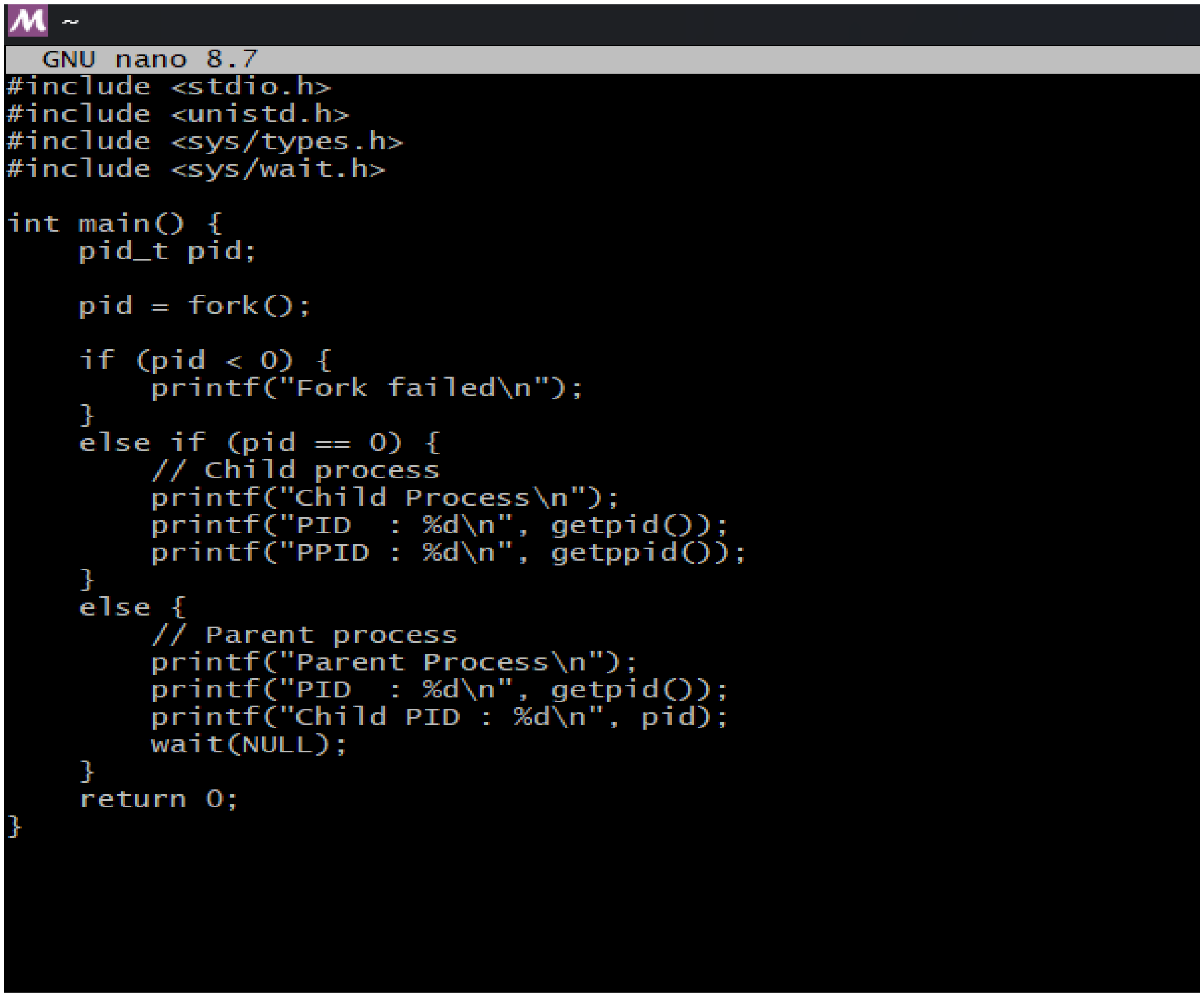
1.Adam is working in an IT company. He has been given a task to reduce the load of a system by killing some of the processes running in the LINUX operating system. Which commands will he use to complete the given task with the help of the following operation?

