

holtpontmentesség szempontjából a rendszer - a következő kiinduló állapot alapján. Igazolja a processzek végrehajtásának sorrendjét – számolással.”

Az összes osztály -erőforrások száma: (10,5,7)									
1.Lépés			2. Lépés			3.Lépés			
		Max. igény		Foglal			MAX-I - Fogla	igény	
P0	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	7	5	3	P0	0	0	P0	7	4
P2	3	2	2	P1	2	0	P1	1	2
P3	9	0	2	P2	0	2	P2	6	0
P4	2	2	2	P3	2	1	P3	0	1
	4	3	3	P4	0	2	P4	4	3
4. Lépés			5.Lépés			6.Lépés			
	Foglal+Öss	Készlet		P1	1	2	2	P1 kielégíthető	
	R1	R2	R3	Készlet	3	3	2		
	10-7=3	5-2=3	7-5=2						
A szabad erőforrások száma (3,3,2)			7.Lépés			8.Lépés			
				Újra számolas	R1	R2	R3		
				P1 Foglal+ Erf	5	3	2		
					Készlet				
				9.Lépés			10.Lépés		
				Újra számolas	R1	R2	R3	Újra számolas	R1
				P3 Foglal+ Erf	7	4	3	P2 Foglal+ Erf	10
					Készlet				
				11.Lépés			12.Lépés		
				Újra számolas	R1	R2	R3	Újra számolas	R1
				P0 Foglal+ Erf	7	4	3	P4 Foglal+ Erf	6
					Készlet				
				13.Lépés			14.Lépés		
				Újra számolas	R1	R2	R3	Újra számolas	R1
				P4 Foglal+ Erf	7	4	3	P4 Foglal+ Erf	4
					Készlet				
				15.Lépés			16.Lépés		
				Újra számolas	R1	R2	R3	Újra számolas	R1
				P0 Foglal+ Erf	7	4	3	P4 Foglal+ Erf	4
					Készlet				

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	
Az összes osztály –erőforrások száma: (10,5,7)							Kérések + 2 P0 0,2,0 + P0 0,1,0			P4 3,3,0 + P4 0,0,2			Amennyiben a P4 ÉS P0 kérését is megnevelem készlet nem lesz megfelelő a lefutás biztosításához. Végrehajtási sorrend : Nem fut le mert a szabad erőforrások száma negatív lenne ami nem lehetséges.						
1.Lépés				2. Lépés				3.Lépés											
		Max. igény					Foglal				MAX-1 - Foglal	igény							
	R1	R2	R3				R1	R2	R3		R1	R2	R3						
P0	7	5	3				P0	0	3	0	P0	7	2	3					
P1	3	2	2				P1	2	0	0	P1	1	2	2					
P2	9	0	2				P2	3	0	2	P2	6	0	0					
P3	2	2	2				P3	2	1	1	P3	0	1	1					
P4	4	3	3				P4	3	2		P4	1	0	1					
4. Lépés				5.Lépés				6.Lépés											
	Össz-Foglal	Készlet					R1	R2	R3										
	R1	R2	R3				P0	7	2	3	Nincs megfelelő								
	10-10=0	5-7=-2	7-5=2				P1	1	2	2									
							P2	6	0	0									
							P3	0	1	1									
							P4	1	0	1									
A szabad erőforrások száma (0,-2,-) Nem lehet				Készlet				3				3				2			

Írjanak három C nyelvű programot, ahol készít egy üzenetsort és ebbe két üzenetet tesz bele – msgcreate.c, majd olvassa ki az üzenetet – msgrcv.c, majd szüntesse meg az üzenetsort

(takarít) - msgctl.c.

A futtatás eredményét is tartalmazza a jegyzőkönyv.

Mentés: msgcreate.c; msgrcv.c; msgctl.c.

2a. Írjon egy C nyelvű programot, melyben az egyik processz létrehozza az

üzenetsort, és szövegeket küld bele, exit üzenetre kilép,

másik processzben lehet választani a feladatok közül: üzenetek darabszámának

lekérdezése, 1 üzenet kiolvasása, összes üzenet kiolvasása, üzenetsor megszüntetése, kilépés.

Mentés: gyak10_2.c

A futtatás eredményét is tartalmazza a jegyzőkönyv.

3. Gyakorló feladat: Először tanulmányozzák Vadász Dénes: Operációs rendszer jegyzetét - a témához kapcsolódó fejezetét (5.3.2), azaz

Írjon három C nyelvű programot, ahol készít egy osztott memóriát, melyben

választott kulccsal kreál/azonosít osztott memória szegmenst - shmcreate.c.

az shmcreate.c készített osztott memória szegmens státusának lekérdezése –

shmctl.c opcionális: shmop.c shmid-del azonosít osztott memória szegmenst. Ezután

segm nevű pointervál-tozót használva a processz virtuális címtartományába

kapcsolja (attach) a szegmest (shmat() rendszerhívás). Olvassa, írja ezt a

címtartományt, végül lekapcsolja (detach) a shmdt() rendszerhívással).

3a. Írjon egy C nyelvű programot, melyben egyik processz létrehozza az osztott

memóriát, másik processz rácsatlakozik az osztott memóriára, ha van benne

valamilyen szöveg,

akkor kiolvassa, majd beleír új üzenetet, harmadik processznél lehet választani a feladatok közül: státus lekérése (szegmens

mérete, utolsó shmop-os proc. pid-je), osztott memória megszüntetése, kilépés (2. és 3.

proc. lehet egyben is)”

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#define MSGKEY 654311L
#define SIZE 2
```

```
struct msgbuf {
    long mtype;
    int mtext[SIZE+1];
} message, *msgPointer;
```

```
int main() {
    int mymsg;
    key_t mykey;
    int myflag;
    int myreturn, mysize;
    int i;

    mykey = MSGKEY;
    myflag = 00666 | IPC_CREAT;
    mymsg = msgget(mykey, myflag);
    if (mymsg == -1) {
        perror("Hiba!");
        exit(-1);
    }
    printf("Sikeres létrehozás: %d, %x\n", mymsg, mymsg);
    msgPointer = &message;
    msgPointer->mtype = 0;

    for (i = 1; i < SIZE + 1; i++) {
        printf("Szamot kerek: %d\n", i);
        scanf("%d", &(msgPointer->mtext[i]));
        printf("%d", msgPointer->mtext[i]);
    }
}
```

```

10 struct msgbuf1 {
11     long mtype;
12     int mtext[SIZE+1];
13 } message, *msgPointer;
14
15 int main() {
16     int mymsg;
17     key_t mykey;
18     int myflag;
19     int myreturn, mysize;
20     int i;
21
22     mykey = MSGKEY;
23     myflag = 00666 | IPC_CREAT;
24     mymsg = msgget(mykey, myflag);
25     if (mymsg == -1) {
26         perror("Hiba!");
27         exit(-1);
28     }
29     printf("Sikeres létrehozás: %d, %x\n", mymsg, mymsg);
30     msgPointer = &message;
31     msgPointer->mtype = 0;
32
33     for (i = 1; i < SIZE + 1; i++) {
34         printf("Szamot kerek: %d\n", i);
35         scanf("%d", &(msgPointer->mtext[i]));
36         printf("%d", msgPointer->mtext[i]);
37     }
38     mysize = sizeof(int) * (SIZE + 1);
39     myreturn = msgsnd(mymsg, (struct msgbuf *)msgPointer, mysize, myflag);
40     printf("Visszatérési érték: %d\n", myreturn);
41     printf("Üzenet: %s\n", msgPointer->mtext);
42
43     exit(0);
44 }
45

```

Futás eredménye:

```

kiss@kiss-VirtualBox:~/Asztal/ez/gyak10$ cd gyak10
kiss@kiss-VirtualBox:~/Asztal/ez/gyak10/gyak10$ ls
bin  gyak10.cbp  gyak10.layout  main.c  obj
kiss@kiss-VirtualBox:~/Asztal/ez/gyak10/gyak10$ gcc main.c -o main.out
main.c: In function 'main':
main.c:41:19: warning: format '%s' expects argument of type 'char *', but argume
nt 2 has type 'int *' [-Wformat=]
   41 |     printf("Üzenet: %s\n", msgPointer->mtext);
      |                   ^~
      |                   |
      |               char *      int *
      |               %ls
kiss@kiss-VirtualBox:~/Asztal/ez/gyak10/gyak10$ ls
bin  gyak10.cbp  gyak10.layout  main.c  main.out  obj
kiss@kiss-VirtualBox:~/Asztal/ez/gyak10/gyak10$ ./main.out
Sikeres létrehozás: 0, 0
Szamot kerek: 1

```