```c
#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <unistd.h>

#include <sys/wait.h>


int main() {

    pid_t pid = fork();


    if (pid < 0) {

        printf("Fork failed!\n");

        exit(1);

    }

    else if (pid == 0) { // Child Process

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


        // To demonstrate execve: replacing child process image

        char *args[] = {"/bin/ls", NULL};

        printf("Child is replacing its code with 'ls' using execve...\n");

        execve("/bin/ls", args, NULL);


        // If execve fails

        printf("Execve failed in child.\n");

        exit(1);

    }

    else { // Parent Process

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

        printf("Parent is waiting for child to terminate using wait()...\n");
        int status;
        wait(&status);

        printf("Child exited. Parent exiting.\n");
    }

    return 0;
}

/*OUTPUT-
Parent Process: PID=10256, Child PID=10257
Parent is waiting for child to terminate using wait()...
Child Process: PID=10257, Parent PID=10256
Child is replacing its code with 'ls' using execve...
[Child process runs `ls` command here — directory listing is printed]
Child exited. Parent exiting.
*/
```