



# National Textile University

## Department of Computer Science

Subject:

Operating System

---

Submitted to:

Sir Nasir Mehmood

---

Submitted by:

Eman Marium Tariq Rao

---

Reg number:

23-NTU-CS-1150

---

Semester:

05

---

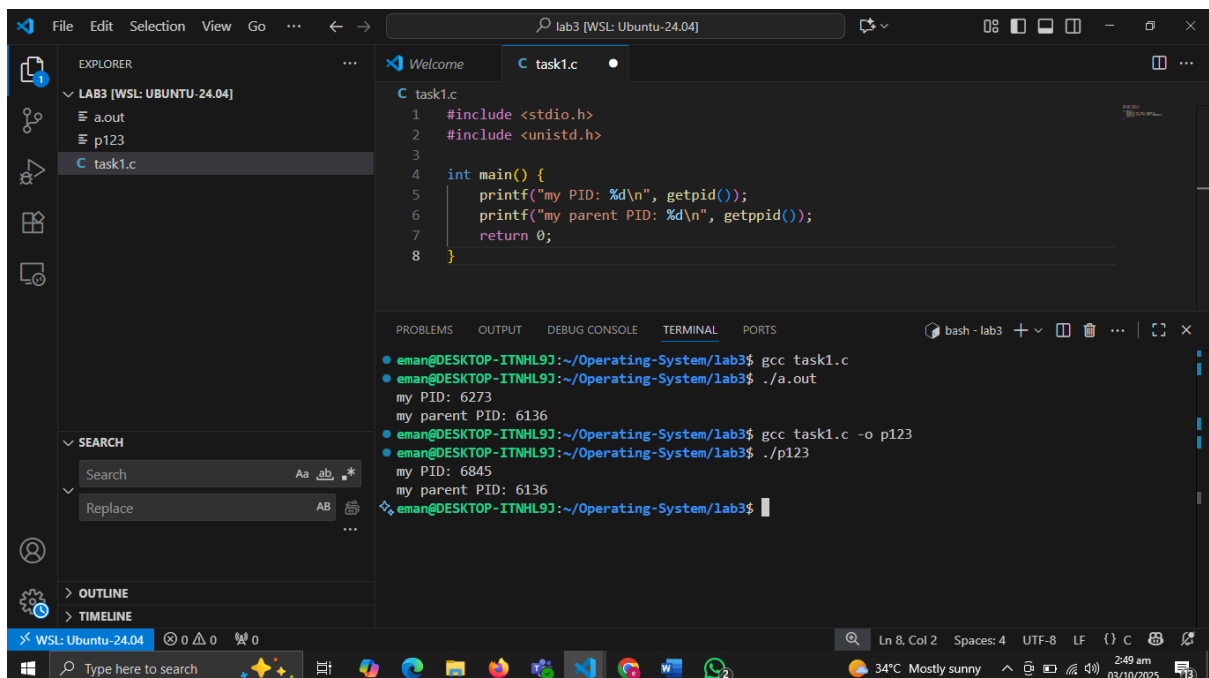
### 3. C Programs on Processes

#### Program 1: Print PID and PPID

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

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

```
int main() {  
    printf("my PID: %d\n", getpid());  
    printf("my parent PID: %d\n", getppid());  
    return 0;  
}
```



The screenshot shows the Visual Studio Code editor with a file named `task1.c` open. The code in the editor is as follows:

```
1 #include <stdio.h>  
2 #include <unistd.h>  
3  
4 int main() {  
5     printf("my PID: %d\n", getpid());  
6     printf("my parent PID: %d\n", getppid());  
7     return 0;  
8 }
```

The terminal window at the bottom shows the execution of the program. The commands and their outputs are:

```
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$ gcc task1.c  
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$ ./a.out  
my PID: 6273  
my parent PID: 6136  
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$ gcc task1.c -o p123  
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$ ./p123  
my PID: 6845  
my parent PID: 6136  
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$
```

#### Program 2: Fork – Creating Child Process

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

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

```
#include <sys/types.h> // add this header for pid_t
```

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

```

if (pid == 0) {

    // This block runs in the child process

    printf("Child: PID=%d, Parent=%d\n", getpid(), getppid());

} else {

    // This block runs in the parent process

    printf("Parent: PID=%d, Child=%d\n", getpid(), pid);

}

return 0;

}

```

The screenshot shows the Visual Studio Code editor interface. The Explorer pane on the left displays the file structure of a project named 'LAB3 [WSL: UBUNTU-24.04]', including files like 'a.out', 'p.exe.c', 'p123', 'simple', 'simple-process.c', and 'task1.c'. The main editor window shows the code for 'simple-process.c', which is a C program that uses `fork()` to create a child process and `printf()` to display the PIDs of both parent and child processes. The terminal at the bottom shows the execution of the program, with the output: 'Parent: PID=11436, Child=11437' and 'Child: PID=11437, Parent=11436'.

```

1  #include <stdio.h>
2  #include <unistd.h>
3  #include <sys/types.h> // add this header for pid_t
4
5  int main() {
6      pid_t pid = fork();
7
8      if (pid == 0) {
9          // This block runs in the child process
10         printf("Child: PID=%d, Parent=%d\n", getpid(), getppid());
11     } else {
12         // This block runs in the parent process
13         printf("Parent: PID=%d, Child=%d\n", getpid(), pid);
14     }
15
16     return 0;
17 }

```

```

eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$ gcc simple-process.c
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$ gcc simple-process.c -o simple
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$ ./a.out
Parent: PID=11436, Child=11437
Child: PID=11437, Parent=11436
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$

```

### Program 3: Execl – Replacing a Process

```

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h> // required for pid_t

int main() {

    pid_t pid = fork();

```

```

if (pid == 0) {

    // Child process

    execlp("ls", "ls", "-l", NULL);

    // This line only runs if execlp fails

    printf("This will not print if exec succeeds.\n");

} else {

    // Parent process

    printf("Parent still running...\n");

}

return 0;

}

```

The screenshot shows the Visual Studio Code editor with a C program named `p.exe.c` open. The program uses `fork()` to create a child process. The child process attempts to execute `ls` using `execlp`. The parent process prints "Parent still running..." and then lists the contents of the directory using `ls`. The terminal output shows the execution of the program and the resulting directory listing.

