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

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

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

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

Igazolja a processzek végrehajtásának sorrendjét – számolással.

P4 (3,3,0)

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

	FOGLAL		
	R1	R2	R3
P0	0	1	0
P1	2	0	0
P2	3	0	2
P3	2	1	1
P4	3	3	2

	IGÉNY		
	R1	R2	R3
P0	7	4	3
P1	1	2	2
P2	6	0	0
P3	0	1	1
P4	1	0	1

R1 = 10-10=0
R2 = 5-5=0
R3 = 7-5=2

KÉSZLET (0, 0, 2)

Nem elégíthető ki a P4 kérése.

P0 (0,2,0)

	FOGLAL		
	R1	R2	R3
P0	0	3	0
P1	2	0	0
P2	3	0	2
P3	2	1	1
P4	0	0	2

	IGÉNY		
	R1	R2	R3
P0	7	2	3
P1	1	2	2
P2	6	0	0
P3	0	1	1
P4	4	3	1

R1 = 10-7=3
R2 = 5-4=1
R3 = 7-5=2

KÉSZLET (3, 1, 2)

A P3 processz igénye elégíthető ki.

Új KÉSZLET (5, 2, 3)

	FOGLAL		
	R1	R2	R3
P0	0	3	0
P1	2	0	0
P2	3	0	2
P4	0	0	2

	IGÉNY		
	R1	R2	R3
P0	7	2	3
P1	1	2	2
P2	6	0	0
P4	4	3	1

A P1 processz igénye elégíthető ki.

Új KÉSZLET (7, 2, 3)

	FOGLAL		
	R1	R2	R3
P0	0	3	0
P2	3	0	2
P4	0	0	2

	IGÉNY		
	R1	R2	R3
P0	7	2	3
P2	6	0	0
P4	4	3	1

A P2 processz igénye elégíthető ki.

Új KÉSZLET (10, 2, 5)

	FOGLAL		
	R1	R2	R3
P0	0	3	0
P4	0	0	2

	IGÉNY		
	R1	R2	R3
P0	7	2	3
P4	4	3	1

A P0 processz igénye elégíthető ki.

Új KÉSZLET (10, 5, 5)

	FOGLAL		
	R1	R2	R3
P4	0	0	2

	IGÉNY		
	R1	R2	R3
P4	4	3	1

A P4 processz igénye elégíthető ki.

Új KÉSZLET (10, 5, 7)

A P0 kérése teljesíthető: a rendszer biztonságos állapotban van.

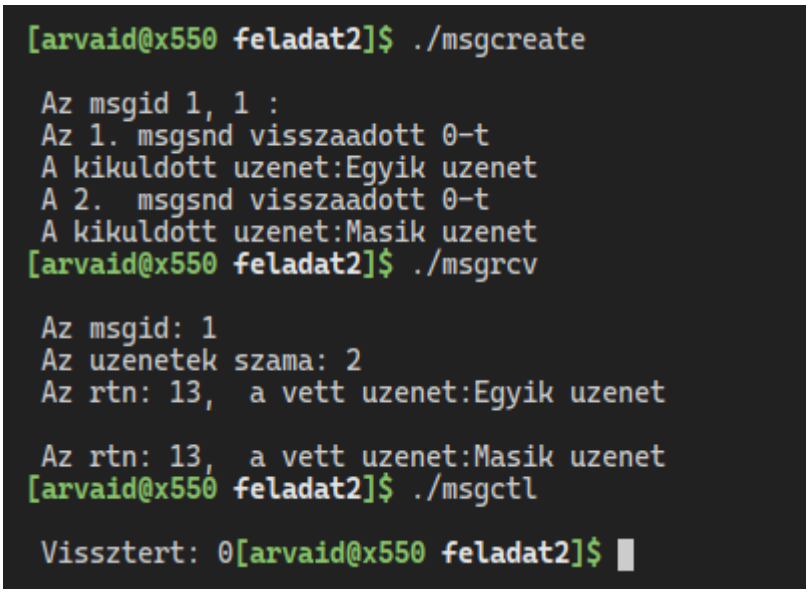
A processzek végrehajtásának sorrendje: P3 – P1 – P2 – P0 – P4.

