Operációs rendszerek BSc

10. Gyak.

2022. 04. 13.

Készítette:

Szelényi Szabolcs Bsc

Mérnökinformatikus hallgató

TYNYS9

Miskolc, 2022

„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?”

	MAX. IGÉNY				FOGLALÁS			KIELÉGÍTETLEN IGÉNYEK		
	R1	R2	R3		R1	R2	R3	R1	R2	R3
P0	7	5	3		0	1	0	7	4	3
P1	3	2	2		2	0	0	1	2	2
P2	9	0	2		3	0	2	6	0	0
P3	2	2	2		2	1	1	0	1	1
P4	4	3	3		0	0	2	4	3	1

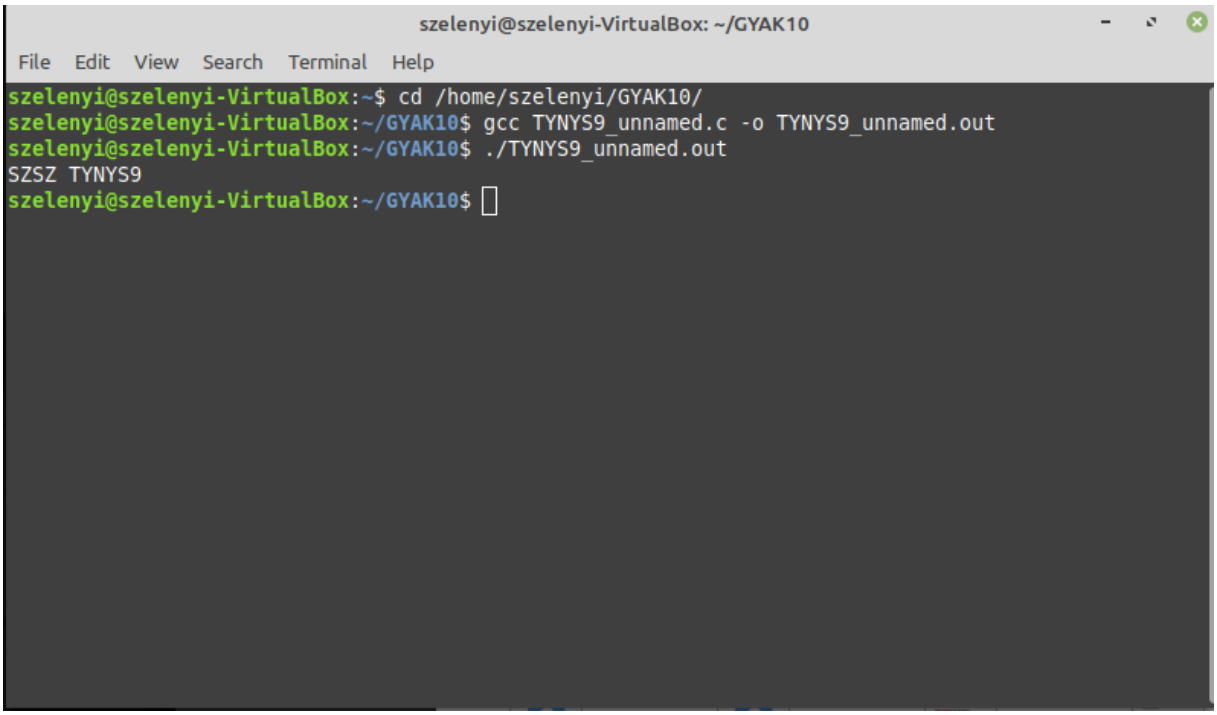
				KÉSZLET-IGÉNY			
Foglaltak	7	2	5	R1	R2	R3	
Összesen	10	5	7	-4	-1	-1	P0
Szabad erőforrás szám	3	3	2	2	1	0	P1
				-3	3	2	P2
				3	2	1	P3
				-1	0	1	P4

