

# **Operációs rendszerek BSc**

**11. Gyak.**

2022. 04. 20.

**Készítette:**

Pázmán András Bsc  
Mérnökinformatikus  
H2Z4X3

**Miskolc, 2022**

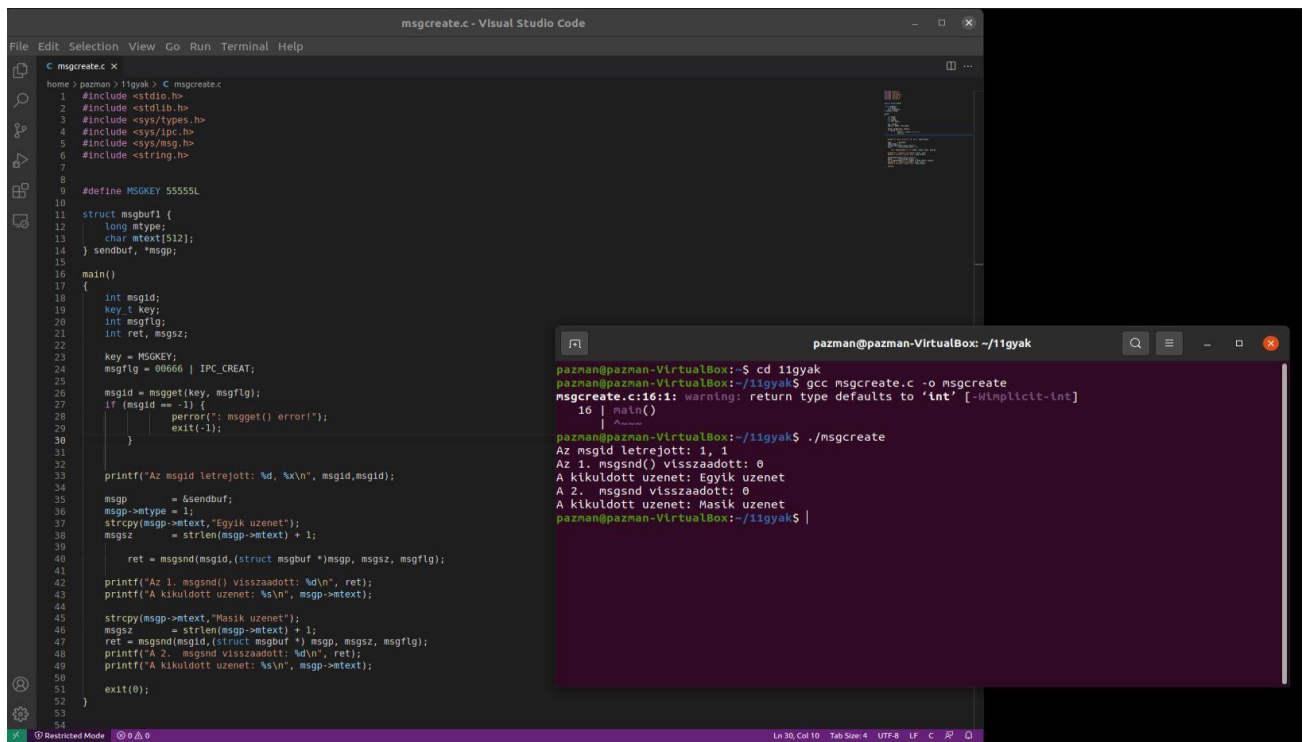
## 4. Gyakorló feladat

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**.

