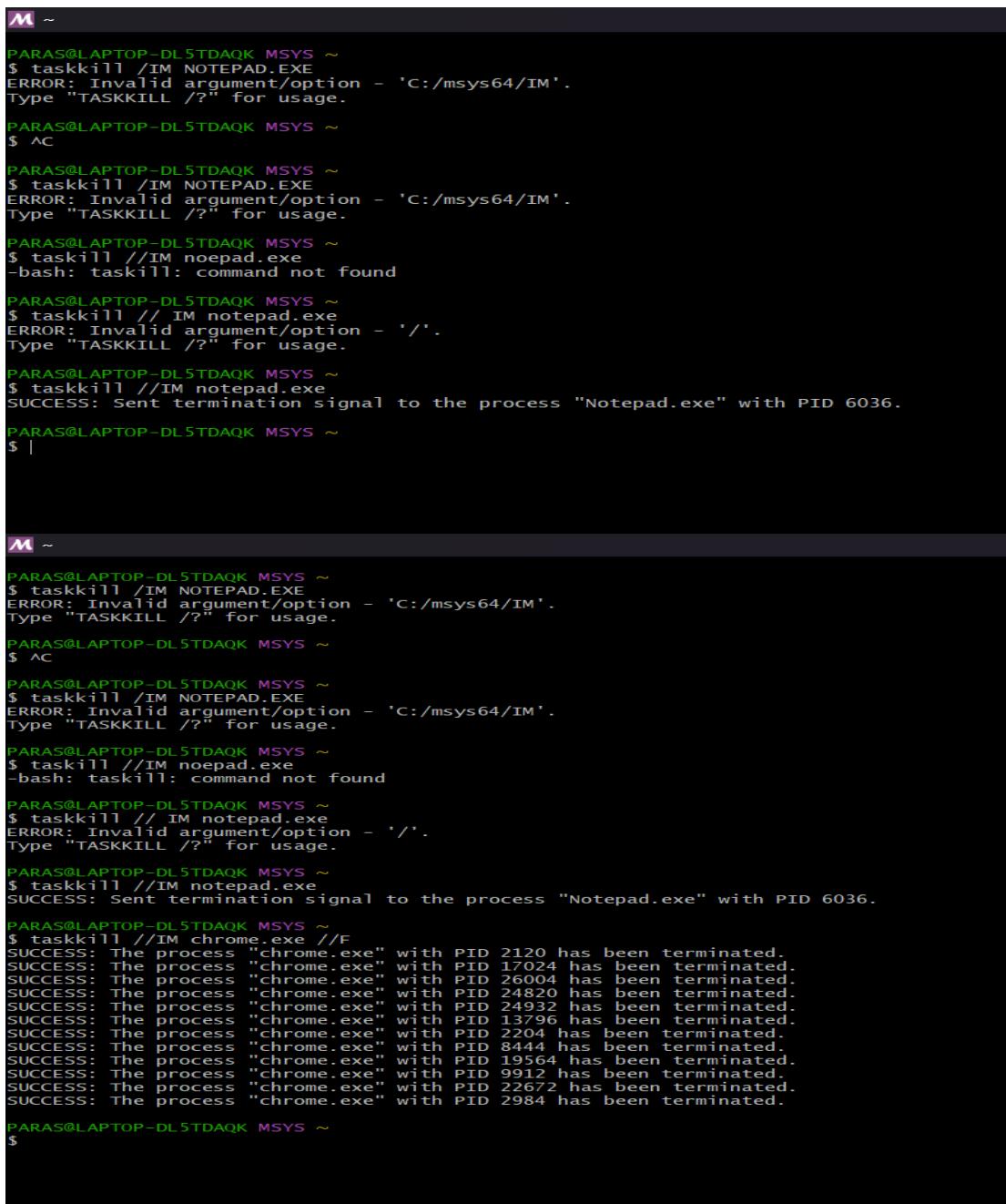


4 [Process management Based practical]