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include <sys/types.h> // required for pid_t
4
5 int main() {
6     pid_t pid = fork();
7
8     if (pid == 0) {
9         // Child process
10        execlp("ls", "ls", "-l", NULL);
11        // This line only runs if execlp fails
12        printf("This will not print if exec succeeds.\n");
13    } else {
14        // Parent process
15        printf("Parent still running...\n");
16    }
17
18    return 0;
19 }

```

```

eman@DESKTOP-ITNHL92:~/Operating-System/lab3$ ./execute
Parent still running...
total 96
-rwxr-xr-x 1 eman eman 16048 Oct  3 03:11 a.out
-rwxr-xr-x 1 eman eman 16048 Oct  3 03:13 execute
-rw-r--r-- 1 eman eman 422 Oct  3 03:04 p.exe.c
-rwxr-xr-x 1 eman eman 16048 Oct  3 02:47 p123
-rwxr-xr-x 1 eman eman 16096 Oct  3 03:06 simple
-rw-r--r-- 1 eman eman 412 Oct  3 03:02 simple-process.c
-rw-r--r-- 1 eman eman 152 Oct  3 03:05 task1.c
-rwxr-xr-x 1 eman eman 16088 Oct  3 03:08 wait
-rw-r--r-- 1 eman eman 328 Oct  3 03:07 wait.c

```

## Program 4: Wait – Synchronization

```

#include <stdio.h>

#include <unistd.h>

#include <sys/wait.h>

int main() {

    pid_t pid = fork();

```

```

if (pid == 0) {
    execlp("ls", "ls", "-l", NULL);

    printf("This will not print if exec succeeds.\n");
} else {
    waitpid(pid, NULL, 0); // Wait for the child process to finish

    printf("Parent still running...\n");
}

return 0;
}

```

```

C wait.c
1  #include <stdio.h>
2  #include <unistd.h>
3  #include <sys/wait.h>
4  int main() {
5      pid_t pid = fork();
6      if (pid == 0) {
7          execlp("ls", "ls", "-l", NULL);
8          printf("This will not print if exec succeeds.\n");
9      } else {
10         waitpid(pid, NULL, 0); // Wait for the child process to finish
11         printf("Parent still running...\n");
12     }
13     return 0;

```

```

eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$ ./wait
total 80
-rwxr-xr-x 1 eman eman 16088 Oct 3 03:09 a.out
-rw-r--r-- 1 eman eman 422 Oct 3 03:04 p.exe.c
-rwxr-xr-x 1 eman eman 16048 Oct 3 02:47 p123
-rwxr-xr-x 1 eman eman 16096 Oct 3 03:06 simple
-rw-r--r-- 1 eman eman 412 Oct 3 03:02 simple-process.c
-rw-r--r-- 1 eman eman 152 Oct 3 03:05 task1.c
-rwxr-xr-x 1 eman eman 16088 Oct 3 03:08 wait
-rw-r--r-- 1 eman eman 328 Oct 3 03:07 wait.c
Parent still running...
eman@DESKTOP-ITNHL9J:~/Operating-System/lab3$

```

## 2. Linux Process Commands

### 2.1 Viewing Processes

#### ps → Process Status

Shows processes in the current terminal session.

```
eman@DESKTOP-ITNHL9J: ~$ ps
  PID TTY          TIME CMD
 14996 pts/8    00:00:00 bash
 15117 pts/8    00:00:00 ps
eman@DESKTOP-ITNHL9J: ~$
```

**ps -ef → Full list of all processes**

```
eman@DESKTOP-ITNHL9J: ~$ ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1        0  0  02:06 ?        00:00:02 /sbin/init
root           2          1  0  02:06 ?        00:00:00 /init
root           9          2  0  02:06 ?        00:00:00 plan9 --control-socket 7 --log-level 4 --server-fd 8 --pipe-1
root          70          1  0  02:07 ?        00:00:01 /usr/lib/systemd/systemd-journald
root         122          1  0  02:07 ?        00:00:01 /usr/lib/systemd/systemd-udevd
systemd+     135          1  0  02:07 ?        00:00:00 /usr/lib/systemd/systemd-resolved
systemd+     136          1  0  02:07 ?        00:00:00 /usr/lib/systemd/systemd-timesyncd
root         183          1  0  02:07 ?        00:00:00 /usr/sbin/cron -f -P
message+     184          1  0  02:07 ?        00:00:00 @dbus-daemon --system --address=systemd: --nofork --nopidfil
root         197          1  0  02:07 ?        00:00:00 /usr/lib/systemd/systemd-logind
root         203          1  0  02:07 ?        00:00:00 /usr/libexec/wsl-pro-service -vv
syslog       207          1  0  02:07 ?        00:00:00 /usr/sbin/rsyslogd -n -iNONE
root         211          1  0  02:07 hvcd0  00:00:00 /sbin/agetty -o -p -- \u --noclear --keep-baud - 115200,3840
root         215          1  0  02:07 tty1    00:00:00 /sbin/agetty -o -p -- \u --noclear - linux
root         222          1  0  02:07 ?        00:00:00 /usr/bin/python3 /usr/share/unattended-upgrades/unattended-up
root         342          2  0  02:07 pts/1    00:00:00 /bin/login -f
eman         395          1  0  02:07 ?        00:00:00 /usr/lib/systemd/systemd --user
eman         396        395  0  02:07 ?        00:00:00 (sd-pam)
eman         418        342  0  02:07 pts/1    00:00:00 -bash
root         500          1  0  02:07 ?        00:00:00 /init
root        2457          2  0  02:24 ?        00:00:00 /init
root        2458       2457  0  02:24 ?        00:00:00 /init
eman         2459       2458  0  02:24 pts/0    00:00:00 sh -c "$VSCODE_WSL_EXT_LOCATION/scripts/wslServer.sh" e3a5ac
eman         2460       2459  0  02:24 pts/0    00:00:00 sh /mnt/c/Users/DELL/.vscode/extensions/ms-vscode-remote.rem
eman         2466       2460  0  02:24 pts/0    00:00:00 sh /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413
eman         2470       2466  0  02:24 pts/0    00:00:34 /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d56
```

## ps-ef | grep bash

This finds all processes related to the bash shell.

