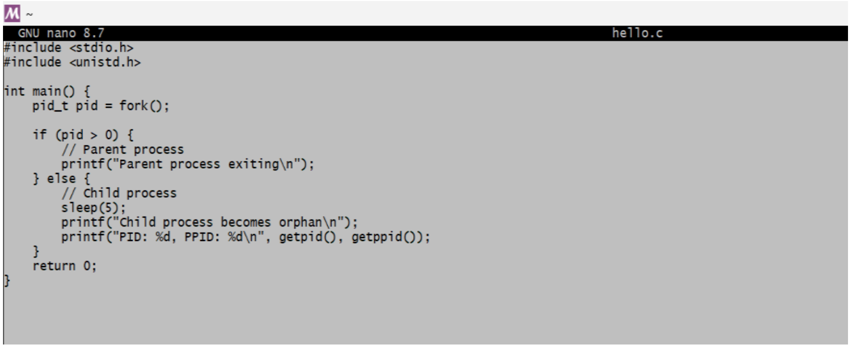
```
athar@THOR MSYS ~
$ sleep 300
```

```
athar@THOR MSYS ~
$ sleep 300 &
[1] 1840

athar@THOR MSYS ~
$ kill 1840

athar@THOR MSYS ~
$ kill %1
-bash: kill: (1840) - No such process
[1]+  Terminated              sleep 300

athar@THOR MSYS ~
$ 
```

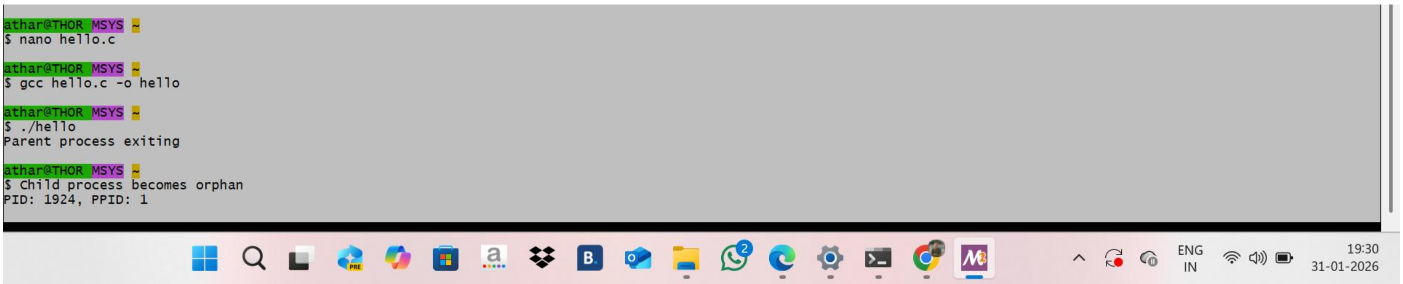
2 .Write a program for process creation using C

- Orphan Process
- Zombie Process

orphan process

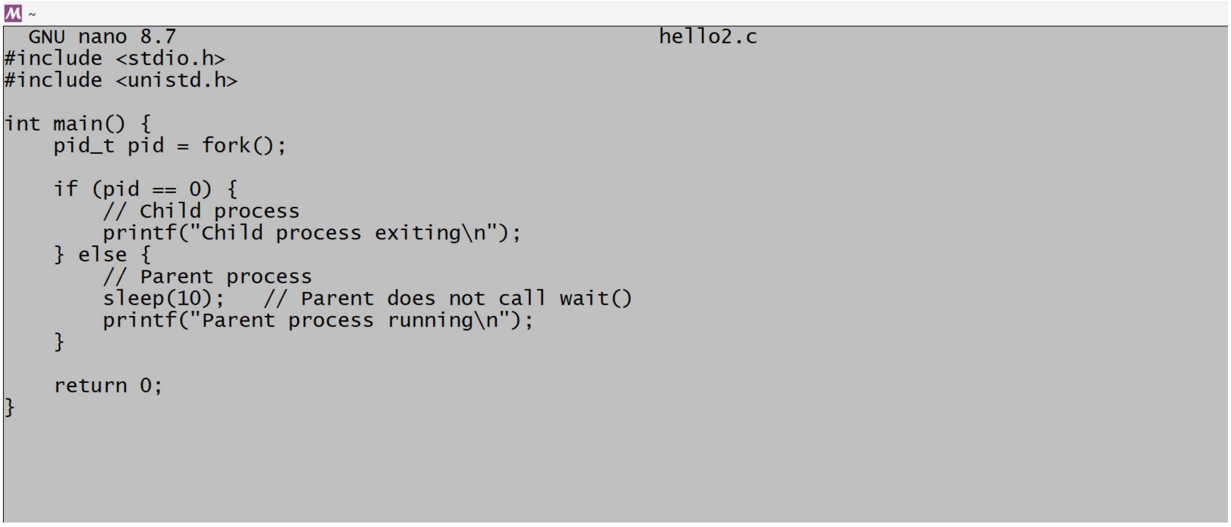


```c
#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid = fork();

    if (pid > 0) {
        // Parent process
        printf("Parent process exiting\n");
    } else {
        // Child process
        sleep(5);
        printf("Child process becomes orphan\n");
        printf("PID: %d, PPID: %d\n", getpid(), getppid());
    }
    return 0;
}
```