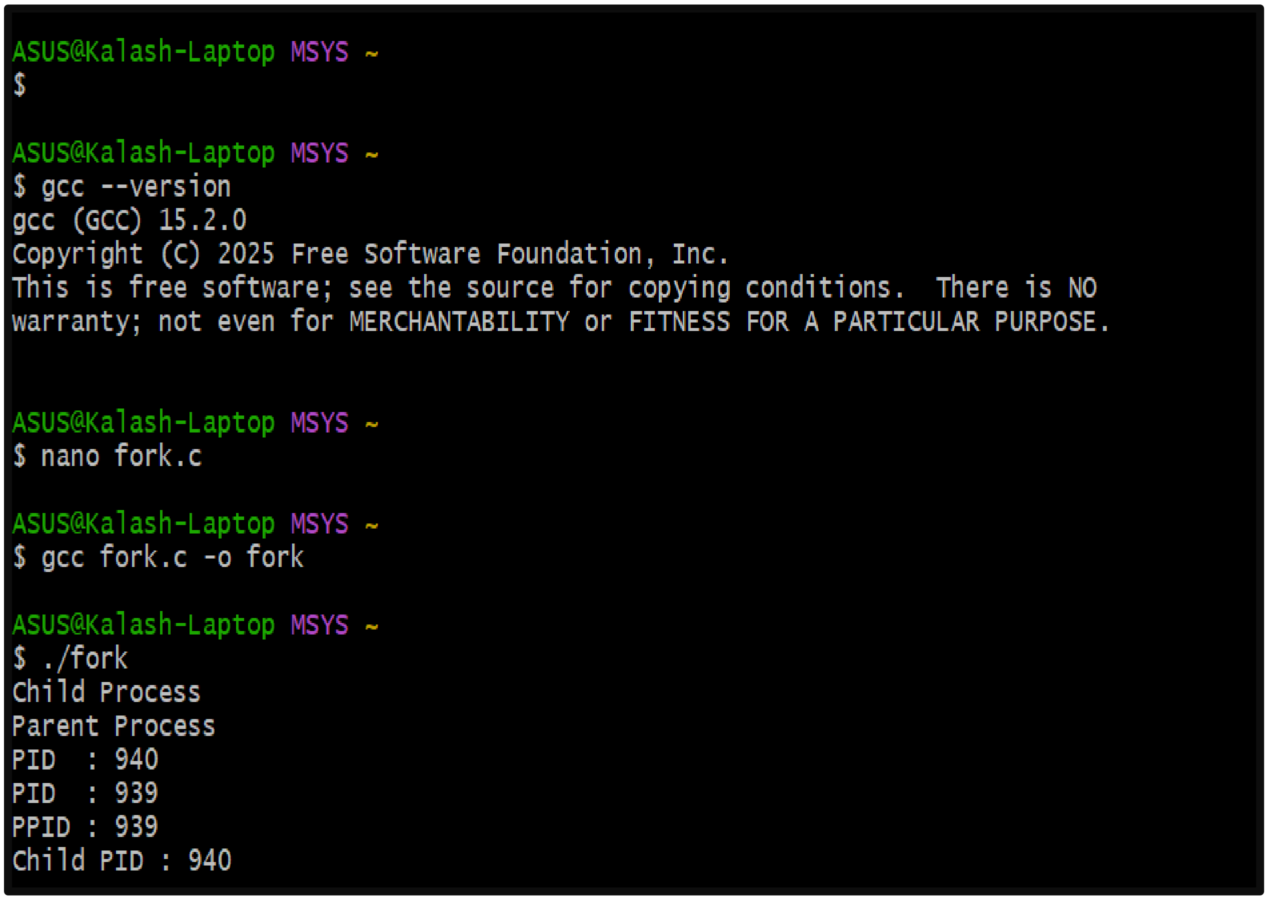
(i) Kill processes by name
(ii) Kill a process based on the process name
(iii) Kill a single process at a time with the given process ID

CODE:



```
  GNU nano 8.7
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

int main() {
    pid_t pid;

    pid = fork();

    if (pid < 0) {
        printf("Fork failed\n");
    }
    else 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);
        wait(NULL);
    }
    return 0;
}
```

OUTPUT:

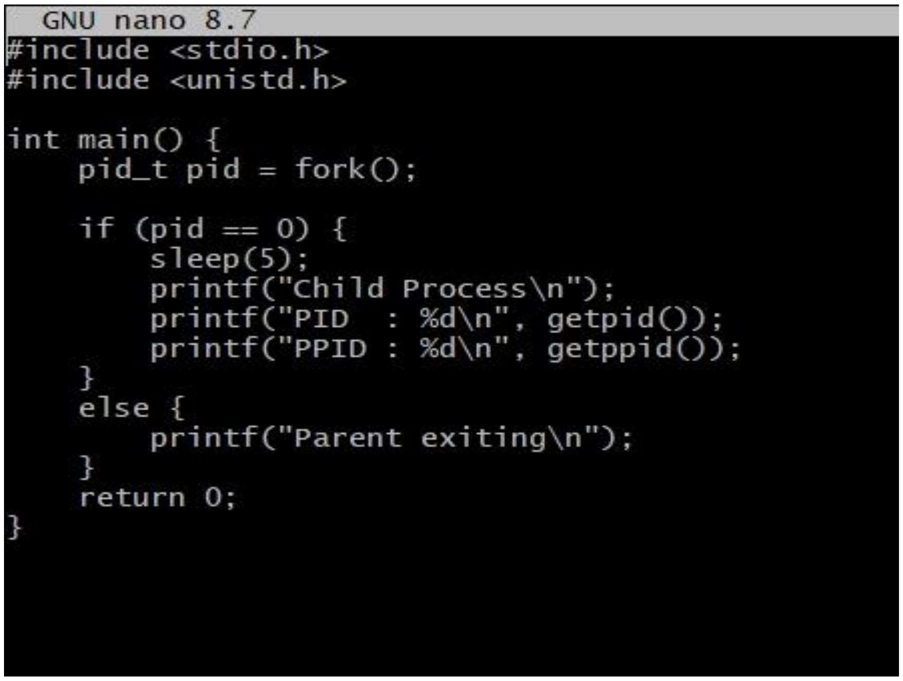


```
ASUS@Kalash-Laptop MSYS ~
$

ASUS@Kalash-Laptop MSYS ~
$ gcc --version
gcc (GCC) 15.2.0
Copyright (C) 2025 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.


ASUS@Kalash-Laptop MSYS ~
$ nano fork.c

ASUS@Kalash-Laptop MSYS ~
$ gcc fork.c -o fork

ASUS@Kalash-Laptop MSYS ~
$ ./fork
Child Process
Parent Process
PID  : 940
PID  : 939
PPID : 939
Child PID : 940
```

2. Write a program for process creation using C

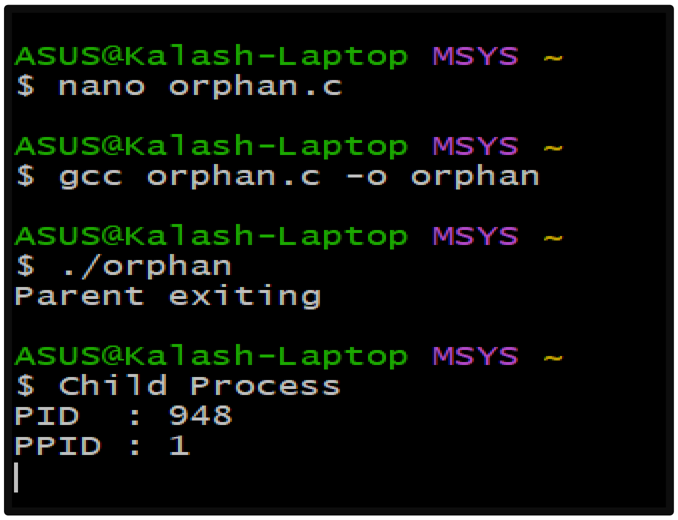
(i) Orphan Process
(ii) Zombine Process

CODE:



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

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

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

OUTPUT:

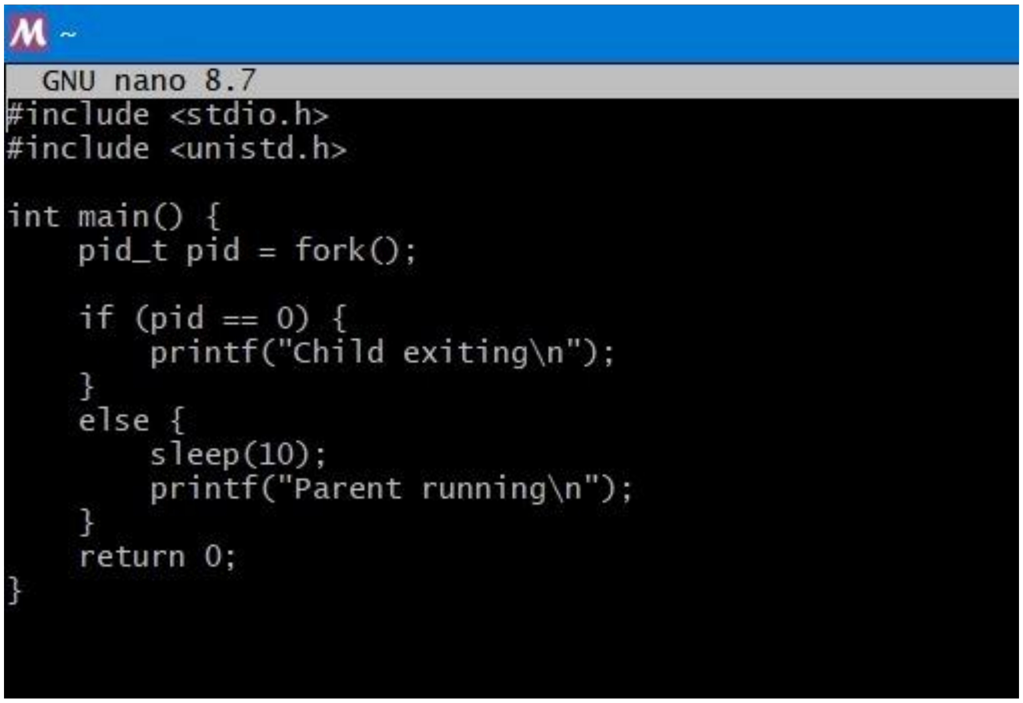


```
ASUS@Kalash-Laptop MSYS ~
$ nano orphan.c

ASUS@Kalash-Laptop MSYS ~
$ gcc orphan.c -o orphan

ASUS@Kalash-Laptop MSYS ~
$ ./orphan
Parent exiting

ASUS@Kalash-Laptop MSYS ~
$ Child Process
PID   : 948
PPID : 1
```