```
eman@DESKTOP-ITNHL9J: ~$ ps -ef | grep eman
eman      395      1    0 02:07 ?           00:00:00 /usr/lib/systemd/systemd --user
eman      396     395    0 02:07 ?           00:00:00 (sd-pam)
eman      418     342    0 02:07 pts/1        00:00:00 -bash
eman     2459    2458    0 02:24 pts/0        00:00:00 sh -c "$VSCODE_WSL_EXT_LOCATION/scripts/wslServer.sh" e3a5acfb517a443235981655413d566533107e92 stable code-server .vscode-server --host=127.0.0.1 --port=0 --connection-token=2661082583-3824556303-3716370881-3996798894 --use-host-proxy --without-browser-env-var --disable-websocket-compression --accept-server-license-terms --telemetry-level=all
eman     2460    2459    0 02:24 pts/0        00:00:00 sh /mnt/c/Users/DELL/.vscode/extensions/ms-vscode-remote.remote-wsl-0.104.3/scripts/wslServer.sh e3a5acfb517a443235981655413d566533107e92 stable code-server .vscode-server --host=127.0.0.1 --port=0 --connection-token=2661082583-3824556303-3716370881-3996798894 --use-host-proxy --without-browser-env-var --disable-websocket-compression --accept-server-license-terms --telemetry-level=all
eman     2466    2460    0 02:24 pts/0        00:00:00 sh /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d566533107e92/bin/code-server --host=127.0.0.1 --port=0 --connection-token=2661082583-3824556303-3716370881-3996798894 --use-host-proxy --without-browser-env-var --disable-websocket-compression --accept-server-license-terms --telemetry-level=all
eman     2470    2466    0 02:24 pts/0        00:00:35 /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d566533107e92/node /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d566533107e92/out/server-main.js --host=127.0.0.1 --port=0 --connection-token=2661082583-3824556303-3716370881-3996798894 --use-host-proxy --without-browser-env-var --disable-websocket-compression --accept-server-license-terms --telemetry-level=all
eman     2771    2770    0 02:25 pts/3        00:00:02 /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d566533107e92/node -e const net = require('net'); process.stdin.pause(); const client = net.createConnection({ host: '127.0.0.1', port: 45787 }, () => { client.pipe(process.stdout); process.stdin.pipe(client); }); client.on('close', function (hadError) { console.error(hadError ? 'Remote close with error' : 'Remote close'); process.exit(hadError ? 1 : 0); }); client.on('error', function (err) { process.stderr.write(err && (err.stack || err.message) || String(err)); });
eman     2780    2779    0 02:25 pts/4        00:00:02 /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d566533107e92/node -e const net = require('net'); process.stdin.pause(); const client = net.createConnection({ host: '127.0.0.1', port: 45787 }, () => { client.pipe(process.stdout); process.stdin.pipe(client); }); client.on('close', function (hadError) { console.error(hadError ? 'Remote close with error' : 'Remote close'); process.exit(hadError ? 1 : 0); }); client.on('error', function (err) { process.stderr.write(err && (err.stack || err.message) || String(err)); });
```

```
eman@DESKTOP-ITNHL9J: ~$ ps -ef | grep "eman"
eman      395      1    0 02:07 ?           00:00:00 /usr/lib/systemd/systemd --user
eman      396     395    0 02:07 ?           00:00:00 (sd-pam)
eman      418     342    0 02:07 pts/1        00:00:00 -bash
eman     2459    2458    0 02:24 pts/0        00:00:00 sh -c "$VSCODE_WSL_EXT_LOCATION/scripts/wslServer.sh" e3a5acfb517a443235981655413d566533107e92 stable code-server .vscode-server --host=127.0.0.1 --port=0 --connection-token=2661082583-3824556303-3716370881-3996798894 --use-host-proxy --without-browser-env-var --disable-websocket-compression --accept-server-license-terms --telemetry-level=all
eman     2460    2459    0 02:24 pts/0        00:00:00 sh /mnt/c/Users/DELL/.vscode/extensions/ms-vscode-remote.remote-wsl-0.104.3/scripts/wslServer.sh e3a5acfb517a443235981655413d566533107e92 stable code-server .vscode-server --host=127.0.0.1 --port=0 --connection-token=2661082583-3824556303-3716370881-3996798894 --use-host-proxy --without-browser-env-var --disable-websocket-compression --accept-server-license-terms --telemetry-level=all
eman     2466    2460    0 02:24 pts/0        00:00:00 sh /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d566533107e92/bin/code-server --host=127.0.0.1 --port=0 --connection-token=2661082583-3824556303-3716370881-3996798894 --use-host-proxy --without-browser-env-var --disable-websocket-compression --accept-server-license-terms --telemetry-level=all
eman     2470    2466    0 02:24 pts/0        00:00:35 /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d566533107e92/node /home/eman/.vscode-server/bin/e3a5acfb517a443235981655413d566533107e92/out/server-main.js --host=127.0.0.1 --port=0 --connection-token=2661082583-3824556303-3716370881-3996798894 --use-host-proxy --without-browser-env-var --disable-websocket-compression --accept-server-license-terms --telemetry-level=all
```

## 2.2 Monitoring Processes Interactively

top → Dynamic process viewer