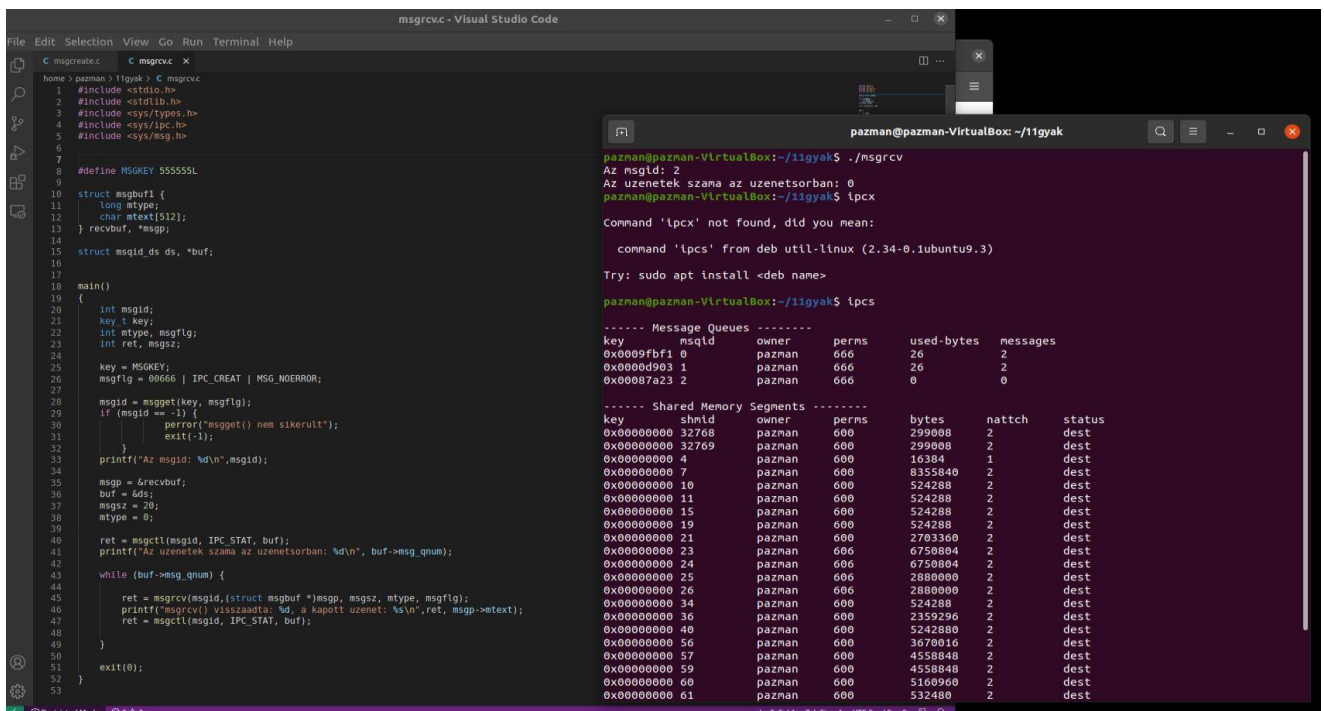
A futtatás eredményét is tartalmazza a jegyzőkönyv.

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



The screenshot shows the Visual Studio Code editor with the **msgcreate.c** file open. The code defines a message key, a message type, and a message buffer. It then creates a message queue and sends two messages to it. The terminal output shows the execution of the program, which prints the message key, the message type, and the message buffer. The output is as follows:

```
pazman@pazman-VirtualBox: ~/11gyak
pazman@pazman-VirtualBox:~/11gyak$ gcc msgcreate.c -o msgcreate
msgcreate.c:16:11: warning: return type defaults to 'int' [-Wimplicit-int]
16 |   main()
   |   ^~~~~
pazman@pazman-VirtualBox:~/11gyak$ ./msgcreate
Az msqid létrejött: 1, 1
Az 1. msgsnd() visszaadott: 0
A kiküldött üzenet: Egyik üzenet
A 2. msgsnd visszaadott: 0
A kiküldött üzenet: Másik üzenet
pazman@pazman-VirtualBox:~/11gyak$
```



The screenshot shows the Visual Studio Code editor with the **msgrcv.c** file open. The code receives messages from the message queue and prints them. The terminal output shows the execution of the program, which prints the message key, the message type, and the message buffer. The output is as follows:

```
pazman@pazman-VirtualBox:~/11gyak$ ./msgrcv
Az msqid: 2
Az üzenetek száma az üzenetsorban: 0
pazman@pazman-VirtualBox:~/11gyak$ ipcs
Command 'ipcs' not found, did you mean:
  command 'ipcs' from deb util-linux (2.34-0.1ubuntu9.3)
Try: sudo apt install <deb name>
pazman@pazman-VirtualBox:~/11gyak$ ipcs
----- Message Queues -----
key      msqid    owner    perms    used-bytes  messages
0x00009fb1 0      pazman   666      26          2
0x0000d903 1      pazman   666      26          2
0x00007a23 2      pazman   666      0           0

----- Shared Memory Segments -----
key      shmid     owner    perms    bytes       nattch     status
0x00000000 32768     pazman   600      299808      2          dest
0x00000000 32769     pazman   600      299808      2          dest
0x00000000 4        pazman   600      16384       1          dest
0x00000000 7        pazman   600      8355840     2          dest
0x00000000 10       pazman   600      524288      2          dest
0x00000000 11       pazman   600      524288      2          dest
0x00000000 15       pazman   600      524288      2          dest
0x00000000 19       pazman   600      524288      2          dest
0x00000000 21       pazman   600      2703360     2          dest
0x00000000 23       pazman   600      6758804     2          dest
0x00000000 24       pazman   600      6758804     2          dest
0x00000000 25       pazman   600      2888000     2          dest
0x00000000 26       pazman   600      2888000     2          dest
0x00000000 34       pazman   600      524288      2          dest
0x00000000 36       pazman   600      2359296     2          dest
0x00000000 40       pazman   600      5242880     2          dest
0x00000000 56       pazman   600      3670016     2          dest
0x00000000 57       pazman   600      4558848     2          dest
0x00000000 59       pazman   600      4558848     2          dest
0x00000000 60       pazman   600      5160960     2          dest
0x00000000 61       pazman   600      532480      2          dest
```