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

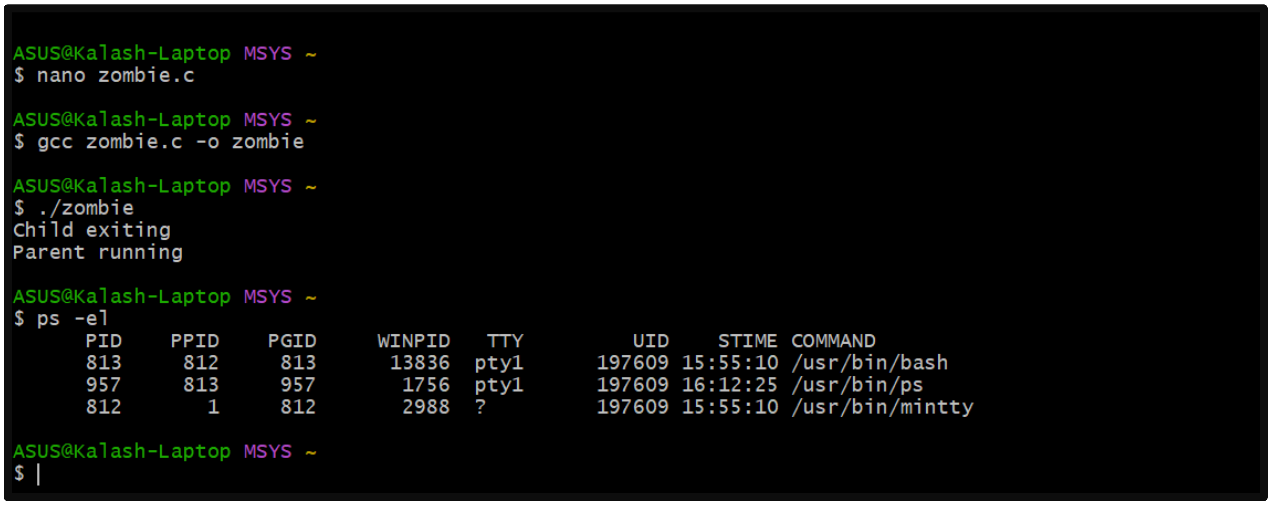
(i) Child Process creation
(ii) Parent process creation
(iii) PPID and PID

CODE:



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

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

    if (pid == 0) {
        printf("Child exiting\n");
    }
    else {
        sleep(10);
        printf("Parent running\n");
    }
    return 0;
}
```

OUTPUT:



```
ASUS@Kalash-Laptop MSYS ~
$ nano zombie.c

ASUS@Kalash-Laptop MSYS ~
$ gcc zombie.c -o zombie

ASUS@Kalash-Laptop MSYS ~
$ ./zombie
Child exiting
Parent running

ASUS@Kalash-Laptop MSYS ~
$ ps -el
      PID    PPID    PGID     WINPID   TTY         UID    STIME COMMAND
      813     812     813      13836  pty1      197609 15:55:10 /usr/bin/bash
      957     813     957       1756  pty1      197609 16:12:25 /usr/bin/ps
      812       1     812       2988  ?         197609 15:55:10 /usr/bin/mintty

ASUS@Kalash-Laptop MSYS ~
$ |
```