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

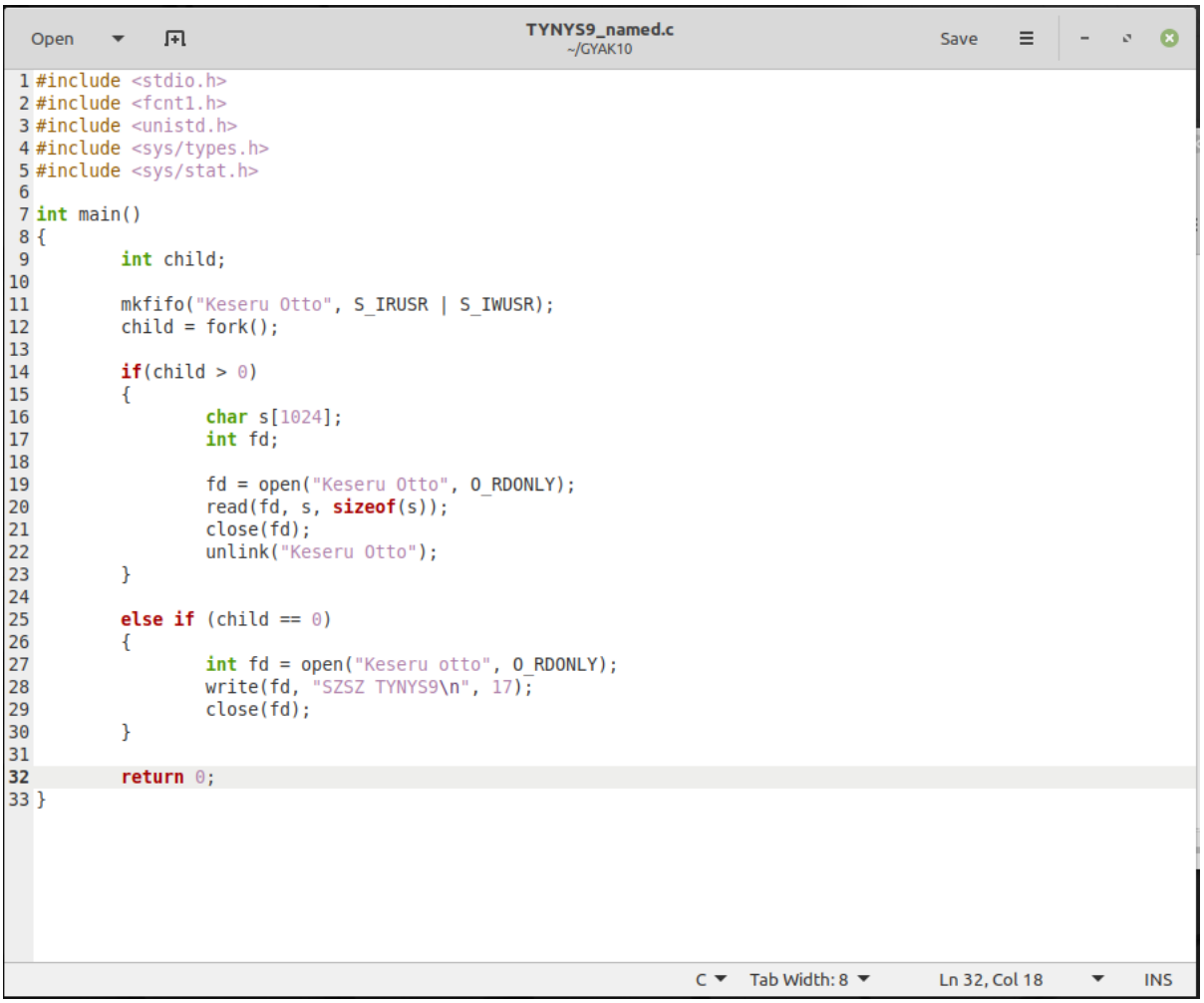
Open — TYNYS9_unnamed.c — ~/GYAK10 — Save