2. Gyakorló feladat: Először tanulmányozzák Vadász Dénes: Operációs rendszer jegyzet, a témához kapcsolódó fejezetét (5.3)., azaz

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.

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

Futtatás eredménye:



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

gyak10_2.c:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#define MSGKEY 654321L

struct msgbuf1 {
                        long mtype;
                        char mtext[512];
} msgbuf, *msgp;          /* message buffer es pointere */

struct msqid_ds ds, *buf; /* uzenetsorhoz asszocialt struktura
                                                es pointere*/

int main() {
    int child = 0;

    int msgid;
    key_t key = MSGKEY;
    int msgflg;
    int rtn, msgsz;

    if ((child = fork()) == 0) {
        msgflg = 00666 | IPC_CREAT;
        msgid = msgget(key, msgflg);
        if (msgid == -1) {
            perror("\n The msgget system call failed!");
            return -1;
        }
        msgp = &msgbuf;
        msgp->mtype = 1; // text

        strcpy(msgp->mtext, "uzenet");
        msgsz = strlen(msgp->mtext) + 1;
        rtn = msgsnd(msgid, (struct msguf *) msgp, msgsz, msgflg);

        return 0;

    }
    else {
        int mtype;

        msgflg = 00666 | IPC_CREAT | MSG_NOERROR;
        msgid = msgget(key, msgflg);
        if (msgid == -1) {
            perror("\n The msgget system call failed!");
```

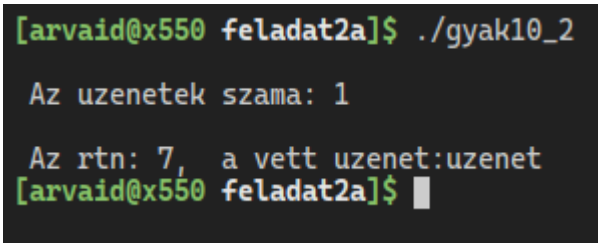
```
            return -1;
        }

        msgp = &msgbuf;
        buf = &ds;
        msgsz = 20;
        mtype = 0;

        rtn = msgctl(msgid, IPC_STAT, buf);
        printf("\n Az uzenetek szama: %d\n",buf->msg_qnum);

        if (buf->msg_qnum) {           /* van-e uzenet?*/
                        /* veszem a kovetkezo uzenetet: */
            rtn = msgrcv(msgid,(struct msgbuf *)msgp, msgsz, mtype, msgflg);
            printf("\n Az rtn: %d,  a vett uzenet:%s\n",rtn, msgp->mtext);
            rtn = msgctl(msgid,IPC_STAT,buf); /* uzenetsor adatokat lekerdezem,
                        benne azt is, hany uzenet van meg */
        }
        return 0;
    }
}
```

Futtatás eredménye:



3. Gyakorló feladat: Először tanulmányozzák Vadász Dénes: Operációs rendszer jegyzetet - 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 a segm nevű pointervál-tozót használva a processz virtuális címtartomanyába kapcsolja (attach) a szegmest (shmat() rendszerhívás). Olvassa, írja ezt a címtartományt, végül lekapcsolja (detach) a shmdt() rendszerhívással).

Futtatás eredménye:

```
[arvaid@x550 feladat3]$ ./shmcreate

 Nincs meg szegmens! Keszitsuk el!  Az shmid azonositoja 3702800:
[arvaid@x550 feladat3]$ ./shmctl

 Add meg a parancs szamat
 0 IPC_STAT (status)
 1 IPC_RMID (torles)   >  0

  Segm. meret: 512
  Utolso shmop-os proc. pid: 0
 [arvaid@x550 feladat3]$ ./shmop

 Uj szoveget kerek!
hello world

 Az uj szoveg: hello world
[arvaid@x550 feladat3]$ ./shmctl

 Add meg a parancs szamat
 0 IPC_STAT (status)
 1 IPC_RMID (torles)   >  0

  Segm. meret: 512
  Utolso shmop-os proc. pid: 12657
 [arvaid@x550 feladat3]$ ./shmctl

