

Operációs rendszerek BSc

10. Gyak.

2022. 04. 13.

Készítette:

Pázmán András Bsc

Mérnökinformatikus

H2Z4X3

Miskolc, 2022

Operációs rendszerek –10. Gyakorlat

Bankár algoritmus, IPC mechanizmus

Töltse fel az aktuális mappába: **Neptunkod_....**

Jegyzőkönyv neve: *gyak10.pdf*

Forrás fájlok:

A futás eredményét is tartalmazza a jegyzőkönyv.

Határidő: aktuális gyakorlat időpontja, ill. módosítás esetén 2022.04.24.

Irodalom

Tanulmányozzák a Vadász Dénes: Operációs rendszerek, 2006. ME, jegyzet, ill. Vincze

Dávid: Operációs rendszerek - diasort.

Szintén tanulmányozzák az előadáson kivetített URL linkhez tartozó irodalmat, majd oldják meg a feladatot.

Feladatok

„1. Az előadáson bemutatott mintaprogram alapján készítse el a következő feladatot.

Adott egy rendszerbe az alábbi erőforrások: R (R1: 10; R2: 5; R3: 7)

A rendszerbe 5 processz van: P0, P1, P2, P3, P4

Kérdés: Kielégíthető-e P1 (1,0,2), P4 (3,3,0) ill. P0 (0,2,0) kérése úgy, hogy biztonságos legyen, holtpontmentesség szempontjából a rendszer - a következő *kiinduló állapot* alapján.

Külön-külön táblázatba oldja meg a feladatot!

- a) Határozza meg a processzek *által igényelt erőforrások mátrixát*?
- b) Határozza meg *pillanatnyilag szabad erőforrások számát*?
- c) Igazolja, magyarázza az egyes *processzek végrehajtásának lehetséges sorrendjét - számolással?*”

Az összes osztály -erőforrások száma: (10, 5, 7)							
Kiinduló állapot							
	1. lépés				2. lépés		
	MAX. IGÉNY				FOGLAL		
	R1	R2	R3		R1	R2	R3
P0	7	5	3		0	1	0
P1	3	2	2		2	0	0
P2	9	0	2		3	0	2
P3	2	2	2		2	1	1
P4	4	3	3		0	0	2

$P1(1, 0, 2)$

$R1: 10 ; R2: 5 ; R3: 7$

	MAX IGENY				FOGLAL				IGENY (MAX. - FOGLAL)			
	R1	R2	R3		R1	R2	R3		R1	R2	R3	
P0	7	5	3		0	1	0		7	4	3	T. 5.
P1	3	2	2		$2+1=3$	$0+0=0$	$0+2=2$		0	2	0	T. 1.
P2	9	0	2		3	0	2		6	0	0	T. 4.
P3	2	2	2		2	1	1		0	1	1	T. 2.
P4	4	3	3		0	0	2		4	3	1	T. 3.
FOGLAL:					8	2	7					
ERŐFORRÁS:					10	5	7					
KÉSZLET:					2	3	0					

1. lépés (megvizsgáljuk melyik igény \leq készlet)

KÉSZLET: 2 3 0 Most a P1 kielégíthető \Rightarrow P1 foglalt felszabadul.
 foglalt P1: ~~3/0/2~~ (+)
 új KÉSZLET: 5 4 2

2. lépés (melyik igény \leq készlet)

KÉSZLET: 5 3 0 2 Most a P3 kielégíthető \Rightarrow P3 foglalt (free)
 foglalt P3: ~~2/1/1~~ (+)
 új KÉSZLET: 7 4 3

3. lépés (igény \leq készlet)

KÉSZLET: 7 4 3 Most P4 kielégíthető \Rightarrow P4 foglalt felszabadul.
 foglalt P4: ~~0/0/2~~ (+)
 új KÉSZLET: 7 4 5

4. lépés (igény \leq készlet?)

KÉSZLET: 7 4 5 Most P2 kielégíthető \Rightarrow P2 foglalt felszabadul.
 foglalt P2: ~~3/0/2~~ (+)
 új Készlet: 10 4 7

5. lépés (igény \leq készlet) KÉSZLET: 10 4 7 \Rightarrow P0 kielégíthető \Rightarrow P0 felszabadul.
 új készlet: 10 5 7.

