JEGYZŐKÖNYV

Operációs rendszerek BSc

2022. tavasz féléves feladat

Készítette: **Danyi Kristóf Milán**

Neptunkód: **GQOKMW**

A feladat leírása:

4. Adott az alábbi terhelés esetén a rendszer. Határozza meg az *indulás, befejezés, várakozás/átlagos várakozás és körülfordulás/átlagos körülfordulás, válasz/átlagos válaszidő* és a *CPU kihasználtság* értékeket az SJF ütemezési algoritmusok mellett! (cs: 0,1ms; sch: 0,1ms)

	P1	P2	P3	P4
Érkezés	0	8	12	20
CPU idő	15	7	26	10
Indulás				
Befejezés				
Várakozás				

4. 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)

A feladat elkészítésének lépései: Az excelben egyyszerű számolásokat végeztem, a C programban pedig úgy érzem, hogy a forráskódot megfelelő mértékben kommenteltem, így felhasználom a feladat lépéseinek magyarázataként

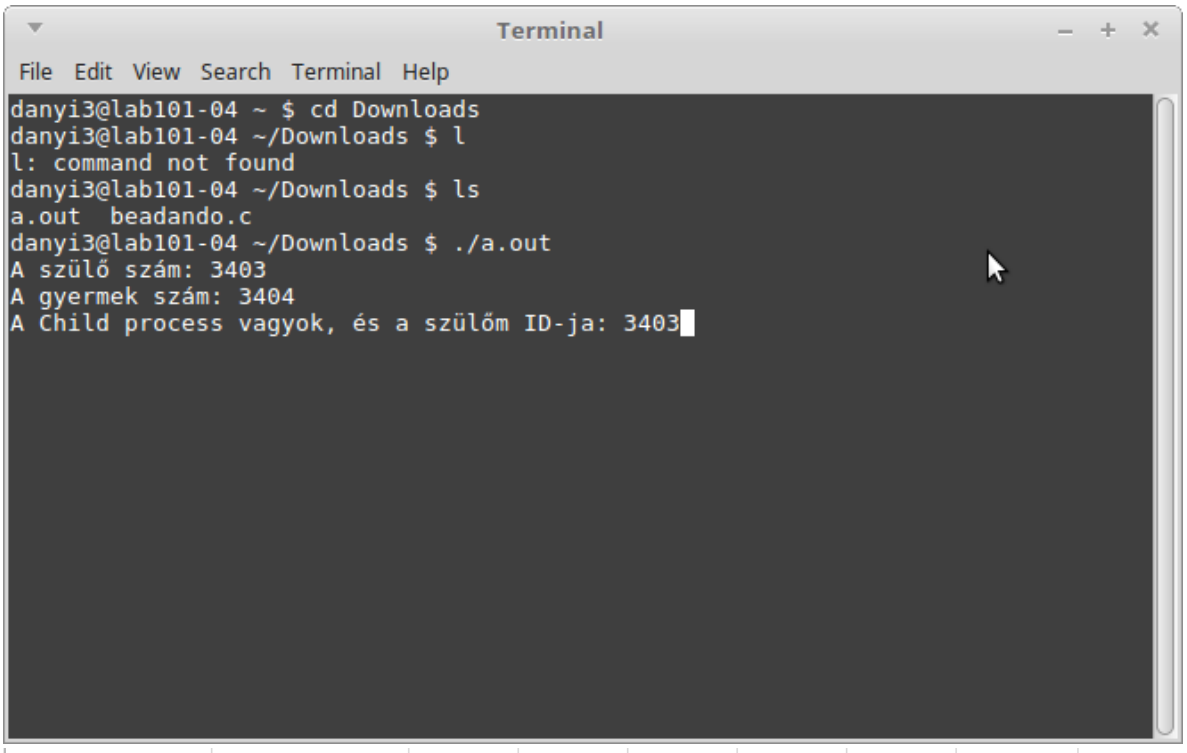
*beadando.c (~/Downloads)

File Edit View Search Tools Documents Help

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#define SIZE 256

int main()
{
    int fd1[2];                         //Létrehozzuk az első csövet
    int fd2[2];                         //Létrehozzuk a második csövet
    int parentmsg;                      //Ide mentjük a processz ID-t
    int childmsg;

    if(pipe(fd1) < 0){                  // Megpipeoljuk az fd1 és fd2-t
        exit(1);
    }
    if(pipe(fd2) < 0){
        exit(1);
    }

    pid_t pid = fork();                 //létrehozzuk a child process


    if(pid != 0){
    //szülő részleg
        pid_t Ppid = getpid();          //Eltároljuk a szülő Pid-jét
        printf("A szülő szám: %d\n",Ppid);
        write(fd1[1], &Ppid, SIZE);     //Beleírjuk az első csőbe a szülő pid-et
        wait(NULL);                     //Megvárjuk még a gyermek process megcsinálja a dolgait
        read(fd2[0],&childmsg,SIZE);    //Kiolvassuk a második csőből a gyermek pid-et
        kill(childmsg,SIGTERM);         //Megöljük a gyermeket
        kill(Ppid,SIGTERM);             // Utána Megöljük a szülőt



    }
    else if (pid ==0){
    //gyermek részleg
        pid_t Cpid = getpid();          //Eltároljuk a gyemerk pid-et
        printf("A gyermek szám: %d\n",Cpid);
        read(fd1[0],&parentmsg,SIZE);   //Kiolvassuk a szülő pid-et az első csőből
        write(fd2[1],Cpid, SIZE);       //Beleírjuk a második csőbe a gyermek pid-et
        printf("A Child process vagyok, és a szülőm ID-ja: %d",parentmsg); //Kiirattatjuk a gyermekkel a szülő pidet

    }
return 0;
}
```

C Tab Width: 4 Ln 31, Col 85 INS

A futtatás eredménye:



Danyi Kristóf Milán GQOKMW							
		1.Munka	2.Munka	4.Munka	3.Munka		
	SJF	P1	P2	P3	P4		
	Érkezés	0	8	12	20		
	CPU idő	15	7	26	10		
	Indulás	0	15	32	22		
	Befejezés	15	22	58	32	Várakozás:	29
	Várakozás	0	7	20	2	Átlagos:	7,25
sch:0,1ms	Körülfordulási idő	15	14	46	12		
cs:0,1ms	Átlagos:	21,75				Válaszidő:	29
Váltások:	0,8 ms					Átlagos:	7,25
CPU kihasználtság:	99%