The image shows a Visual Studio Code editor with a C program named `msgctl.c` open. The program is a simple test for message queues. It includes `stdio.h`, `stdlib.h`, `sys/types.h`, `sys/ipc.h`, and `sys/msg.h`. It defines a message key `MSGKEY 555555L`. In the `main` function, it creates a message queue `msgq` with `key = MSGKEY` and `msgflg = 00666 | IPC_CREAT`. It then gets the message queue ID `msgid` using `msgget`. Finally, it prints the message queue ID and the return value of `msgctl(msgid, IPC_RMID, NULL)` using `printf`.

The terminal window shows the execution of the program. It displays the message queue ID `0x00009fbf1` and the return value `0`. Below this, it shows the output of the `ipcs` command, which displays the message queue information and the shared memory segments.

```
----- Message Queues -----
key      msgid  owner    perms    used-bytes  messages
0x00009fbf1 0      pazman   666      26           2
0x0000d903 1      pazman   666      26           2

----- Shared Memory Segments -----
key      shmid   owner    perms    bytes       nattch     status
0x00000000 32768    pazman   600      299008      2          dest
0x00000000 32769    pazman   600      299008      2          dest
0x00000000 4        pazman   600      16384       1          dest
0x00000000 7        pazman   600      8355840     2          dest
0x00000000 32777    pazman   600      3670016     2          dest
0x00000000 10       pazman   600      524288      2          dest
0x00000000 11       pazman   600      524288      2          dest
0x00000000 32781    pazman   600      2703360     2          dest
0x00000000 19       pazman   600      524288      2          dest
0x00000000 21       pazman   600      2703360     2          dest
0x00000000 23       pazman   606      6750804     2          dest
0x00000000 24       pazman   606      6750804     2          dest
0x00000000 25       pazman   606      2880000     2          dest
0x00000000 26       pazman   606      2880000     2          dest
0x00000000 32799    pazman   600      5242880     2          dest
0x00000000 32801    pazman   600      524288      2          dest
0x00000000 34       pazman   600      524288      2          dest
0x00000000 36       pazman   600      2359296     2          dest
0x00000000 57       pazman   600      4558048     2          dest
0x00000000 59       pazman   600      4558048     2          dest
0x00000000 60       pazman   600      5160960     2          dest
0x00000000 61       pazman   600      532480      2          dest
0x00000000 62       pazman   600      532480      2          dest

----- Semaphore Arrays -----
key      semid   owner    perms    nsems
pazman@pazman-VirtualBox: ~/11gyak$
```

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.

Mentés: **gyak10\_4.c**

A futtatás eredményét is tartalmazza a jegyzőkönyv.

The image shows a Visual Studio Code editor with a C program named `gyak10_4.c` open. The program is a simple test for message queues. It includes `sys/types.h`, `sys/ipc.h`, and `sys/msg.h`. It defines a message key `MSGKEY 555555L`. In the `main` function, it creates a message queue `msgq` with `key = MSGKEY` and `msgflg = 00666 | IPC_CREAT`. It then gets the message queue ID `msgid` using `msgget`. Finally, it prints the message queue ID and the return value of `msgctl(msgid, IPC_RMID, NULL)` using `printf`.