Lehetséges végrehajtási sorrend: $P1 \rightarrow P3 \rightarrow P4 \rightarrow P2 \rightarrow P0$

$P_4: (3, 3, 0)$

Tráfoforrások: $R_1: 10; R_2: 5; R_3: 7$.

	MAX IGÉNY				FOGLAL				IGÉNY (MAX IGÉNY - FOGLAL)		
	R_1	R_2	R_3		R_1	R_2	R_3		R_1	R_2	R_3
P_0	7	5	3		0	1	0		7	4	3
P_1	3	2	2		2	0	0		1	2	2
P_2	9	0	2		3	0	2		6	0	0
P_3	2	2	2		2	1	1		0	1	1
P_4	4	3	3		4-3=1	3-3=0	2-0=2		1	0	1

FOGLAL: 10 5 3
 ERŐFORRÁS: 10 5 7
KÉSZLET: 0 0 4

1. lépés (megigény < készlet?)

KÉSZLET 0 0 4

{ A P_4 nem teljesíthető mivel az igény nagyobb mint a készlet;
 A rendszer nem lesz biztonságos állapotban. }

P0(0, 0, 0)

R1:10 R2:5 R3:7

MAX igény

Foglal

igény

	R1	R2	R3		R1	R2	R3		R1	R2	R3	
P0	7	5	3		0+0=0	1+2=3	0+0=0		7	2	3	T 4.
P1	3	2	2		2	0	0		1	2	0	T 1.
P2	9	0	2		3	0	2		6	0	0	T 4.
P3	2	2	2		2	1	1		0	1	1	T 3.
P4	4	3	3		0	0	2		4	3	1	T 2.

Foglal 7 4 5
 Előfordulás 10 5 7
 Készlet 3 4 2

1. lépés (melyik igény \leq készlet)

Készlet: 3 4 2 | P1 kielégíthető \Rightarrow P1 foglal felszabadul.
 P1 foglal ~~2/0/0~~ \oplus
 Új Kész. 5 4 2

2. lépés (melyik igény \leq készlet)

Készlet: 5 4 2 | P4 kielégíthető \Rightarrow P4 foglal felszabadul.
 P4 foglal ~~0/0/2~~ \oplus
 Új Kész. 5 4 4

3. lépés (igény \leq készlet?)

Készlet: 5 4 4 | P3 kielégíthető \Rightarrow P3 foglal felszabadul.
 P3 foglal ~~2/1/1~~ \oplus
 Új Kész. 7 5 5

4. lépés (igény \leq készlet?)

Készlet 7 5 5 | P2 kielégíthető \Rightarrow P2 foglal felszabadul.
 P2 foglal ~~3/0/2~~ \oplus
 Új Kész. 10 5 7

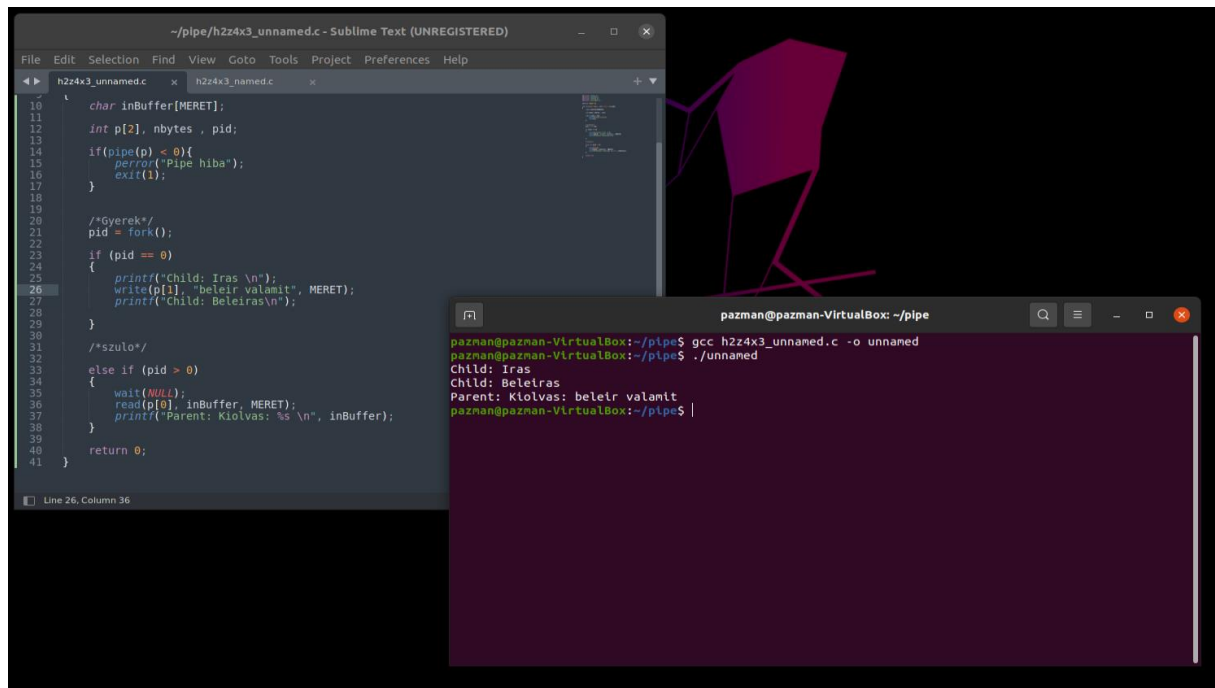
5. lépés (igény \leq készlet)

