Practical-7

Write a C program to fork a seperate process using fork() or exec() systel calls.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>

int main() {
    pid_t pid;

    pid = fork(); // create a new process

    if (pid == -1) {
        // error occurred
        fprintf(stderr, "Failed to fork()\n");
        return 1;
    } else if (pid == 0) {
        // child process
        printf("Hello from child process! My PID is %d\n", getpid());
        execlp("/bin/ls", "ls", NULL);
        printf("This line should never be printed!\n");
        return 1;
    } else {
        // parent process
 printf("Hello from parent process! My PID is %d and my child's PID is %d\n", getpid(), pid);
        wait(NULL); // wait for child process to finish
        printf("Child process finished\n");
        return 0;
    }
}
```

Practical-7

```
Hello from parent process! My PID is 280 and my child's PID is 281
Hello from child process! My PID is 281
apache2  node_modules  sendsigs.omit.d          systemd
lock     programiz-oc  shm              user
log  pty.node      swift-5.7.2-RELEASE-ubuntu22.04
mount    secrets       swift.tar.gz
Child process finished
```