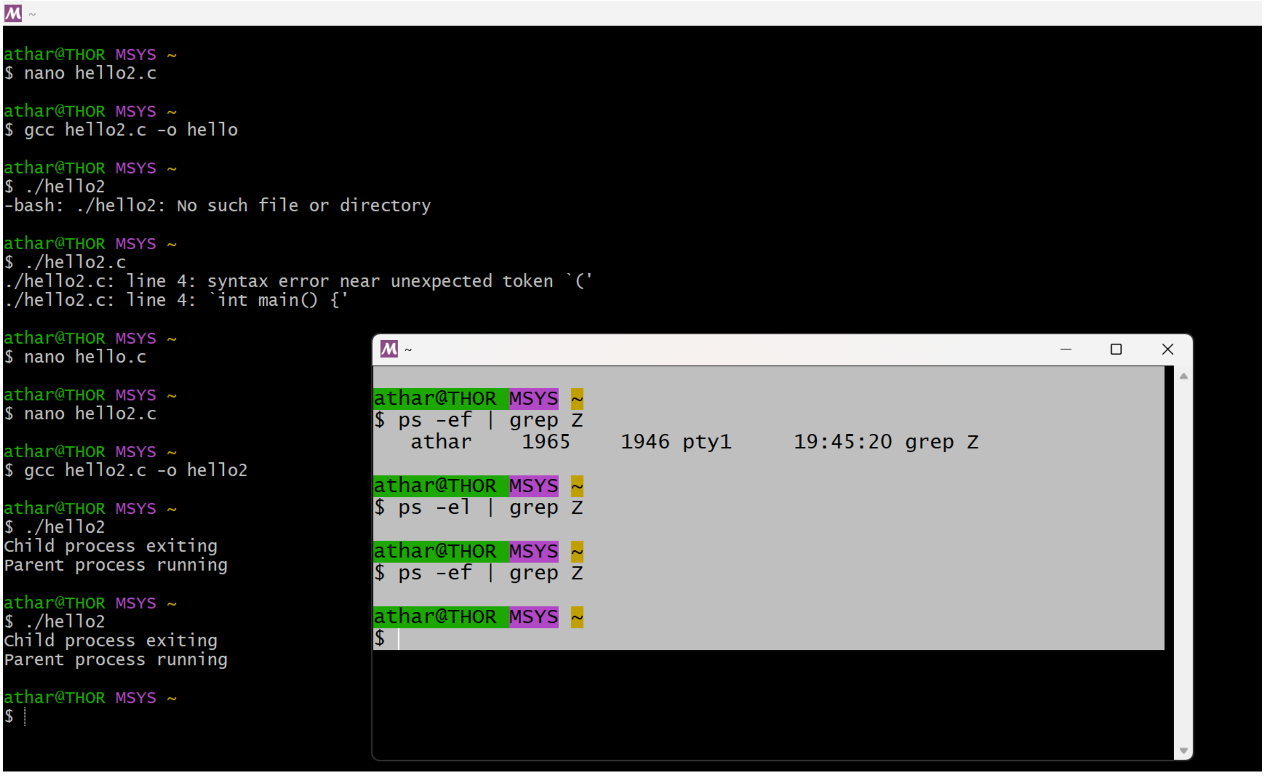


```
athar@THOR MSYS ~
$ nano hello.c

athar@THOR MSYS ~
$ gcc hello.c -o hello

athar@THOR MSYS ~
$ ./hello
Parent process exiting

athar@THOR MSYS ~
$ Child process becomes orphan
PID: 1924, PPID: 1
```

zombie Process



```
  GNU nano 8.7                                  hello2.c
#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid = fork();

    if (pid == 0) {
        // Child process
        printf("Child process exiting\n");
    } else {
        // Parent process
        sleep(10);    // Parent does not call wait()
        printf("Parent process running\n");
    }

    return 0;
}
```

