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:

```
Pranjali@DESKTOP-1KIKHDU 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.


Pranjali@DESKTOP-1KIKHDU MSYS ~
$ nano fork.c

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ gcc fork.c

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ gcc fork.c -o fork

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ ./fork
Parent Process
Child Process
PID  : 248
Child PID : 249
PID  : 249
PPID : 248

Pranjali@DESKTOP-1KIKHDU MSYS ~
$
```

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:

```
Pranjali@DESKTOP-1KIKHDU MSYS ~
$ nano orphan.c

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ gcc orphan.c -o orphan

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ ./orphan
Parent exiting

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ Child Process
PID  : 273
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:

```
Pranjali@DESKTOP-1KIKHDU MSYS ~
$ nano zombie.c

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ gcc zombie.c -o zombie

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ ./zombie
Child exiting
Parent running

Pranjali@DESKTOP-1KIKHDU MSYS ~
$ ps el
      PID    PPID    PGID     WINPID   TTY         UID    STIME COMMAND
      283      90     283       2720  pty0      197609 20:05:32 /usr/bin/ps
       90      89      90      19872  pty0      197609 19:43:39 /usr/bin/bash
       89       1      89      15652  ?         197609 19:43:39 /usr/bin/mintty

Pranjali@DESKTOP-1KIKHDU MSYS ~
$
```