Készlet 10 5 7 | P0 kielégíthető \Rightarrow P0 foglal felsz.
 Új Kész. 10 8 7

Lehetséges sorrend: P1 \rightarrow P4 \rightarrow P3 \rightarrow P2 \rightarrow P0

2. Készítsen C nyelvű programot, ahol egy szülő processz létrehoz egy csővezetékot, a gyerek processz beleír egy szöveget a csővezetékbe (A kiírt szöveg: XY neptunkod), a szülő processz ezt kiolvassa, és kiírja a standard kimenetre.

Mentés: neptunkod_unnamed.c



The screenshot shows a Sublime Text editor window titled `~/pipe/h2z4x3_unnamed.c - Sublime Text (UNREGISTERED)` with the following C code:

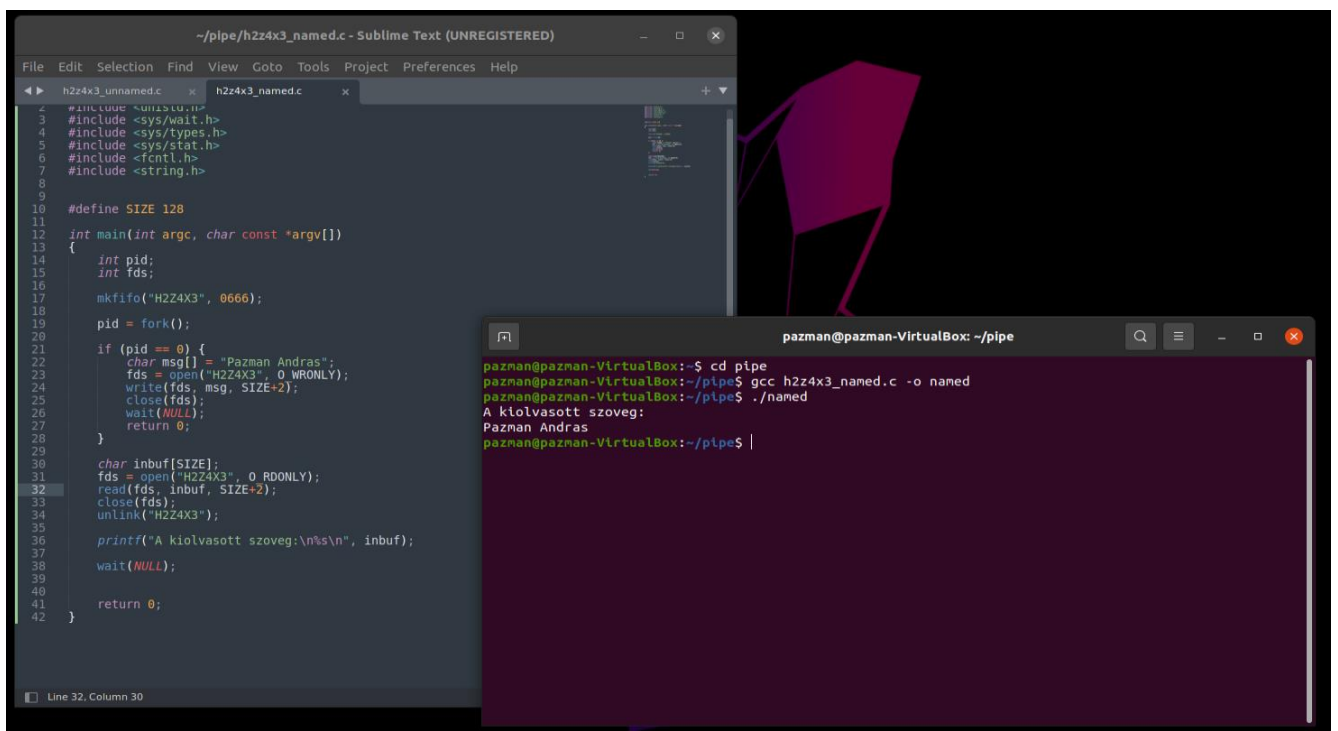
```
10 char inBuffer[MERET];
11
12 int p[2], nbytes, pid;
13
14 if(pipe(p) < 0){
15     perror("Pipe hiba");
16     exit(1);
17 }
18
19 /*Gyerek*/
20 pid = fork();
21
22 if (pid == 0)
23 {
24     printf("Child: Iras \n");
25     write(p[1], "beleir valamit", MERET);
26     printf("Child: Beleiras\n");
27 }
28
29 /*szulo*/
30
31 else if (pid > 0)
32 {
33     wait(NULL);
34     read(p[0], inBuffer, MERET);
35     printf("Parent: Kiolvas: %s \n", inBuffer);
36 }
37
38 return 0;
39
40 }
41
```

Below the editor is a terminal window titled `pazman@pazman-VirtualBox: ~/pipe` showing the execution of the program:

```
pazman@pazman-VirtualBox:~/pipe$ gcc h2z4x3_unnamed.c -o unnamed
pazman@pazman-VirtualBox:~/pipe$ ./unnamed
Child: Iras
Child: Beleiras
Parent: Kiolvas: beleir valamit
pazman@pazman-VirtualBox:~/pipe$
```

3. Készítsen C nyelvű programot, ahol egy szülő processz létrehoz egy nevesített csővezetékot (neve: neptunkod), a gyerek processz beleír egy szöveget a csővezetékbe (A hallgató neve: pl.: Keserű Ottó), a szülő processz ezt kiolvassa, és kiírja a standard kimenetre.

Mentés: neptunkod_named.c



The screenshot shows a Sublime Text editor window titled `~/pipe/h2z4x3_named.c - Sublime Text (UNREGISTERED)` with the following C code:

```
1 #include <unistd.h>
2 #include <sys/wait.h>
3 #include <sys/types.h>
4 #include <sys/stat.h>
5 #include <fcntl.h>
6 #include <string.h>
7
8
9 #define SIZE 128
10
11 int main(int argc, char const *argv[])
12 {
13     int pid;
14     int fds;
15
16     mkfifo("H2Z4X3", 0666);
17
18     pid = fork();
19
20     if (pid == 0) {
21         char msg[] = "Pazman Andras";
22         fds = open("H2Z4X3", O_WRONLY);
23         write(fds, msg, SIZE+2);
24         close(fds);
25         wait(NULL);
26         return 0;
27     }
28
29     char inbuf[SIZE];
30     fds = open("H2Z4X3", O_RDONLY);
31     read(fds, inbuf, SIZE+2);
32     close(fds);
33     unlink("H2Z4X3");
34
35     printf("A kiolvasott szoveg:\n%s\n", inbuf);
36
37     wait(NULL);
38
39     return 0;
40 }
41
42
```

Below the editor is a terminal window titled `pazman@pazman-VirtualBox: ~/pipe` showing the execution of the program:

```
pazman@pazman-VirtualBox:~$ cd pipe
pazman@pazman-VirtualBox:~/pipe$ gcc h2z4x3_named.c -o named
pazman@pazman-VirtualBox:~/pipe$ ./named
A kiolvasott szoveg:
Pazman Andras
pazman@pazman-VirtualBox:~/pipe$
```