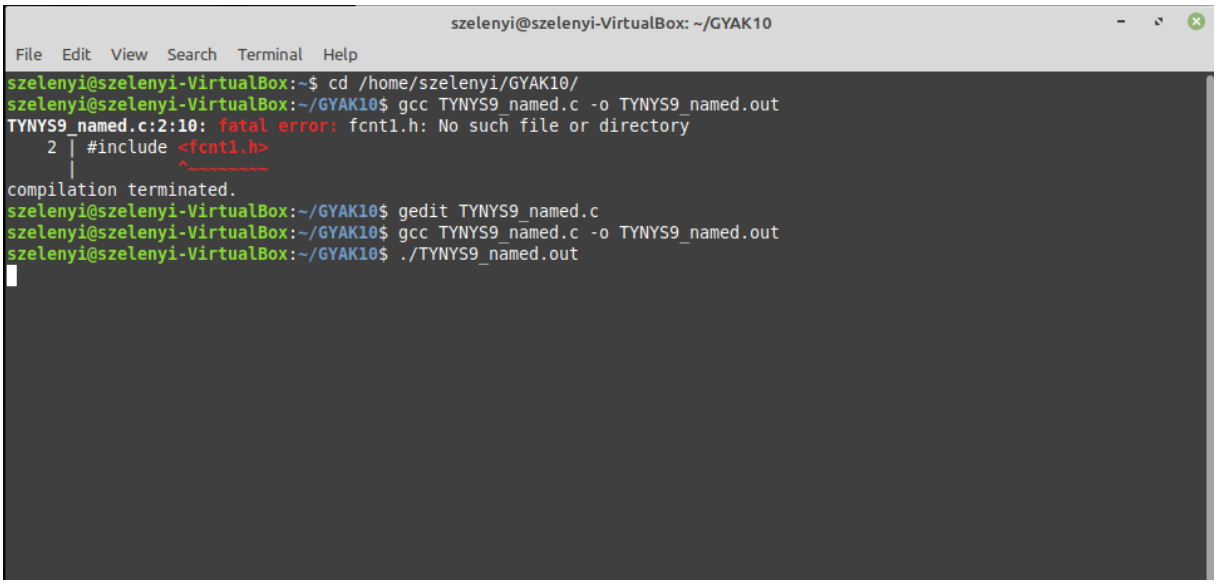
```c
#include <stdio.h>
#include <unistd.h>

int main()
{
        int fd[2];
        int child;

        if(pipe(fd)
        {
                perror("pipe");
                return 1;
        }

        child = fork();

        if(child > 0)
        {
                char s[1024];
                close(fd[1]);
                read(fd[0], s, sizeof(s));
                printf("%s", s);

                close(fd[0]);
        }

        else if(child == 0)
        {
                close(fd[0]);
                write(fd[1], "SZSZ TYNYS9\n", 17);
                close(fd[1]);
        }
        return 0;
}
```

