JEGYZŐKÖNYV

Operációs rendszerek BSc

2022. tavaszi féléves feladat

Készítette: **Pogonyi Ábel Kürt**

Neptunkód: **TR6FKP**

1. feladat

Írjon egy olyan C programot, mely egy fájlból számpárokat kiolvasva meghatározza a legnagyobb közös osztóját. A feladat megoldása során használjon message queue(üzenetsoros) IPC mechanizmust, valamint a kimenet kerüljön egy másik fájlba.
A kimeneti fájl struktúrája kötött!

Példa a bemeneti és kimeneti fájl struktúrájára:

Bemeneti fájl:

i (Ez jelzi a számpárok darabszámát)

x y

Kimeneti fájl (Az x, y jelzi a bemeneti adatokat a z pedig a kimeneti eredményt)

x y z

Megoldás:

A feladat 3 lépésből áll. Elsőnek fel kell dolgozni a bemeneti fájlt, majd az adatokból ki kell számolni az eredményt, azután kiírni az eredményeket egy külön fájlba.

Az input fájlból egy kétdimenziós string tömbbe rakom az adatokat, amit ezután megformázok, hogy az adatait majd intekké lehessen alakítani.

Létrehozok egy struktúrát, ami a memória szegmens struktúrája lesz. Létrehozom a memória szegmenset, benne a struktúrát és ide feltöltöm az adatokat, itt már int típusban tárolódnak.

Ezután létrejön egy gyermek processz, ami kiolvassa a szegmens tartalmát majd egy algoritmussal kiszámolja a legnagyobb közös osztót, és az eredményeket egy másik tömbbe írja a memória szegmensben, ezután terminálódik.

A fő program bevárja a gyerek processzt, majd kiolvassa a megoldott tömböt és kiírja egy file-ba a megoldást és végül felszabadítja a memóriát és leáll.

A program csak Linux alatt fut.

Képek:

futás eredménye:

```
┌──(kali㉿kali)-[~/TR6FKPOsGyak/OSSemTask_tr6fkp]
└─$ gcc lnko.c -o lnko

┌──(kali㉿kali)-[~/TR6FKPOsGyak/OSSemTask_tr6fkp]
└─$ ./lnko
Input fájl megnyitása sikeres!
Az osztott memóriaszegmens létrejött! (ID: 32807)
A memóriaszegmensre csatlakozás sikeres!
CHILD: Gyermekprocessz létrejött! (PID: 0)
CHILD: A memóriaszegmensre csatlakozás sikeres!
CHILD: lnko(32, 120) = 8
CHILD: lnko(42, 654) = 6
CHILD: lnko(21, 217) = 7
CHILD: lnko(35, 245) = 35
CHILD: lnko(46, 230) = 46
A 1465 PID-jű gyermekprocessz megszűnt!
Az 32807 ID-jű memóriaszegmens törölve!
Output fájl megnyitása sikeres!
Tartalom kiírva output.txt fájlba! A fájlok bezárásra kerülnek, majd a program leáll!
```

input fájl:

input.txt

```
1 5
2 32 120
3 42 654
4 21 217
5 35 245
6 46 230
```

output fájl:

```
1 32 120 8
2 42 654 6
3 21 217 7
4 35 245 35
5 46 230 46
6
```

2. feladat

10. Adott négy processz (A, B, C, D) a rendszerbe, induláskor a p_cpu értéke A=0, B=6, C=0, D=0. A rendszerben a P_USER = 60

Induláskor a p_usrpri A=60, B=65, C=60 és D=60.

Ha egy processz megkapja a CPU-t a quantum-ában végig használja (azaz 1 quantum-ban), a p_cpu növekmény értéke 70. Mind a négy processznél a p_nice érték 0.

Határozza meg öt quantum-ban hogyan változnak a prioritások és a p_cpu, melyik processz, milyen sorrendben kap CPU-t.

Igazolja az ütemező algoritmus leírásával, képlettel és számítással az eredményeket.

A feladat Excel programban készült el.

Quantum: 70	A process		B process		C process		D process		Reschedule	
Clock tick	p_uspri	p_cpu	p_uspri	p_cpu	p_uspri	p_cpu	p_uspri	p_cpu	running before	running after
Starting point	60	0	65	6	60	0	60	0		A
1	60	1	65	6	60	0	60	0	A	A
2	60	2	65	6	60	0	60	0	A	A
69	60	69	65	6	60	0	60	0	A	A
70	75	60	61	5	60	0	60	0	A	C
71	75	60	61	5	60	1	60	0	C	C
139	75	60	61	5	60	69	60	0	C	C
140	73	51	61	4	75	60	60	0	C	D
141	73	51	61	4	75	60	60	1	D	D
209	73	51	61	4	75	60	60	69	D	D
210	71	44	61	3	73	51	75	60	D	B
211	71	44	61	4	73	51	75	60	B	B
279	71	44	61	72	73	51	75	60	B	B
280	70	38	76	63	71	44	73	51	B	A
281	70	39	76	63	71	44	73	51	A	A
349	70	107	76	63	71	44	73	51	A	A
350	83	93	74	54	70	38	71	44	A	C