JEGYZŐKÖNYV

Operációs rendszerek BSc

2021 tavasz féléves feladat

Készítette: **Árvai Dóra**

Neptunkód: **G2SKZ4**

A feladat leírása

Írjon C nyelvű programot, ami

- létrehoz két csővezetéket (két file deszkriptor párt)
- elforkol
- a szülő elküldi a saját pidjét a gyermeknek az egyik csövön
- a gyermek kiírja a képernyőre és visszaküldi az övét a másik csövön
- megszűnnek a processzek (a szülő megvárja a gyereket).

A feladat elkészítésének lépései

Beillesztem a szükséges header állományokat.

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <wait.h>
```

A *main* függvényben deklarálom a szükséges változókat:

```c
// file descriptorok
int fd1[2]; // ebbe ír a szülő
int fd2[2]; // ebbe ír a gyerek

int pid = 0; // a pid értékének tárolására
int child;   // a fork() hívás eredményének tárolására
```

Létrehozom a két csővezetéket:

```c
// csővezetékek létrehozása
// hiba esetén kilép
if (pipe(fd1) || pipe(fd2)) {
    perror("Hiba a pipe letrehozasakor");
    return 1;
}
```

A szülő processzben lekérdezem az aktuális PID-et, és lezárom a nem használt file descriptorokat, majd elküldöm a PID-et a gyerek processznek. Ezután lezárom a szülő által használt WRITE descriptort.

```
// Gyerek processz létrehozása a fork() paranccsal
if ((child = fork()) ≠ 0) {
    // itt a szülő processzben vagyunk

    // pid lekérése
    pid = getpid();

    // lezárom a nem használt deszkriptorokat
    close(fd1[0]); // szülő READ
    close(fd2[1]); // gyerek WRITE

    printf("Szulo pid kuldese a gyereknek: %d\n", pid);
    // elküldöm a pid-et a gyerek processznek
    write(fd1[1], &pid, sizeof(pid));
    // lezárom a szülő WRITE deszkriptorát
    close(fd1[1]);
```

A gyerek processzben az alábbi módon olvasom be az elküldött PID-et:

```
else if (child == 0) {
    // itt a gyerek processzben vagyunk

    // lezárom a nem használt deszkriptorokat
    close(fd1[1]); // szülő WRITE
    close(fd2[0]); // gyerek READ

    // beolvasom a pid-et a csővezetékről
    read(fd1[0], &pid, sizeof(pid));
    // kiírom
    printf("-A kapott pid: %d\n", pid);
    // lezárom a szülő READ deszkriptorát
    close(fd1[0]);
```

Majd a gyerek processz PID-jét küldöm el a szülőnek, a fentebbihez hasonló módon:

```
// pid lekérése
pid = getpid();

printf("-Gyerek pid kuldese a szulonek %d\n", pid);
// elküldöm a pid-et a szülő processznek
write(fd2[1], &pid, sizeof(pid));
// lezárom a gyerek WRITE deszkriptorát
close(fd2[1]);
```

A szülő processzben fogadom a küldött értéket, végül megvárom a gyerek processz befejeződését.

```
// beolvasom a pid-et a csővezetékről
read(fd2[0], &pid, sizeof(pid));
// kiírom
printf("A kapott pid: %d\n", pid);
// lezárom a gyerek READ deszkriptorát
close(fd2[1]);

// gyerek processz megvárása
wait(&child);
```

A teljes program:

```c
/*L9. Irjon C nyelvu programot, ami
letrehoz ket csovezeteket (ket file deszkriptor part)
elforkol
a szulo elkuldi a sajat pidjet a gyerkmeknek az egyik csovon
a gyermek kiirja a kepernyore es visszkuldi egy az ovet a masik csovon
megszunnek a processzek (a szulo megvarja a gyereket)*/

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

int main() {
    // file descriptorok
    int fd1[2]; // ebbe ír a szülő
    int fd2[2]; // ebbe ír a gyerek

    int pid = 0; // a pid értékének tárolására
    int child;   // a fork() hívás eredményének tárolására

    // csővezetékek létrehozása
    // hiba esetén kilép
    if (pipe(fd1) || pipe(fd2)) {
        perror("Hiba a pipe letrehozasakor");
        return 1;
    }

    // Gyerek processz létrehozása a fork() paranccsal
    if ((child = fork()) != 0) {
        // itt a szülő processzben vagyunk

        // pid lekérése
        pid = getpid();

        // lezárom a nem használt deszkriptorokat
        close(fd1[0]); // szülő READ
        close(fd2[1]); // gyerek WRITE

        printf("Szulo pid kuldese a gyereknek: %d\n", pid);
        // elküldöm a pid-et a gyerek processznek
        write(fd1[1], &pid, sizeof(pid));
        // lezárom a szülő WRITE deszkriptorát
        close(fd1[1]);

        // beolvasom a pid-et a csővezetékről
        read(fd2[0], &pid, sizeof(pid));
        // kiírom
        printf("A kapott pid: %d\n", pid);
        // lezárom a gyerek READ deszkriptorát
```

```
        // gyerek processz megvárása
        wait(&child);
    }
    else if (child == 0) {
        // itt a gyerek processzben vagyunk

        // lezárom a nem használt deszkriptorokat
        close(fd1[1]); // szülő WRITE
        close(fd2[0]); // gyerek READ

        // beolvasom a pid-et a csővezetékről
        read(fd1[0], &pid, sizeof(pid));
        // kiírom
        printf("-A kapott pid: %d\n", pid);
        // lezárom a szülő READ deszkriptorát
        close(fd1[0]);

        // pid lekérése
        pid = getpid();

        printf("-Gyerek pid kuldese a szulonek %d\n", pid);
        // elküldöm a pid-et a szülő processznek
        write(fd2[1], &pid, sizeof(pid));
        // lezárom a gyerek WRITE deszkriptorát
        close(fd2[1]);
    }

    return 0;
}
```

A program futásának eredménye

```
[arvaid@x550 beadando]$ ./feladat
Szulo pid kuldese a gyereknek: 4957
-A kapott pid: 4957
-Gyerek pid kuldese a szulonek 4958
A kapott pid: 4958
[arvaid@x550 beadando]$
```