C — Tab Width: 8 — Ln 34, Col 2 — INS

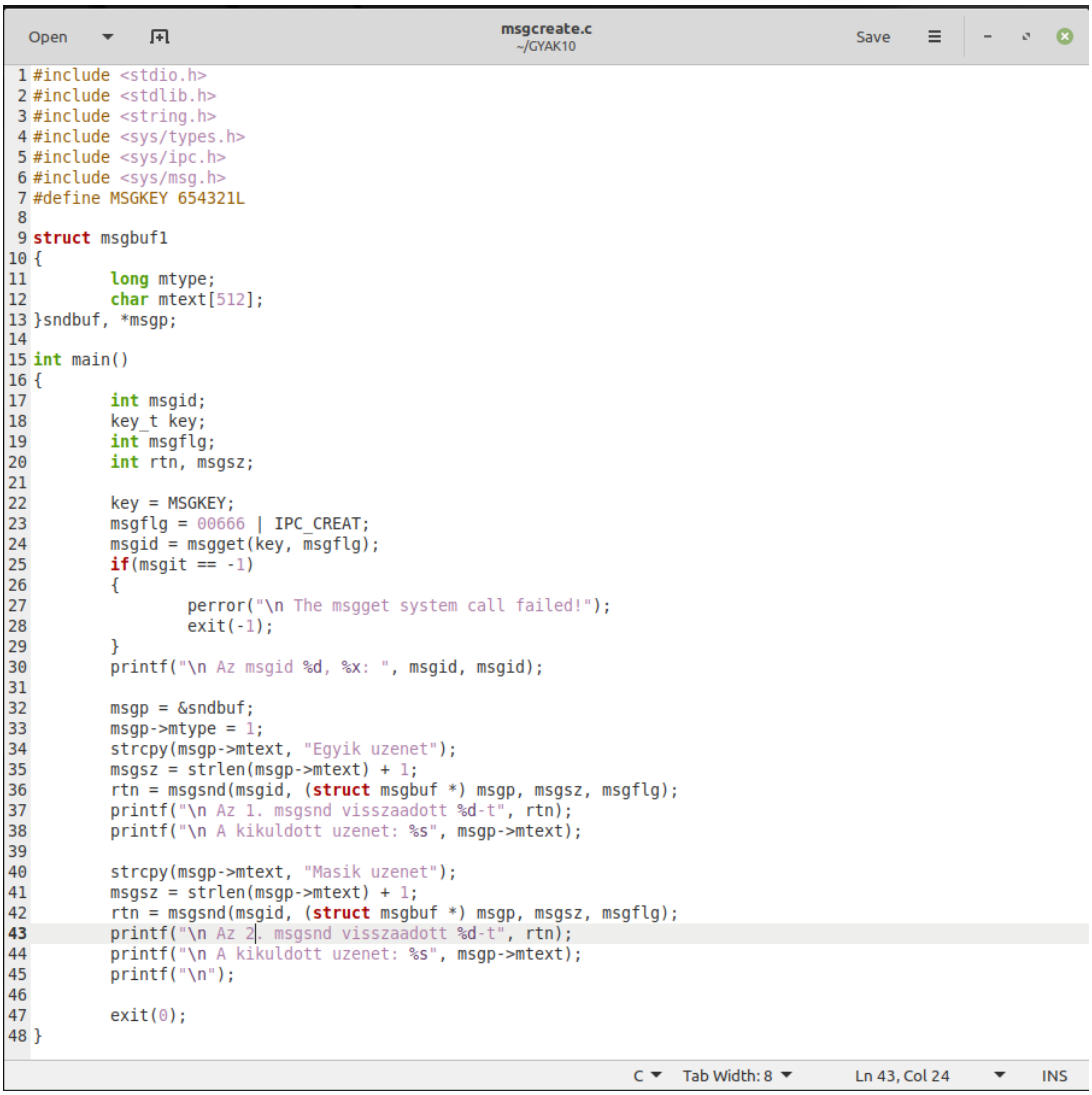


3. Készítsen C nyelvű programot, ahol egy szülő processz létrehoz egy nevesített csővezetéket (neve: neptunkod), a gyerek processz beleír egy szöveget a csővezetékbe (A hallgató neve: pl.: Keserű Ottó), a szülő processz ezt kiolvassa, és kiírja a standard kimenetre.