```
eman@DESKTOP-ITNHL9J: ~
top - 03:28:55 up 1:22, 1 user, load average: 0.16, 0.12, 0.12
Tasks: 49 total, 1 running, 48 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1.5 us, 2.8 sy, 0.0 ni, 95.5 id, 0.0 wa, 0.0 hi, 0.2 si, 0.0 st
MiB Mem : 3851.2 total, 1330.3 free, 2073.8 used, 586.3 buff/cache
MiB Swap: 1024.0 total, 1024.0 free, 0.0 used. 1777.4 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
 2801 eman       20   0   52.4g 741352 55424 S   5.6   18.8   2:38.68 node
 5988 eman       20   0   52.5g 717368 53632 S   5.0   18.2   1:58.61 node
 2470 eman       20   0   11.3g 156444 51840 S   0.3    4.0   0:35.53 node
 2771 eman       20   0 1017740 61656 43008 S   0.3    1.6   0:02.98 node
 5968 eman       20   0 1261784 58860 46848 S   0.3    1.5   0:01.98 node
10947 eman       20   0 1014972 57516 44928 S   0.3    1.5   0:00.55 node
16769 eman       20   0    9268   5376  3328 R   0.3    0.1   0:00.05 top
    1 root        20   0   21760 12400  9200 S   0.0    0.3   0:02.39 systemd
    2 root        20   0    3072   1792  1792 S   0.0    0.0   0:00.04 init-systemd(Ub
    9 root        20   0    3088   2144  1920 S   0.0    0.1   0:00.00 init
   70 root        19  -1   66816 16976 16080 S   0.0    0.4   0:01.06 systemd-journal
  122 root        20   0   25008  5888  4864 S   0.0    0.1   0:01.10 systemd-udev
  135 systemd+    20   0   21456 12544 10368 S   0.0    0.3   0:00.21 systemd-resolve
  136 systemd+    20   0   91024  7424  6656 S   0.0    0.2   0:00.24 systemd-timesyn
  183 root        20   0    4236  2432  2304 S   0.0    0.1   0:00.05 cron
  184 message+    20   0    9592  4864  4352 S   0.0    0.1   0:00.30 dbus-daemon
  197 root        20   0   17960  8320  7424 S   0.0    0.2   0:00.24 systemd-logind
