JEGYZŐKÖNYV

Operációs rendszerek BSc

2021 tavasz féléves feladat

Készítette: **Urbán Milán**

Neptunkód: **T6PXGV**

A feladat leírása:

Írjon C nyelvű programokat, ami hozzon létre egy osztott memória szegmenset a felhasználótól olvasson be szöveget, es ezt írja be az osztott memória területére es küldjön szignált a fogadó felnek, hogy kész az üzenet (SIGUSR1) (segítségképpen a másik program pid-je fixen beletelhető a programba) a másik program pedig olvass ki az osztott memória szegmensből, de csak egy adott súgnál hatására (SIGUSR1) végül szüntesse meg az shm szegmenst.

A feladat elkészítésének lépései:

bead01.c

1. Osztott memória

Első lépésként kulcsot deklarálunk az osztott memóriához, majd ezt követően létrehozzuk azt. Megadjuk a kulcsot, méretét, a hozzáférési jogosultságot és a létrehozás módját. Mindezt kiegészítjük egy error kezeléssel.

2. Pid kiolvasása

Megnyitjuk a „myfifo” fájlunkat, majd kiolvassuk belőle a „pid”-et. Ezt követően bezárjuk azt, majd kiírjuk a kiolvasott értéket

3. Műveletek az osztott memóriában

Csatlakozunk az osztott memóriához és hasonlóan a létrehozásnál, ide is beiktatunk egy error kezelést. Megadjuk az inputot, majd ezt követően átmásoljuk azt az osztott memóriába. Mindezek után lecsatlakozunk róla, elküldünk egy signált, majd kilépünk ebből a programrészből.

bead00.c

1. Main függvény

Kiírjuk a pidet, majd létrehozzuk a „myfifo” fájlunkat, hasonlóan az eddigiekhez hibakezeléssel. Ezt követően megnyitjuk azt és a pid-ünket beleírjuk és bezárjuk a fájt. Itt meghívjuk a signál kezelő modult és várunk a signálra.

2. Signál lekérdező modul

Megadjuk a közös memória kulcsát, majd error kezelés mellett rácsatlakozunk arra, és kiíratjuk az üzenetet és kilépünk a memóriából.

A bead00.c végleges formája:

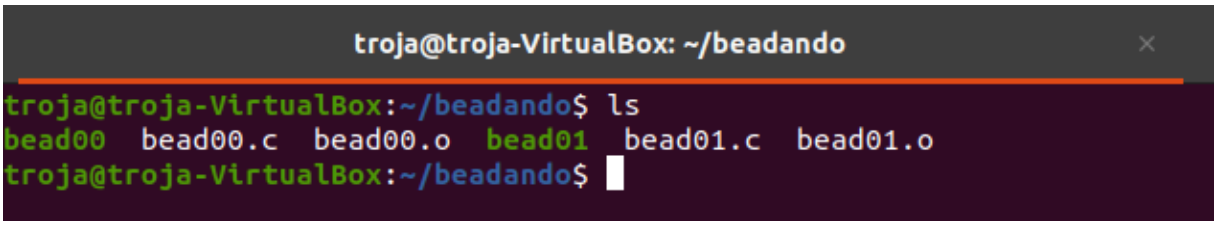
```c
#include<stdio.h>
#include<unistd.h>
#include<signal.h>
#include<stdlib.h>
#include<fcntl.h>
#include<sys/stat.h>
#include<sys/wait.h>
#include<sys/shm.h>
#include<sys/ipc.h>
#include<sys/types.h>


int main(int argc, char *argv[])
{
    key_t key= 111;
    int shmid;
        char *guess_mem, *userinput;
        if(( shmid = shmget(key,1024,0666|IPC_CREAT)) < 0 ){
                perror("shmget");
                exit(1);
        }

     int f,pid;
         f = open("myfifo", O_RDONLY);

         read(f, &pid, sizeof(int));
         close(f);
         unlink("myfifo");

     printf("read pid: %d", pid);

     size_t len = 0;
     ssize_t lineSize = 0;

     if ((guess_mem = shmat(shmid,NULL,0)) == (char *) -1){
                perror("shmat");
                exit(1);
         }
     lineSize = getline(&userinput, &len, stdin);
     strcpy(guess_mem, userinput);
     shmdt(guess_mem);
     kill(pid, SIGUSR1);
     exit(0);
     return 0;
}
```

A bead01.c végleges formája:

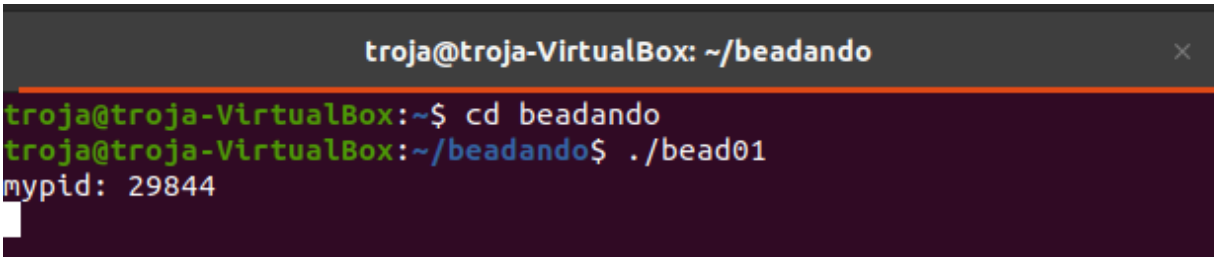
```c
#include<stdio.h>
#include<unistd.h>
#include<signal.h>
#include<stdlib.h>
#include<fcntl.h>
#include<sys/stat.h>
#include<sys/wait.h>
#include<sys/shm.h>
#include<sys/ipc.h>
#include<sys/types.h>

void handle_sigusr1(int signum){
//      signal(SIGUSR1, handle_sigusr1);
//      printf("anyad");

        key_t key= 111;
        int shmid;
        char *guess_mem;
        if ((shmid = shmget(key,1024, 0666 | IPC_CREAT)) < 0){
                perror("shmget");
                exit(1);
        }
        if((guess_mem = shmat(shmid,NULL,0)) == (char *) -1){
                perror("shmat");
                exit(1);
        }

          printf("message: %s \n", guess_mem);
          shmdt(guess_mem);
          exit(0);
          //shmctl(shmid,P_PGID,NULL);
}

int main(int argc, char *argv[])
{

        int f, mypid = getpid();
        printf("mypid: %d\n",mypid);
        if(mkfifo("myfifo", 0666) < 0){
                perror("mkfifo"), exit(EXIT_FAILURE);
        }

        f=open("myfifo", O_WRONLY);
        write(f, &mypid, sizeof(int));
        close(f);

        signal(SIGUSR1, handle_sigusr1);
        while(1); //várunk a signálra
        return 0;
}

```

A futtatás eredménye:

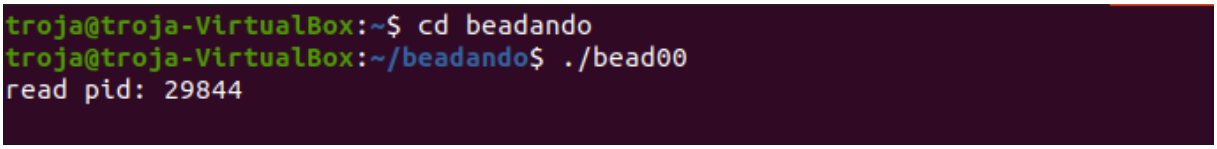
A hibakeresés és futtatást elvégzéséhez szükséges 2 db terminál. Miután ezeket megnyitottuk, elnavigálunk a project mappájába.



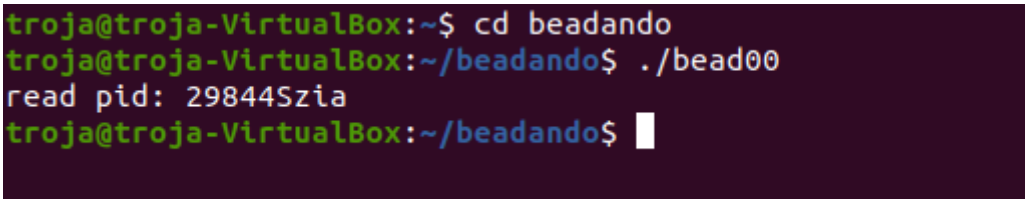
Futtatjuk bead01 -et, ő veszi majd az üzenetet.



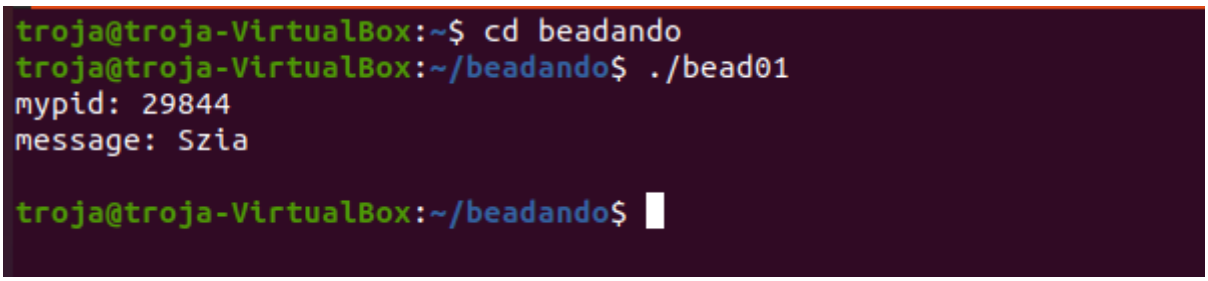
Ezután a második terminálba futtatjuk az adót, a bead00-át.



Majd beírjuk az üzenetet.



Ezután visszalépünk az első terminálba és látjuk az üzenetet.



A programunk megfelelően működik.