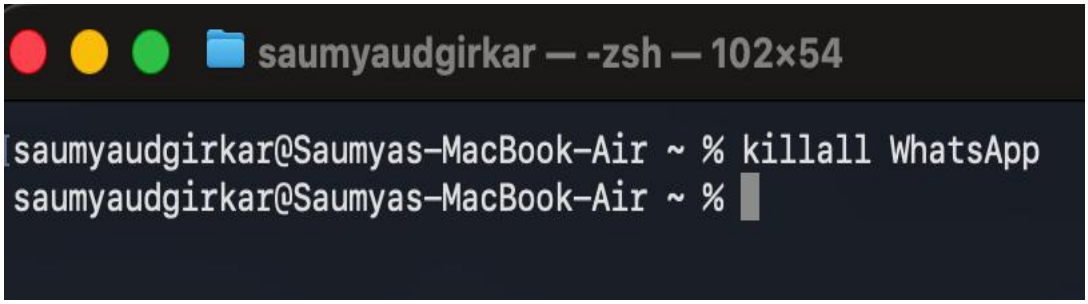
OS Practical No. 4

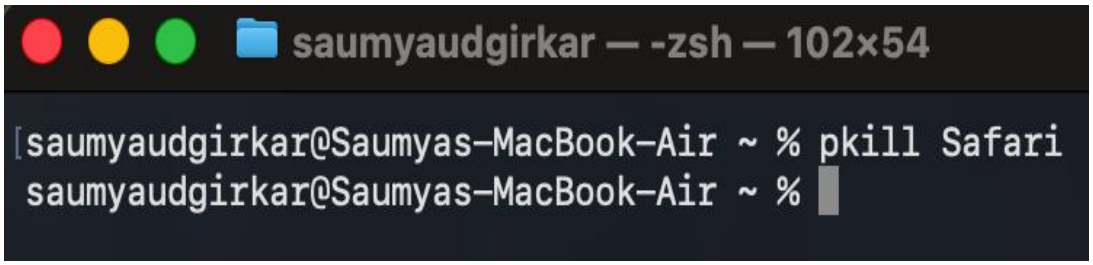
Aim :

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?