```

## 2.3 Foreground and Background Jobs

Foreground: A process that takes control of the terminal until it finishes

**sleep 30 → You cannot type new commands until it finishes**

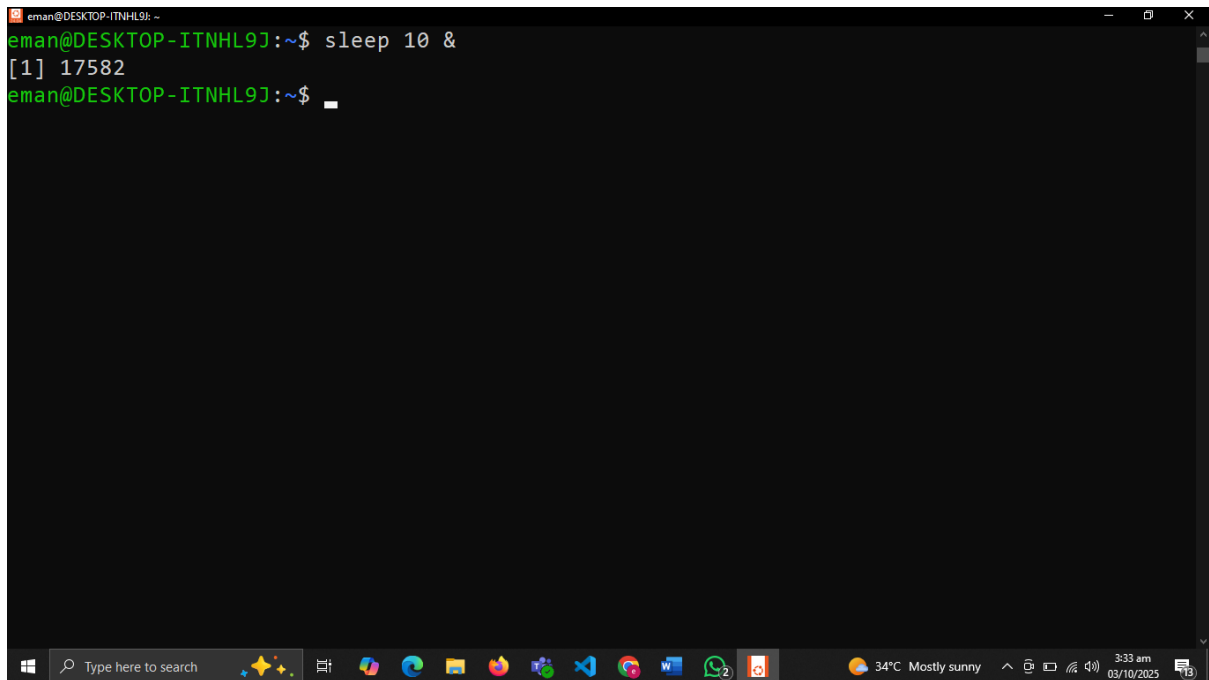
```
eman@DESKTOP-ITNHL9J: ~
eman@DESKTOP-ITNHL9J:~$ sleep 10
```



Background: Add & to run without blocking.

**sleep 30 &**

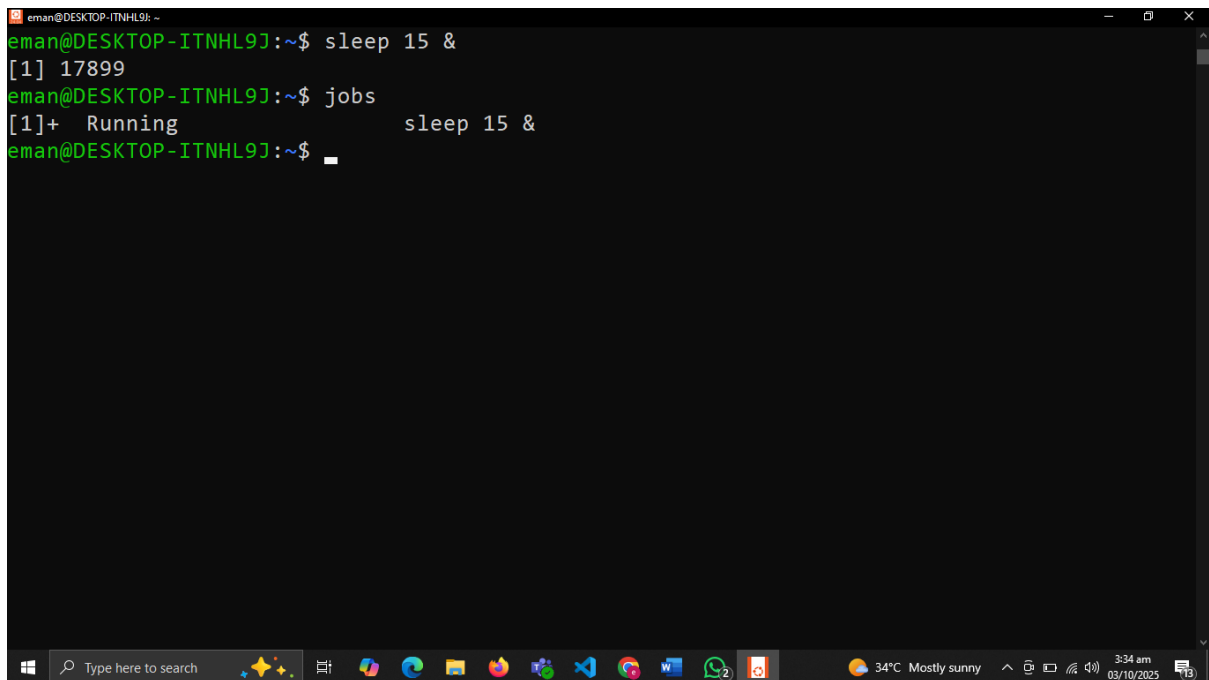
```
eman@DESKTOP-ITNHL9J: ~  
eman@DESKTOP-ITNHL9J:~$ sleep 10 &  
[1] 17582  
eman@DESKTOP-ITNHL9J:~$
```

A screenshot of a Windows terminal window. The title bar shows 'eman@DESKTOP-ITNHL9J: ~'. The terminal content shows the user entering 'sleep 10 &' at the prompt 'eman@DESKTOP-ITNHL9J:~\$'. The output is '[1] 17582'. The prompt returns to 'eman@DESKTOP-ITNHL9J:~\$'. The Windows taskbar is visible at the bottom with various icons and a system tray showing '34°C Mostly sunny' and the time '3:33 am 03/10/2025'.

Check background jobs:

**jobs**

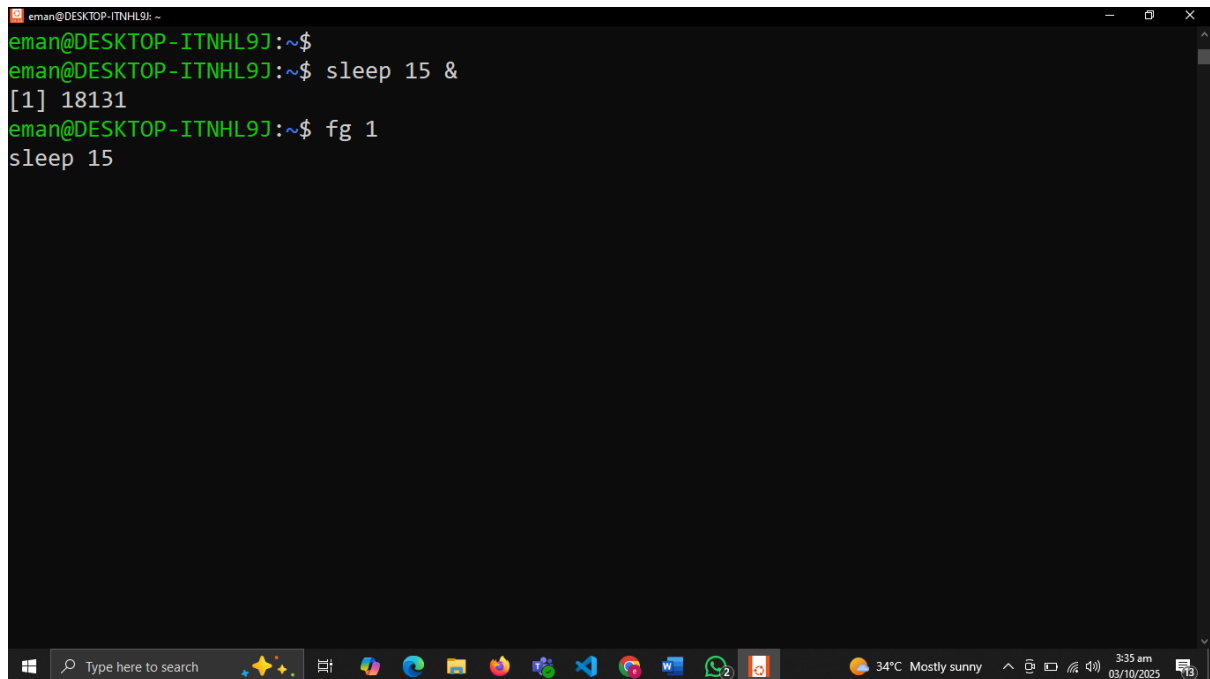
```
eman@DESKTOP-ITNHL9J:~$ sleep 15 &  
[1] 17899  
eman@DESKTOP-ITNHL9J:~$ jobs  
[1]+  Running                 sleep 15 &  
eman@DESKTOP-ITNHL9J:~$
```

A screenshot of a Windows terminal window. The title bar shows 'eman@DESKTOP-ITNHL9J: ~'. The terminal content shows the user entering 'sleep 15 &' at the prompt 'eman@DESKTOP-ITNHL9J:~\$'. The output is '[1] 17899'. The user then enters 'jobs' at the prompt 'eman@DESKTOP-ITNHL9J:~\$'. The output is '[1]+ Running sleep 15 &'. The prompt returns to 'eman@DESKTOP-ITNHL9J:~\$'. The Windows taskbar is visible at the bottom with various icons and a system tray showing '34°C Mostly sunny' and the time '3:34 am 03/10/2025'.

Bring a job to foreground:

**fg %1**

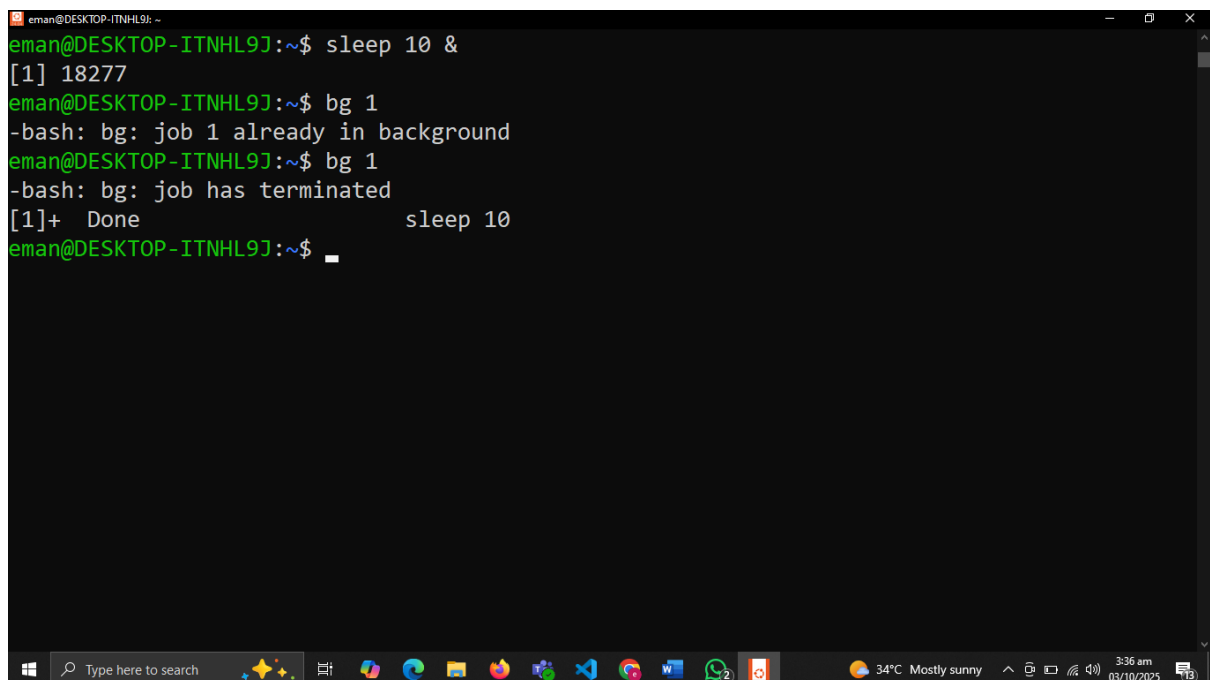
