JEGYZŐKÖNYV

Operációs rendszerek BSc

2021. tavasz féléves feladat

Készítette: **Kovács Krisztián**

Neptunkód: **WIQPM2**

A feladat leírása:

Írjon C nyelvű programot, ami létrehoz egy egy csővezetéket (egy file deszkriptor part), elforkol. A gyermek egy signal hatására elküld egy rövid szöveget a szülőnek (a signaling blokkol). A szülő kiírja, megszünnek a processzek (a szülő megvárja a gyereket).

A feladat elkészítésének lépései:
A forráskódban majdnem minden sorhoz van magyarázat.

```
int pfd[2]; //Pipe változó, a pfd[0]-ba írunk majd, a pfd[1]-ből olvasunk.
pid_t child; //Gyermekprocessz azonosítója.
```

Létrehozom a pipe változót, a pfd[0] az írásra szolgál majd, a pfd[1] pedig kiolvasásra.

```
if((pipe(pfd)) < 0 ) {
    perror("pipe");
    return 1;
}
```

Létrehozom a csővezetéket, ha sikertelen, azaz negatív számmal tér vissza (-1 ha jól emlékszem) , ekkor kiírom a hibát és a program visszatér egyel.

```
if((child=fork()) < 0) {
    perror("fork");
    return 1;
}
```

Létrehozom a gyermekprocesszt, ha sikertelen, akkor kiírom a hibát és a program visszatér egyel.

```
void send(); // Ez fogja lekezelni a signal-t.
```

Ez a függvény lessz a signal kezelője.

```
signal(SIGINT, send); //A SIGINT jelet(CTRL + C) a send fogja kezelni.
```

A SIGINT jelet (CTRL + C) a send fogja kezelni.

```
void send(){
    signal(SIGINT, SIG_DFL);
}
```

A feladat nem kérte ezt, de a kezelő meghívása után visszaállítom az alapértelmezett kezelőt.

```c
if (child == 0) { //Ha a gyermekprocessz fut
    printf("Varom a  CTRL + C szignalt.\n");
    pause(); //Signal-t vár.
    printf("\nA szignal megerkezett.\n");
    close(pfd[0]); //Lezárjuk a felesleges vezetéket, mivel itt csak írni fogunk.
    write(pfd[1], "Kovacs Krisztian WIQPM2\n" , 23); //Beleírjuk az adatot.
    close(pfd[1]); //Lezárjuk.
    exit(0); //Kilépünk a child processzből.
}
```

A gyermekprocessz fut, kiírom, hogy várom a signalt, és míg nem kap a program signal, addig nem megy tovább. Amikor megkapja, kiírom hogy megérkezett a signal, folytatódik a futás. Először is lezárom a fölösleges csővezetéket, hiszen itt csak írni fogunk. Beírom a pipe-ba “Kovács Krisztián WIQPM\n”, ami 23 byte. Lezárjuk, kilépünk a child processzből.

```c
else if (child > 0) { //Ha a szülőprocessz fut
    int returnStatus;
    waitpid(child, &returnStatus, 0); //Szülőprocessz m
    char s[1024]; //Ebbe tároljuk az adatot.
    close(pfd[1]); //Lezárjuk a felesleges vezetéket, m
    printf("Az uzenet:\n");
    read(pfd[0], s, sizeof(s)); //Kiolvassuk az adatot.
    printf("%s", s); //Kiírjuk a szabványos kimenetre.
    close(pfd[0]); //Lezárjuk.
    exit(0); //Kilépünk a parent processzből.
}
```

A szülő processz fut, megvárja míg lefut a gyermekprocessz, akár azt is meg lehet vizsgálni, hogy normal visszatérési értéke van-e a gyermekprocessznek. Létrehozom a karaktertömböt, amiben tárolom majd az üzenetet. Lezárom a pfd[1]-et, mivel cask olvasunk. A read segítségével kiolvassuk a pipe-ból az adatot az s tömbbe 1024 byte-ot, de ezt át lehetne írni 23-ra, mivel itt fix a mérete az üzenetnek. Kiírum a kapott adatot, lezárom a pipe-ot, leáll a parent processz.

A futtatás eredménye:

```
Varom a  CTRL + C szignalt.
^C
A szignal megerkezett.
Az uzenet:
Kovacs Krisztian WIQPM2
Process returned -1 (0xFFFFFFFF)   execution time : 1.702 s
Press ENTER to continue.
```