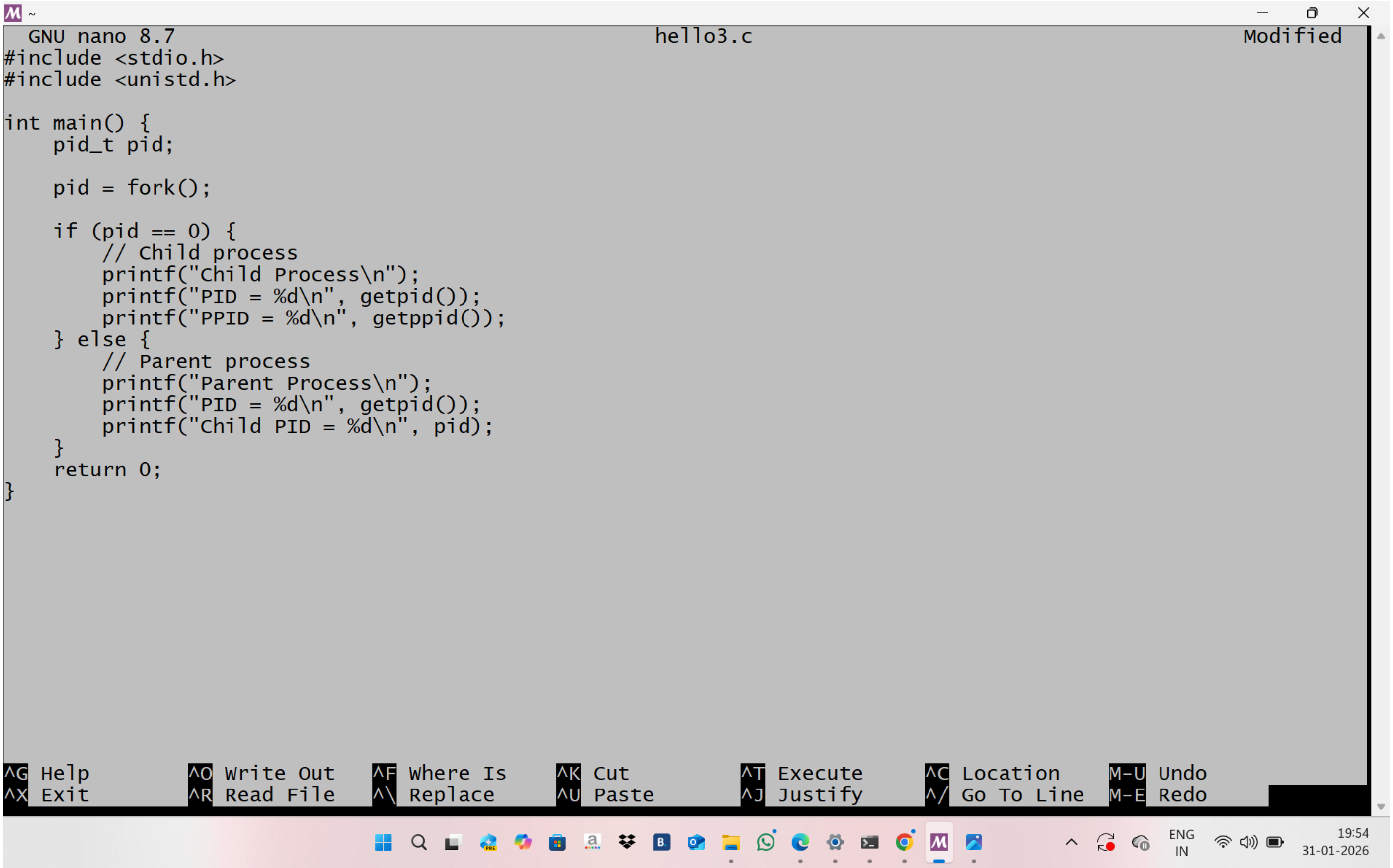


```
athar@THOR MSYS ~
$ nano hello2.c

athar@THOR MSYS ~
$ gcc hello2.c -o hello

athar@THOR MSYS ~
$ ./hello2
-bash: ./hello2: No such file or directory

athar@THOR MSYS ~
$ ./hello2.c
./hello2.c: line 4: syntax error near unexpected token `('
./hello2.c: line 4: `int main() {'

athar@THOR MSYS ~
$ nano hello.c

athar@THOR MSYS ~
$ nano hello2.c

athar@THOR MSYS ~
$ gcc hello2.c -o hello2

athar@THOR MSYS ~
$ ./hello2
Child process exiting
Parent process running

athar@THOR MSYS ~
$ ./hello2
Child process exiting
Parent process running

athar@THOR MSYS ~
$ |
```