```
eman@DESKTOP-ITNHL9J: ~$  
eman@DESKTOP-ITNHL9J:~$ sleep 15 &  
[1] 18131  
eman@DESKTOP-ITNHL9J:~$ fg 1  
sleep 15
```

A terminal window titled 'eman@DESKTOP-ITNHL9J: ~' with a dark background. The user enters 'sleep 15 &' and receives '[1] 18131'. Then they enter 'fg 1' and the prompt returns to 'eman@DESKTOP-ITNHL9J:~\$'. The Windows taskbar is visible at the bottom with various application icons and system status information.

Resume suspended job in background:

**bg %1**

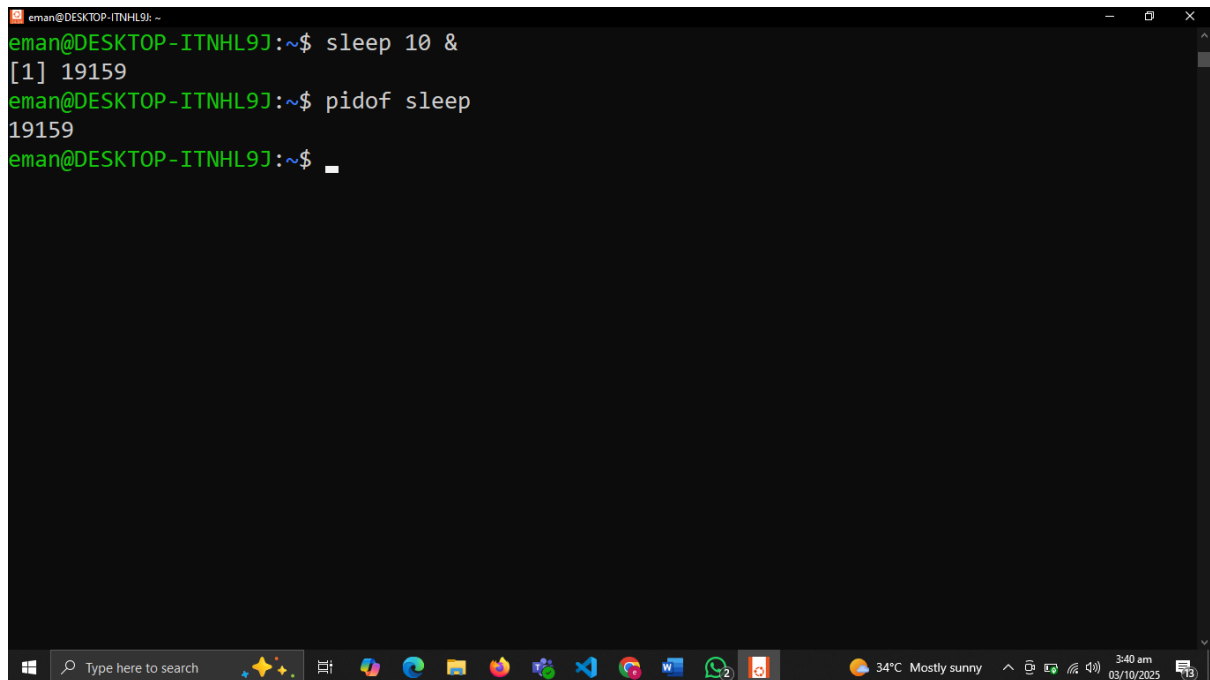
```
eman@DESKTOP-ITNHL9J:~$ sleep 10 &  
[1] 18277  
eman@DESKTOP-ITNHL9J:~$ bg 1  
-bash: bg: job 1 already in background  
eman@DESKTOP-ITNHL9J:~$ bg 1  
-bash: bg: job has terminated  
[1]+  Done                  sleep 10  
eman@DESKTOP-ITNHL9J:~$ _
```

A terminal window titled 'eman@DESKTOP-ITNHL9J: ~' with a dark background. The user enters 'sleep 10 &' and receives '[1] 18277'. Then they enter 'bg 1' and get '-bash: bg: job 1 already in background'. They enter 'bg 1' again and get '-bash: bg: job has terminated'. Then they enter 'bg 1' a third time and get '[1]+ Done sleep 10'. Finally, they enter 'bg 1' and get 'eman@DESKTOP-ITNHL9J:~\$ \_'. The Windows taskbar is visible at the bottom.

## 2.4 Process Identification

Get PID of a process by name:

**pidof sleep**

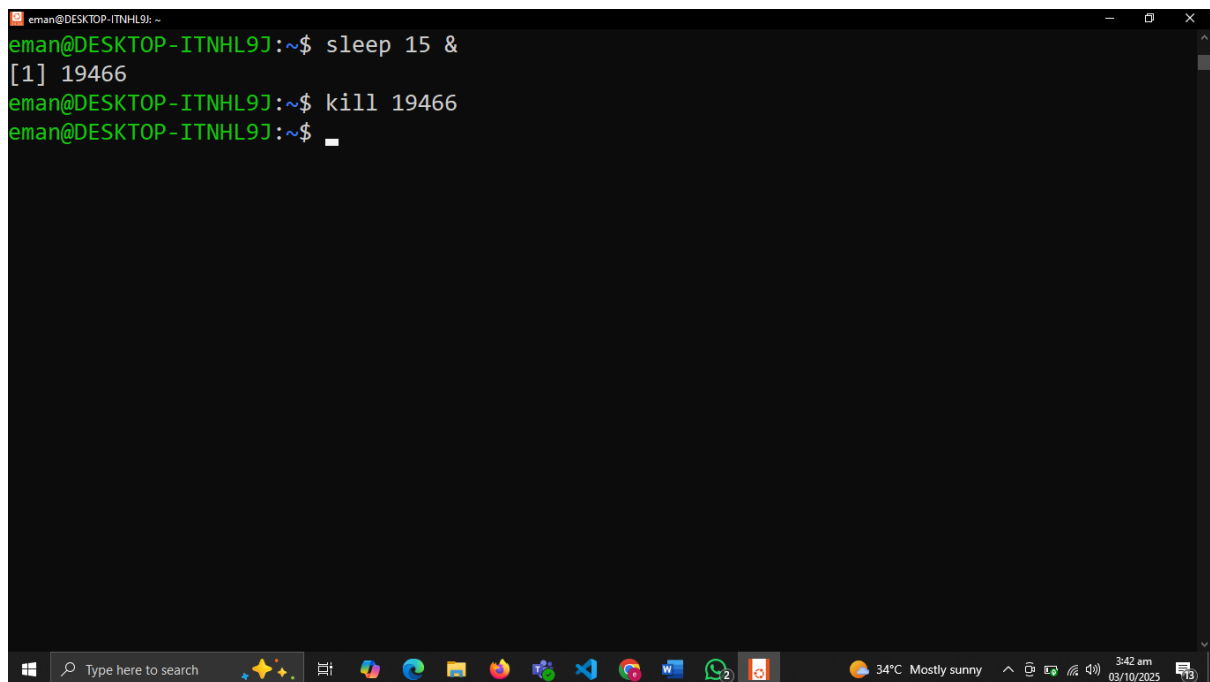
A terminal window titled 'eman@DESKTOP-ITNHL9J: ~' with a dark background. The user enters 'sleep 10 &' and the prompt returns '[1] 19159'. Then, the user enters 'pidof sleep' and the prompt returns '19159'. The terminal is open on a Windows desktop with a taskbar at the bottom showing various application icons and a system tray with weather and time information (3:40 am, 03/10/2025).