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



```
szelenyi@szelenyi-VirtualBox: ~/GYAK10
File  Edit  View  Search  Terminal  Help
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gcc TYNYS9_named.c -o TYNYS9_named.out
szelenyi@szelenyi-VirtualBox:~/GYAK10$ ./TYNYS9_named.out
^[[A^C
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gedit msgcreate.c
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gcc msgcreate.c -o msgcreate.c
gcc: fatal error: input file 'msgcreate.c' is the same as output file
compilation terminated.
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gcc msgcreate.c -o msgcreate.out
msgcreate.c: In function 'main':
msgcreate.c:25:5: error: 'msgit' undeclared (first use in this function); did you mean 'msgid'?
   25 |  if(msgit == -1)
      |     ^~~~~
      |     msgid
msgcreate.c:25:5: note: each undeclared identifier is reported only once for each function it appears in
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gedit msgcreate.c
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gcc msgcreate.c -o msgcreate.out
szelenyi@szelenyi-VirtualBox:~/GYAK10$ ./msgcreate.out

 Az msgid 0, 0:
 Az 1. msgsnd visszaadott 0-t
 A kikuldott uzenet: Egyik uzenet
 Az 2. msgsnd visszaadott 0-t
 A kikuldott uzenet: Masik uzenet
szelenyi@szelenyi-VirtualBox:~/GYAK10$
```

```
Open        msgctl.c
            ~/GYAK10                Save
 1 #include <stdio.h>
 2 #include <stdlib.h>
 3 #include <sys/types.h>
 4 #include <sys/ipc.h>
 5 #include <sys/msg.h>
 6 #define MSGKEY 654321L
 7
 8 int main()
 9 {
10         int msgid, msgflg, rtn;
11         key_t key;
12         key = MSGKEY;
13         msgflg = 006666 | IPC_CREAT
14         msgid = msgget(key, msgflg);
15
16         rtn = msgctl(msgid, IPC_RMID, NULL);
17         printf("\n Visszatert: %d\n", rtn);
18
19         exit(0)
20 }
```

```
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gedit msgctl.c
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gcc msgctl.c -o msgctl.out
szelenyi@szelenyi-VirtualBox:~/GYAK10$ ./msgctl.out

 Visszatert: -1
szelenyi@szelenyi-VirtualBox:~/GYAK10$
```

Open

msgrcv.c
~/GYAK10

Save

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

struct msgbuf1
{
        long mtype;
        char mtext[512];
}rcvbuf, *msgp;

struct msqid_ds ds, *buf;

int main()
{
        int msgid;
        key_t key;
        int mtype, msgflg;
        int rtn, msgsz;

        key = MSGKEY;
        msgflg = 006666 | IPC_CREAT | MSG_NOERROR;

        msgid = msgget(key, msgflg);
        if(msgid == -1)
        {
                perror("\n The msgget system call failed!");
                exit(-1);
        }
        printf("\n Az msgid: %d", msgid);

        msgp = &rcvbuf;
        buf = &ds;
        msgsz = 20;
        mtype = 0;
        rtn = msgctl(msgid, IPC_STAT, buf);
        printf("\nAz uzenetek szama: %ld \n", buf->msg_qnum);

        while(buf->msg_qnum)
        {
                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);
        }
        exit(0);
}
```

C Tab Width: 8 Ln 32, Col 37 INS

```
szelenyi@szelenyi-VirtualBox:~/GYAK10$ ./msgrcv.out

 The msgget system call failed!: File exists
szelenyi@szelenyi-VirtualBox:~/GYAK10$
```

4a. 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.

Open — gyak10_4.c ~/GYAK10 — Save

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

struct msgbuf1
{
        long mtype;
        char mtext[256];
}sndbuf, *msgp;

int main()
{
        int id;
        key_t key;
        int flag;
        int rtn, size;
        int ok = 1, count = 1;

        char teszt[256];
        key = MSGKEY;
        flag = 006666 | IPC_CREAT;
        id = msgget(key, flag);
        if(id == -1)
        {
                perror("\nAz msgget hivas nem valosult meg");
                exit(-1);
        }

        do
        {
                scanf("%s", teszt);
                msgp = &sndbuf;
                msgp->mtype = 1;
                size = strlen(msgp->mtext) + 1;

                if(strcmp("exit", teszt) != 0)
                {
                        rtn = msgsnd(id,(struct msgbuf *) msgp, size, flag);
                        printf("\nAz %d. msgsnd visszaadott %d-t", count, id);
                        printf("\nA kikuldott uzenet: %s\n", msgp->mtext);
                        count++;
                }
                else
                {
                        ok = 0;
```

C — Tab Width: 8 — Ln 6, Col 19 — INS

```
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gedit gyak10_4.c
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gcc gyak10_4.c -o gyak10_4.out
szelenyi@szelenyi-VirtualBox:~/GYAK10$ ./gyak10_4.out

Az msgget hivas nem valosult meg: File exists
szelenyi@szelenyi-VirtualBox:~/GYAK10$
```

5. 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, irja ezt a címtartományt, végül lekapcsolja (detach) a shmdt() rendszerhívással).