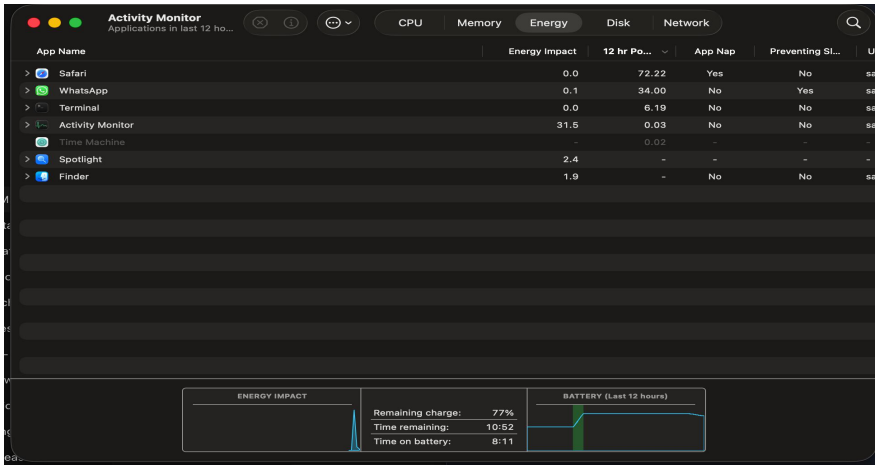
Kill processes by name

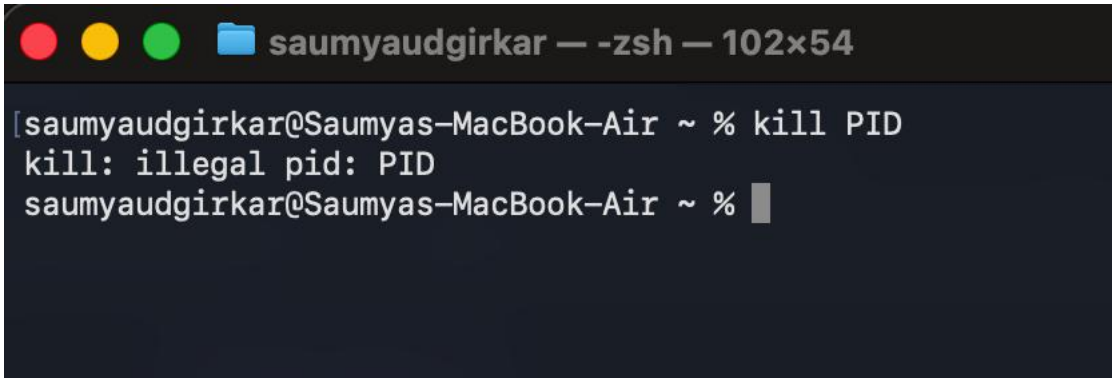


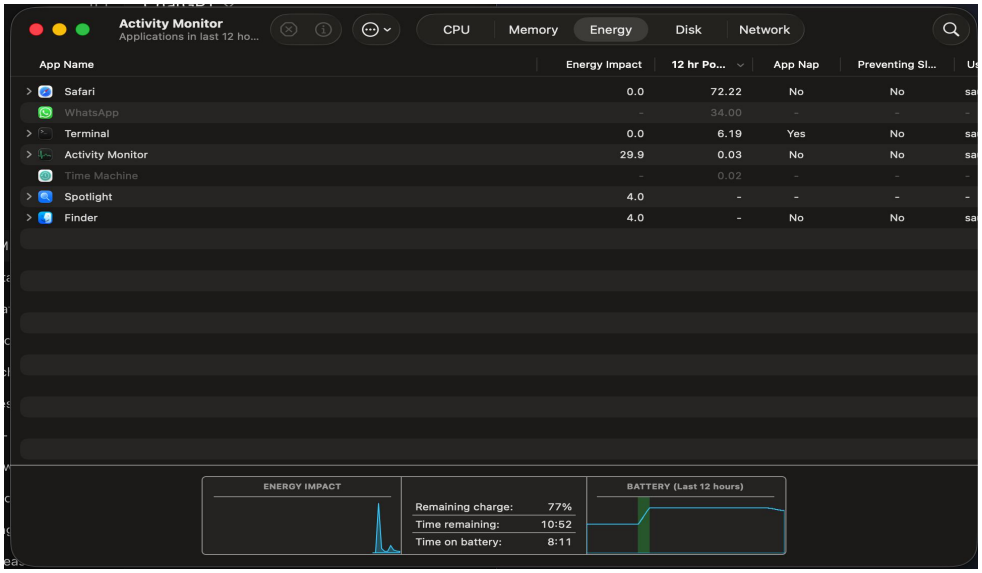
Kill a process based on the process name



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







2. Write a program for process creation using C

- Orphan Process

Code :

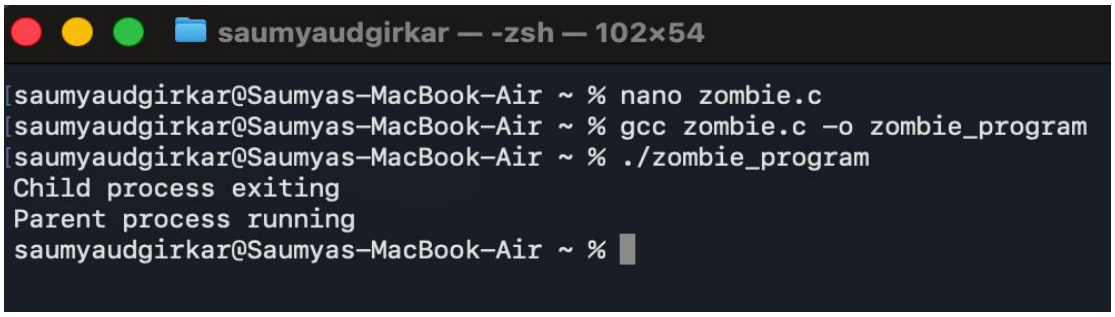
```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
int main() {
    pid_t pid = fork();
 if (pid > 0) {
        printf("Parent process exiting\n");
    }
    else if (pid == 0) {
        sleep(5);
        printf("Child process (Orphan) running\n");
    }
    return 0;
}
```

```
saumyaudgirkar@Saumyas-MacBook-Air ~ % nano hello.c
saumyaudgirkar@Saumyas-MacBook-Air ~ % gcc hello.c -o hello_program
saumyaudgirkar@Saumyas-MacBook-Air ~ % ./hello_program
Parent process exiting
saumyaudgirkar@Saumyas-MacBook-Air ~ % Child process (Orphan) running
```

- Zombie Process

Code :

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
int main() {
    pid_t pid = fork();
if (pid == 0) {
        printf("Child process exiting\n");
    }
    else {
        sleep(10);
        printf("Parent process running\n");
    }
    return 0;
}
```



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

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

Code :

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
int main() {
    pid_t pid;
   pid = fork();
if (pid == 0) {
        // Child process
        printf("Child Process\n");
        printf("Child PID  : %d\n", getpid());
        printf("Parent PID : %d\n", getppid());
    }
    else if (pid > 0) {
        // Parent process
        printf("Parent Process\n");
        printf("Parent PID : %d\n", getpid());
        printf("Child PID  : %d\n", pid);
    }
    else {
        printf("Fork failed\n");
    }
     return 0;
}
```