```
eman@DESKTOP-ITNHL9J: ~  
eman@DESKTOP-ITNHL9J:~$ sleep 10 &  
[1] 19159  
eman@DESKTOP-ITNHL9J:~$ pidof sleep  
19159  
eman@DESKTOP-ITNHL9J:~$
```

## 2.5 Killing Processes

Kill by PID:

**kill-9 3421**

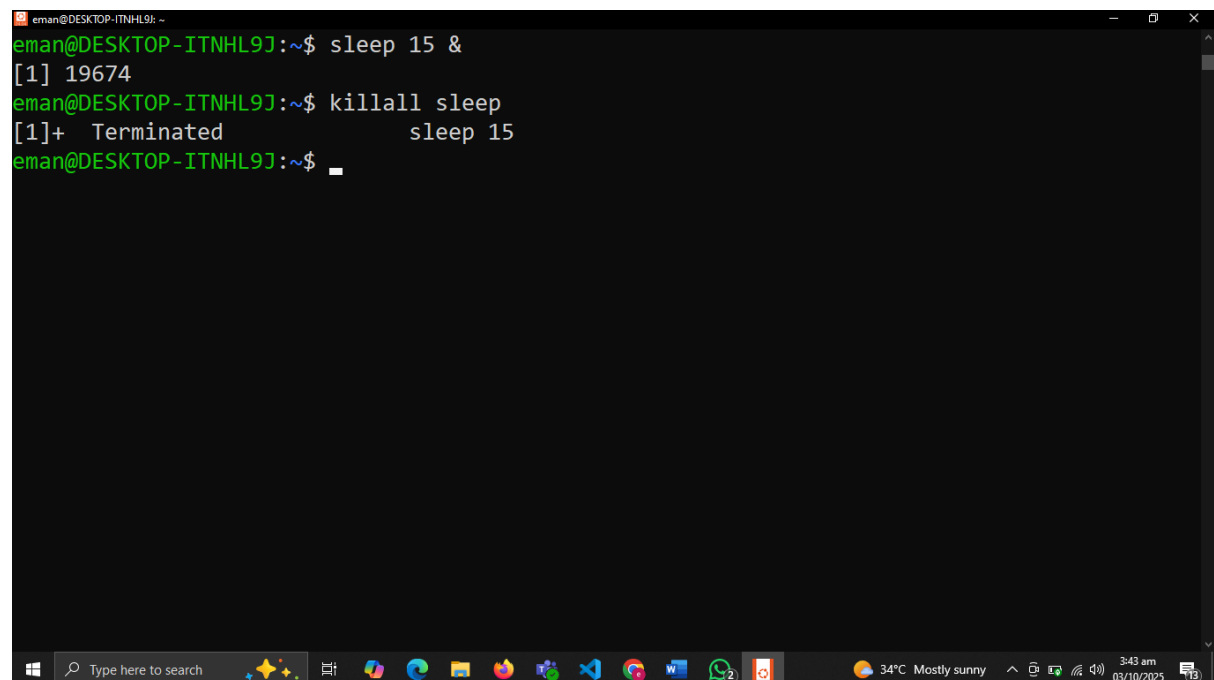
A terminal window titled 'eman@DESKTOP-ITNHL9J: ~' with a dark background. The user enters 'sleep 15 &' and the prompt returns '[1] 19466'. Then, the user enters 'kill 19466' and the prompt returns. The terminal is open on a Windows desktop with a taskbar at the bottom showing various application icons and a system tray with weather and time information (3:42 am, 03/10/2025).