1. Adam is working in an IT company. He has been given a task to reduce the load of a system by killing some of the processes running in the 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



```
M ~
PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill /IM NOTEPAD.EXE
ERROR: Invalid argument/option - 'C:/msys64/IM'.
Type "TASKKILL /?" for usage.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ ^C

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill /IM NOTEPAD.EXE
ERROR: Invalid argument/option - 'C:/msys64/IM'.
Type "TASKKILL /?" for usage.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill //IM noepad.exe
-bash: taskkill: command not found

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill // IM notepad.exe
ERROR: Invalid argument/option - '/'.
Type "TASKKILL /?" for usage.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill //IM notepad.exe
SUCCESS: Sent termination signal to the process "Notepad.exe" with PID 6036.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ |
```



```
M ~
PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill /IM NOTEPAD.EXE
ERROR: Invalid argument/option - 'C:/msys64/IM'.
Type "TASKKILL /?" for usage.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ ^C

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill /IM NOTEPAD.EXE
ERROR: Invalid argument/option - 'C:/msys64/IM'.
Type "TASKKILL /?" for usage.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill //IM noepad.exe
-bash: taskkill: command not found

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill // IM notepad.exe
ERROR: Invalid argument/option - '/'.
Type "TASKKILL /?" for usage.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill //IM notepad.exe
SUCCESS: Sent termination signal to the process "Notepad.exe" with PID 6036.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ taskkill //IM chrome.exe //F
SUCCESS: The process "chrome.exe" with PID 2120 has been terminated.
SUCCESS: The process "chrome.exe" with PID 17024 has been terminated.
SUCCESS: The process "chrome.exe" with PID 26004 has been terminated.
SUCCESS: The process "chrome.exe" with PID 24820 has been terminated.
SUCCESS: The process "chrome.exe" with PID 24932 has been terminated.
SUCCESS: The process "chrome.exe" with PID 13796 has been terminated.
SUCCESS: The process "chrome.exe" with PID 2204 has been terminated.
SUCCESS: The process "chrome.exe" with PID 8444 has been terminated.
SUCCESS: The process "chrome.exe" with PID 19564 has been terminated.
SUCCESS: The process "chrome.exe" with PID 9912 has been terminated.
SUCCESS: The process "chrome.exe" with PID 22672 has been terminated.
SUCCESS: The process "chrome.exe" with PID 2984 has been terminated.

PARAS@LAPTOP-DL5TDAQK MSYS ~
$
```

2. Write a program for process creation using C

- Orphan Process

```
M ~
GNU nano 8.7
#include <stdio.h>
#include <unistd.h>

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

    if (pid > 0) {
        printf("Parent process PID: %d\n", getpid());
        sleep(2);
        printf("Parent exiting...\n");
    } else if (pid == 0) {
        sleep(5);
        printf("Child process PID: %d\n", getpid());
        printf("New Parent PID (init): %d\n", getpid());
    } else {
        printf("Fork failed\n");
    }

    return 0;
}
```

```
PARAS@LAPTOP-DL5TDAQK MSYS ~
$ gcc orphan.c -o orphan
./orphan
Parent process PID: 2246
Parent exiting...
```

```
PARAS@LAPTOP-DL5TDAQK MSYS ~
$ Child process PID: 2247
New Parent PID (init): 2246
```

- Zombie Process

```
M ~
GNU nano 8.7
#include <stdio.h>
#include <unistd.h>

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

    if (pid > 0) {
        printf("Parent PID: %d\n", getpid());
        sleep(10);
        printf("Parent exiting...\n");
    } else if (pid == 0) {
        printf("Child PID: %d\n", getpid());
        printf("Child exiting...\n");
    } else {
        printf("Fork failed\n");
    }

    return 0;
}
```

```
PARAS@LAPTOP-DL5TDAQK MSYS ~
$ nano zombie.c
```

```
PARAS@LAPTOP-DL5TDAQK MSYS ~
$ gcc zombie.c -o zombie
```

```
PARAS@LAPTOP-DL5TDAQK MSYS ~
$ ./zombie
Parent PID: 2257
Child PID: 2258
Child exiting...
Parent exiting...
```

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

- Child Process creation

- Parent process creation
- PPID and PID

```
M ~
GNU nano 8.7
#include <stdio.h>
#include <unistd.h>

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

    if (pid == 0) {
        printf("Child Process\n");
        printf("Child PID: %d\n", getpid());
        printf("Parent PID (PPID): %d\n", getppid());
    }
    else if (pid > 0) {
        printf("Parent Process\n");
        printf("Parent PID: %d\n", getpid());
        printf("Child PID: %d\n", pid);
    }
    else {
        printf("Fork failed\n");
    }

    return 0;
}
```

```
PARAS@LAPTOP-DL5TDAQK MSYS ~
$ nano fork.c

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ gcc fork.c -o fork

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ ./fork.c
./fork.c: line 4: syntax error near unexpected token `('
./fork.c: line 4: `int main() {'

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ nano fork.c

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ gcc fork.c -o fork

PARAS@LAPTOP-DL5TDAQK MSYS ~
$ ./fork
Child Process
Child PID: 2273
Parent PID (PPID): 2272
Parent Process
Parent PID: 2272
Child PID: 2273
```