Open *shmcreate.c ~/GYAK10 Save

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <string.h>
#define KEY 2022

int main()
{
        int sharedMemoryID = shmget(KEY, 256, IPC_CREAT | 0666);
        return 0;
}
```

Open shmctl.c ~/GYAK10 Save

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <string.h>
#define KEY 2022

void main()
{
        int sharedMemoryID = shmget(KEY, 0, 0);
        struct shmid_ds buffer;
        if(shmctl(sharedMemoryID, IPC_STAT, &buffer) == -1)
        {
                perror("Nem sikrult az adatokat lekerdezni");
                exit(-1);
        }
}
```

Open shmop.c ~/GYAK10 Save

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm>
#include <string.h>
#define KEY 2022

void main()
{
        int sharedMemoryID = shmget(KEY, 0, 0);
        char *segm = shmat(sharedMemoryID, NULL, SHM_RND);
        strcpy(segm, "Egy uj uzenet erkezett");
        printf("A kozos memoria tartalma: %s\n", segm);

        shmdt(segm);
}
```

```
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gcc shmop.c -o shmop.out
szelenyi@szelenyi-VirtualBox:~/GYAK10$ ./shmop.out
A kozos memoria tartalma: Egy uj uzenet erkezett
szelenyi@szelenyi-VirtualBox:~/GYAK10$
```

5a. 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)”

Open ▾ gyak10_5.c ~/GYAK10 Save

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <string.h>
#include <unistd.h>
#define KEY 777777

void main()
{
    pid_t process1;
    pid_t process2;
    pid_t process3;

    process1 = fork();
    if (process1 == 0)
    {
        int sharedMemoryID = shmget(KEY, 256, IPC_CREAT | 0666);
        if (sharedMemoryID == -1)
        {
            perror("Nem sikerult lefoglalni a memoriar\n");
            exit(-1);
        }
        printf("Process1 lefoglalta a memoriat!\n");
    }
    else
    {
         process2 = fork();
         if (process2 == 0)
         {
            printf("Process 2 olvas\n");
            int sharedMemoryID = shmget(KEY, 0, 0);
            char *s = shmat(sharedMemoryID, NULL, SHM_RND);
            strlen(s) > 0 ? printf("osztott memoriaban szereplo szoveg : %s\n", s)
                          : printf("Nincs benne szoveg\n");
            strcpy(s, "Ez egy uj szoveg");
            printf("process2 kuldte az uzenetet.\n");
         }
         else
         {
             process3 = fork();
             if (process3 == 0)
             {
                printf("process3: \n");
                int sharedMemoryID = shmget(KEY, 0, 0);
                struct shmid_ds buffer;
                if (shmctl(sharedMemoryID, IPC_STAT, &buffer) == -1)
                {
                    perror("Nem sikerult lekerdezni.\n");
```

C ▾ Tab Width: 8 ▾ Ln 7, Col 20 ▾ INS

```
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gedit gyak10_5.c
szelenyi@szelenyi-VirtualBox:~/GYAK10$ gcc gyak10_5.c -o gyak10_5.out
szelenyi@szelenyi-VirtualBox:~/GYAK10$ ./gyak10_5.out
szelenyi@szelenyi-VirtualBox:~/GYAK10$ process3:
Nem sikerult lekerdezni.
: Invalid argument
Process 2 olvas
Process1 lefoglalta a memoriat!
```