```
eman@DESKTOP-ITNHL9J: ~  
eman@DESKTOP-ITNHL9J:~$ sleep 15 &  
[1] 19466  
eman@DESKTOP-ITNHL9J:~$ kill 19466  
eman@DESKTOP-ITNHL9J:~$
```

Kill all processes by name:

### killall sleep

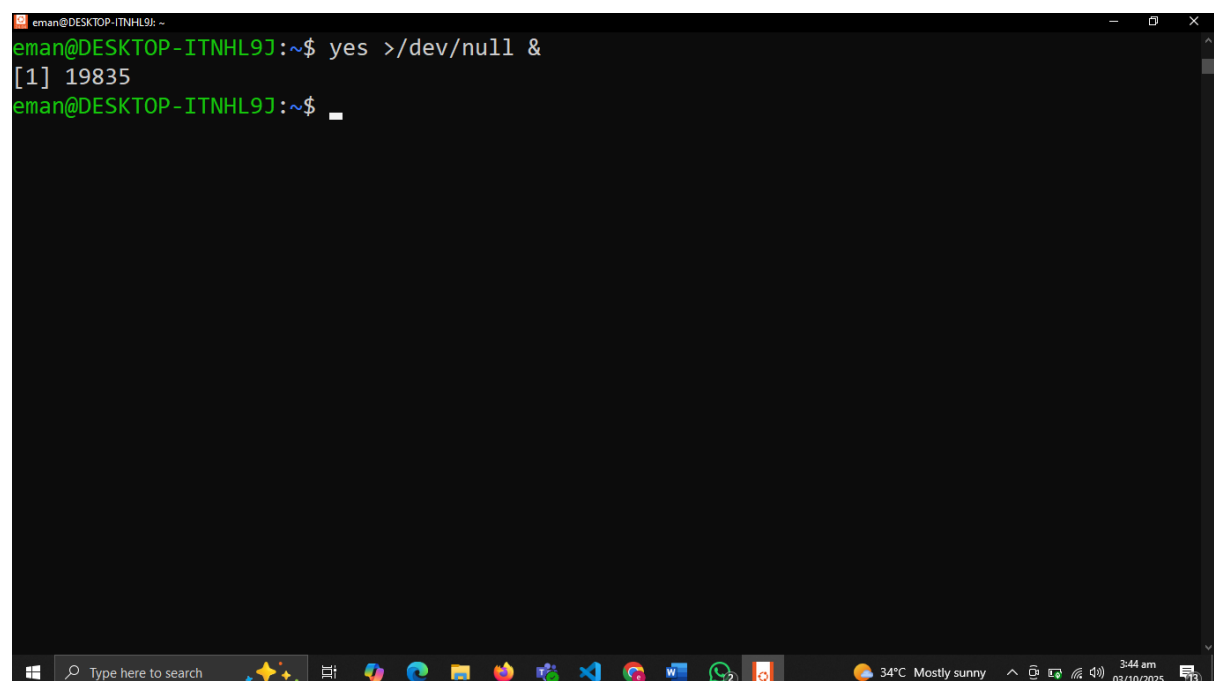
```
eman@DESKTOP-ITNHL9J: ~$ sleep 15 &
[1] 19674
eman@DESKTOP-ITNHL9J: ~$ killall sleep
[1]+  Terminated                  sleep 15
eman@DESKTOP-ITNHL9J: ~$
```



Run an infinite process:

### yes >/dev/null &

```
eman@DESKTOP-ITNHL9J: ~$ yes >/dev/null &
[1] 19835
eman@DESKTOP-ITNHL9J: ~$
```



Find it with:

**ps -ef | grep yes**

```
eman@DESKTOP-ITNHL9J: ~$ ps -ef | grep yes
eman      19835      14996  99 03:44 pts/8      00:01:05 yes
eman      20065      14996   0 03:45 pts/8      00:00:00 grep --color=auto yes
eman@DESKTOP-ITNHL9J: ~$
```

Kill it with:

**kill-9**

```
eman@DESKTOP-ITNHL9J: ~$ yes > /dev/null &
[1] 20608
eman@DESKTOP-ITNHL9J: ~$ kill -9 20608
eman@DESKTOP-ITNHL9J: ~$ jobs
[1]+  Killed                  yes > /dev/null
eman@DESKTOP-ITNHL9J: ~$
```