 Add meg a parancs szamat
 0 IPC_STAT (status)
 1 IPC_RMID (torles)   >  1

 Szegmens torolve
[arvaid@x550 feladat3]$
```

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)"

Mentés: gyak10_3.c

gyak10_3.c:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#define SHMKEY 123456L

int main() {
    int child = 0;

    if ((child = fork()) == 0) {
        int shmid;        /* osztott mem azonosito */
        key_t key;        /* kulcs a shmem-hez */
        int size=512;     /* osztott szegmens meret */
        int shmflg;       /* flag a jellemzokhoz */

        key = SHMKEY;
        /* Megnezzuk, van SHMKEY-es,size meretu szegmens. */
        shmflg = 0;
        if ((shmid=shmget( key, size, shmflg)) < 0) {
            printf("\n Nincs meg szegmens! Keszitsuk el!");
            shmflg = 00666 | IPC_CREAT;
            if ((shmid=shmget( key, size, shmflg)) < 0) {
                perror("\n Az shmget system-call sikertelen!");
                exit(-1);
            }
        } else printf("\n Van mar szegmens!");

        printf("  Az shmid azonositoja %d: \n", shmid);

        exit (0);
    }
    else {
        if (child = fork() == 0) {
            int shmid;    /* osztott mem azonosito */
            key_t key;    /* kulcs a shmem-hez */
            int size=512;       /* osztott szegmens meret */
            int shmflg;   /* flag a jellemzokhoz */
            struct vmi {
                int  hossz;
                char szoveg[512-sizeof(int)];
            } *segm;      /* Ezt a strukturat kepezzuk a szegmensre */
```

```
    key = SHMKEY;
    shmflg = 0; /* Nincs IPC_CREAT, feltetelezzuk, az shmcreate
                keszitett osztott memoria szegmenst */
    if ((shmid=shmget( key, size, shmflg)) < 0) {
        perror("\n Az shmget system-call sikertelen!");
        exit(-1);
    }

    /* Attach */
    shmflg = 00666 | SHM_RND;
    segm = (struct vmi *)shmat(shmid, NULL, shmflg); /* Itt a NULL azt
                jelenti, hogy az OS-re bizom, milyen
                cimtartomanyt hasznaljon. */
    if (segm == (void *)-1) {
        perror(" Sikertelen attach");
        exit (-1);
    }

    /* Sikeres attach utan a segm cimen ott az osztott memoria.
    Ha van benne valami, kiiratom, utana billentyuzetrol kerek
    uj beteendo szoveget */

    if (strlen(segm->szoveg) > 0)
        printf("\n Regi szoveg: %s (%d hosszon)",segm->szoveg,segm->hossz);

    printf("\n Uj szoveget kerek!\n");
    gets(segm->szoveg);
    printf("\n Az uj szoveg: %s\n",segm->szoveg);
    segm->hossz=strlen(segm->szoveg);

    /* Detach */
    shmdt(segm);

    exit(0);
}
else {
    int shmid;    /* osztott mem azonosito */
    key_t key;    /* kulcs a shmem-hez */
    int size=512;       /* osztott szegmens meret */
    int shmflg;   /* flag a jellemzokhoz */
    int rtn;      /* rensz.hivas visszter. ertek */
    int cmd;      /* parancskod */
    struct shmid_ds shmid_ds, *buf; /* adatstruktura a status
                    adatok fogadasahoz */
    buf = &shmid_ds;    /* es annak pointere */

    key = SHMKEY;
    shmflg = 0; /* Nincs IPC_CREAT, feltetelezzuk, az shmcreate
                keszitett osztott memoria szegmenst */
    if ((shmid=shmget( key, size, shmflg)) < 0) {
        perror("\n Az shmget system-call sikertelen!");
        exit(-1);
    }

/* Get the command */
do {
    printf("\n Add meg a parancs szamat ");
    printf("\n 0 IPC_STAT (status) ");
    printf("\n 1 IPC_RMID (torles)   >  ");
```

```
            scanf("%d",&cmd);
            } while (cmd < 0 && cmd > 1);

            switch (cmd)
            {
            case 0: rtn = shmctl(shmid, IPC_STAT, buf);
                printf("\n  Segm. meret: %d",buf->shm_segsz);
                printf("\n  Utolso shmop-os proc. pid: %d\n ",buf->shm_lpid);
                break;
            case 1: rtn = shmctl(shmid, IPC_RMID, NULL);
                printf("\n Szegmens torolve\n");
            }

            exit(0);
        }
    }
}
```

Futtatás eredménye: