

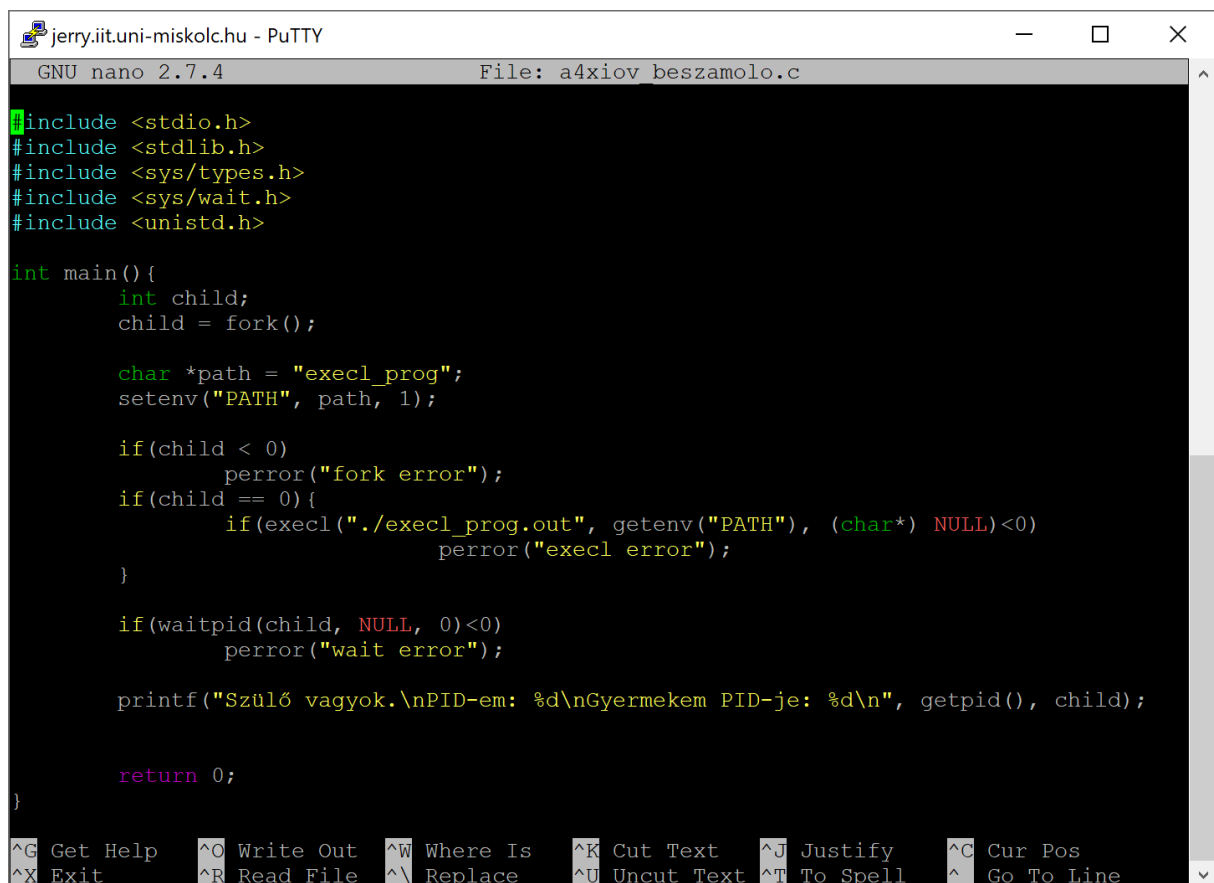
Beszámoló – Operációs Rendszerek

Feladat:

1. 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.

Szülő:

a4xiov_beszamolo.c



```
GNU nano 2.7.4 File: a4xiov_beszamolo.c

#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>

int main() {
    int child;
    child = fork();

    char *path = "execl_prog";
    setenv("PATH", path, 1);

    if(child < 0)
        perror("fork error");
    if(child == 0) {
        if(execl("./execl_prog.out", getenv("PATH"), (char*) NULL) < 0)
            perror("execl error");
    }

    if(waitpid(child, NULL, 0) < 0)
        perror("wait error");

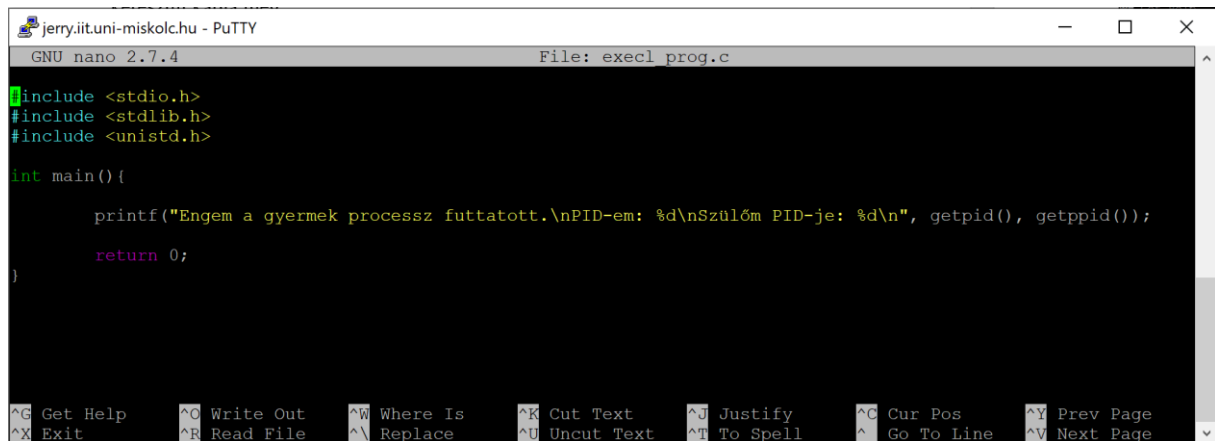
    printf("Szülő vagyok.\nPID-em: %d\nGyermeke PID-je: %d\n", getpid(), child);

    return 0;
}
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^ Go To Line

Gyermek:

execl_prog.c



```
GNU nano 2.7.4 File: execl_prog.c

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(){

    printf("Engem a gyermek processz futtatott.\nPID-em: %d\nSzülőm PID-je: %d\n", getpid(), getppid());

    return 0;
}
```

Futtatva:

```
abraham1@jerry:~$ ./a4xiov_beszamolo.out
Engem a gyermek processz futtatott.
PID-em: 23402
Szülőm PID-je: 23399
Szülő vagyok.
PID-em: 23399
Gyermekek PID-je: 23402
abraham1@jerry:~$
```