

# JEGYZŐKÖNYV

Operációs rendszerek BSc

2021 tavasz féléves feladat

Készítette: **Garamszegi Márton**

Neptunkód: **AJYKQ3**

## A feladat leírása:

Írjon egy C programot, amely egy szülőprocessz révén készít egy gyermekprocesszt, a gyermekben futtasson egy másik programot az `execl()` hívással (Environemnten keresztül kapja meg, hogy mit indítson el a program), mely kiírja a PID-jét és szülője PID-jét, majd a szülő is kiírja mi a PID-je és a gyermeke PID-je.

## A feladat elkészítésének lépései:

A fordítónak szóló direktívák inkludálását követően a `main` függvénybe deklaráltam az `execl()` függvény argumentumait, illetve forkoltam a gyerek processzt.

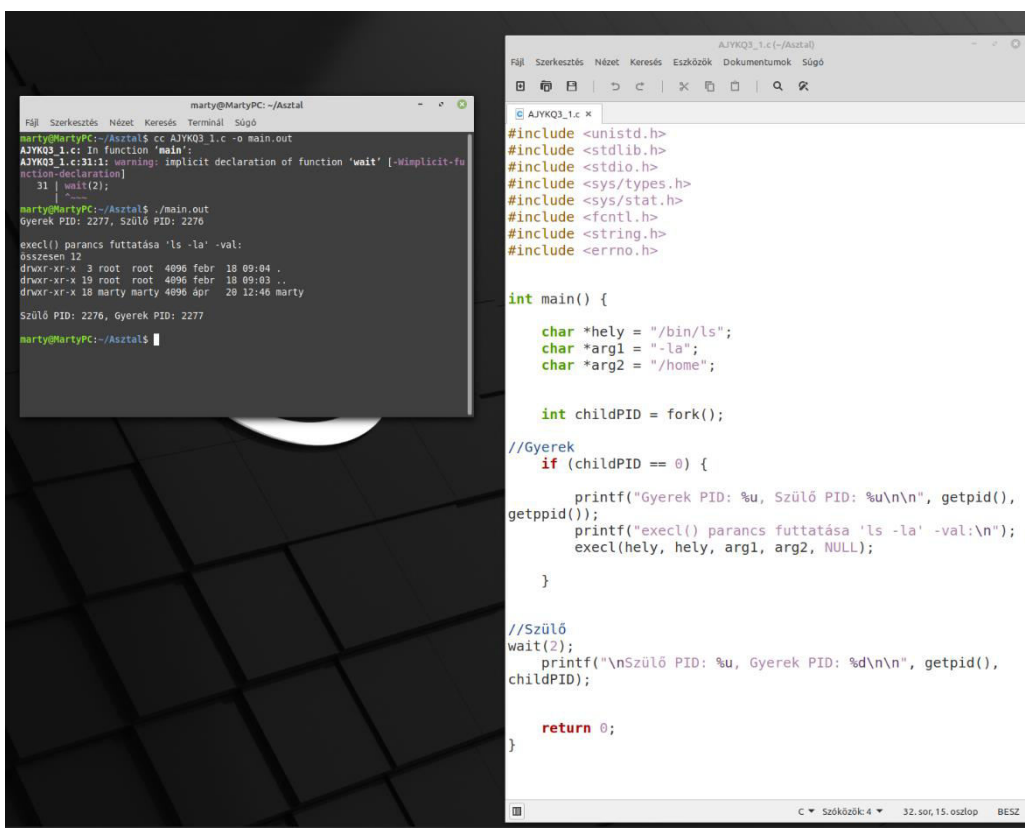
Ezután megírom a gyerek processzt:

Egy `if` szerkezet elindítja a gyerek processzt, ami kiírja a saját illetve a szülő PID-jét, majd futtatja az `execl()`-t a megadott argumentumokkal.

A szülő processz vár amíg a gyerek processz befejeződik, majd kiírja a saját és a gyerek PID-jét.

## A futtatás eredménye:

Fordításnál a fordító warning-ot dob, de ez nem befolyásolja a program futását vagy funkcionalitását.



The image shows a C program and its execution output. On the right, a code editor displays the source code for `AJYKQ3_1.c`. The code includes standard headers, defines `hely` as `"/bin/ls"`, `arg1` as `"-la"`, and `arg2` as `"/home"`. The `main` function forks a child process. The child process prints its own PID and the parent's PID, then calls `execl(hely, hely, arg1, arg2, NULL)` to execute `ls -la` in the parent's environment. The parent process calls `wait(2)` to wait for the child to finish, then prints both PIDs and returns 0.

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>
#include <errno.h>

int main() {

    char *hely = "/bin/ls";
    char *arg1 = "-la";
    char *arg2 = "/home";

    int childPID = fork();

    //Gyerek
    if (childPID == 0) {

        printf("Gyerek PID: %u, Szülő PID: %u\n\n", getpid(),
getppid());
        printf("execl() parancs futtatása 'ls -la' -val:\n");
        execl(hely, hely, arg1, arg2, NULL);

    }

    //Szülő
    wait(2);
    printf("\nSzülő PID: %u, Gyerek PID: %d\n\n", getpid(),
childPID);

    return 0;
}
```

On the left, a terminal window shows the execution of the program. It starts with a warning about the implicit declaration of `wait`. The program then prints the child PID (2277) and the parent PID (2276). It then shows the output of `ls -la` in the parent's environment, which lists files in the `/home` directory. Finally, it prints the parent PID (2276) and the child PID (2277).

```
marty@martyPC: ~/Asztal
marty@martyPC:~/Asztal$ cc AJYKQ3_1.c -o main.out
AJYKQ3_1.c: In function 'main':
AJYKQ3_1.c:31:1: warning: implicit declaration of function 'wait' [-Wimplicit-fu
nction-declaration]
   31 |     wait(2);
      |     ~~~~~
marty@martyPC:~/Asztal$ ./main.out
Gyerek PID: 2277, Szülő PID: 2276

execl() parancs futtatása 'ls -la' -val:
összesen 12
drwxr-xr-x 3 root root 4096 febr 18 09:04 .
drwxr-xr-x 18 root root 4096 febr 18 09:03 ..
drwxr-xr-x 18 marty marty 4096 apr 20 12:46 marty

Szülő PID: 2276, Gyerek PID: 2277
marty@martyPC:~/Asztal$
```