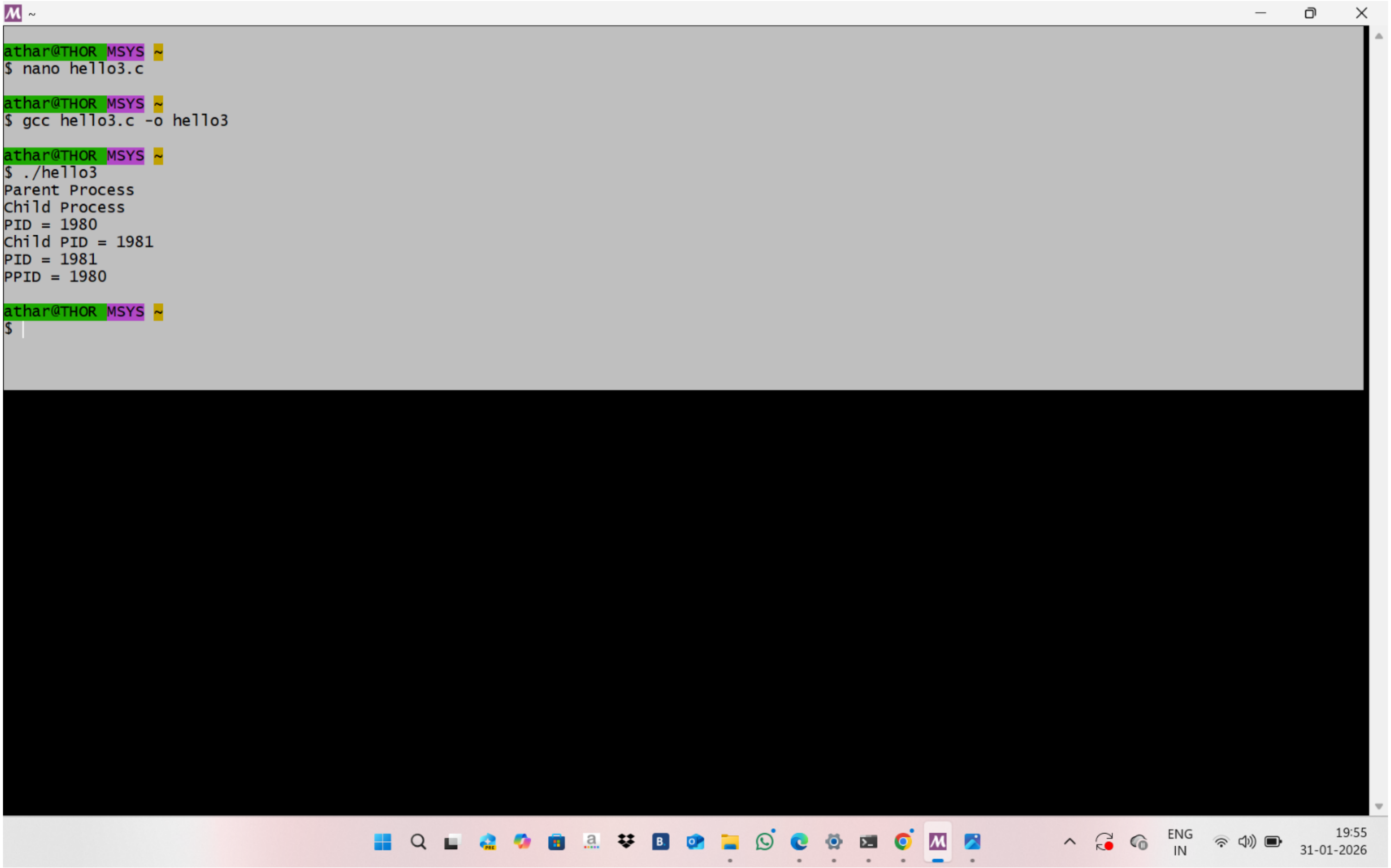
```
athar@THOR MSYS ~
$ ps -ef | grep Z
   athar     1965     1946 pty1      19:45:20 grep Z

athar@THOR MSYS ~
$ ps -el | grep Z

athar@THOR MSYS ~
$ ps -ef | grep Z

athar@THOR MSYS ~
$ 
```

3. Create the process using fork () system call.

- **Child Process creation**
- **Parent process creation**
- **PPID and PID**



```
  GNU nano 8.7                        hello3.c                         Modified
#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid;

    pid = fork();

    if (pid == 0) {
        // Child process
        printf("Child Process\n");
        printf("PID = %d\n", getpid());
        printf("PPID = %d\n", getppid());
    } else {
        // Parent process
        printf("Parent Process\n");
        printf("PID = %d\n", getpid());
        printf("Child PID = %d\n", pid);
    }
    return 0;
}

^G Help         ^O Write Out    ^F Where Is     ^K Cut          ^T Execute      ^C Location     M-U Undo
^X Exit         ^R Read File    ^\ Replace      ^U Paste        ^J Justify      ^/ Go To Line   M-E Redo
```



```
athar@THOR MSYS ~
$ nano hello3.c

athar@THOR MSYS ~
$ gcc hello3.c -o hello3

athar@THOR MSYS ~
$ ./hello3
Parent Process
Child Process
PID = 1980
Child PID = 1981
PID = 1981
PPID = 1980

athar@THOR MSYS ~
$ 
```