The terminal window shows the execution of the program. It displays the message queue ID `0x00009fbf1` and the return value `0`. Below this, it shows the output of the `ipcs` command, which displays the message queue information and the shared memory segments.

```
----- Message Queues -----
key      msgid  owner    perms    used-bytes  messages
0x00009fbf1 0      pazman   666      26           2
0x0000d903 1      pazman   666      26           2

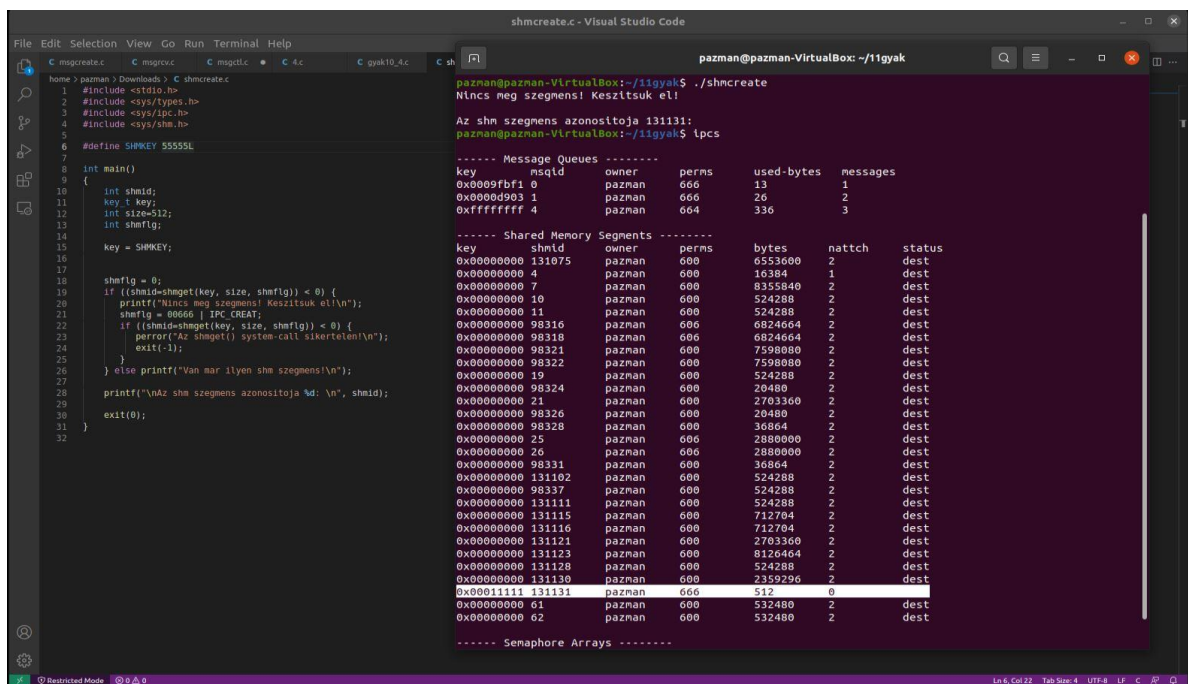
----- Shared Memory Segments -----
key      shmid   owner    perms    bytes       nattch     status
0x00000000 32768    pazman   600      299008      2          dest
0x00000000 32769    pazman   600      299008      2          dest
0x00000000 4        pazman   600      16384       1          dest
0x00000000 7        pazman   600      8355840     2          dest
0x00000000 32777    pazman   600      3670016     2          dest
0x00000000 10       pazman   600      524288      2          dest
0x00000000 11       pazman   600      524288      2          dest
0x00000000 32781    pazman   600      2703360     2          dest
0x00000000 19       pazman   600      524288      2          dest
0x00000000 21       pazman   600      2703360     2          dest
0x00000000 23       pazman   606      6750804     2          dest
0x00000000 24       pazman   606      6750804     2          dest
0x00000000 25       pazman   606      2880000     2          dest
0x00000000 26       pazman   606      2880000     2          dest
0x00000000 32799    pazman   600      5242880     2          dest
0x00000000 32801    pazman   600      524288      2          dest
0x00000000 34       pazman   600      524288      2          dest
0x00000000 36       pazman   600      2359296     2          dest
0x00000000 57       pazman   600      4558048     2          dest
0x00000000 59       pazman   600      4558048     2          dest
0x00000000 60       pazman   600      5160960     2          dest
0x00000000 61       pazman   600      532480      2          dest
0x00000000 62       pazman   600      532480      2          dest

----- Semaphore Arrays -----
key      semid   owner    perms    nsems
pazman@pazman-VirtualBox: ~/11gyak$
```

**5. 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** shm-id-del azonosít osztott memória szegmenst. Ezután a 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).



```
shmcreate.c - Visual Studio Code

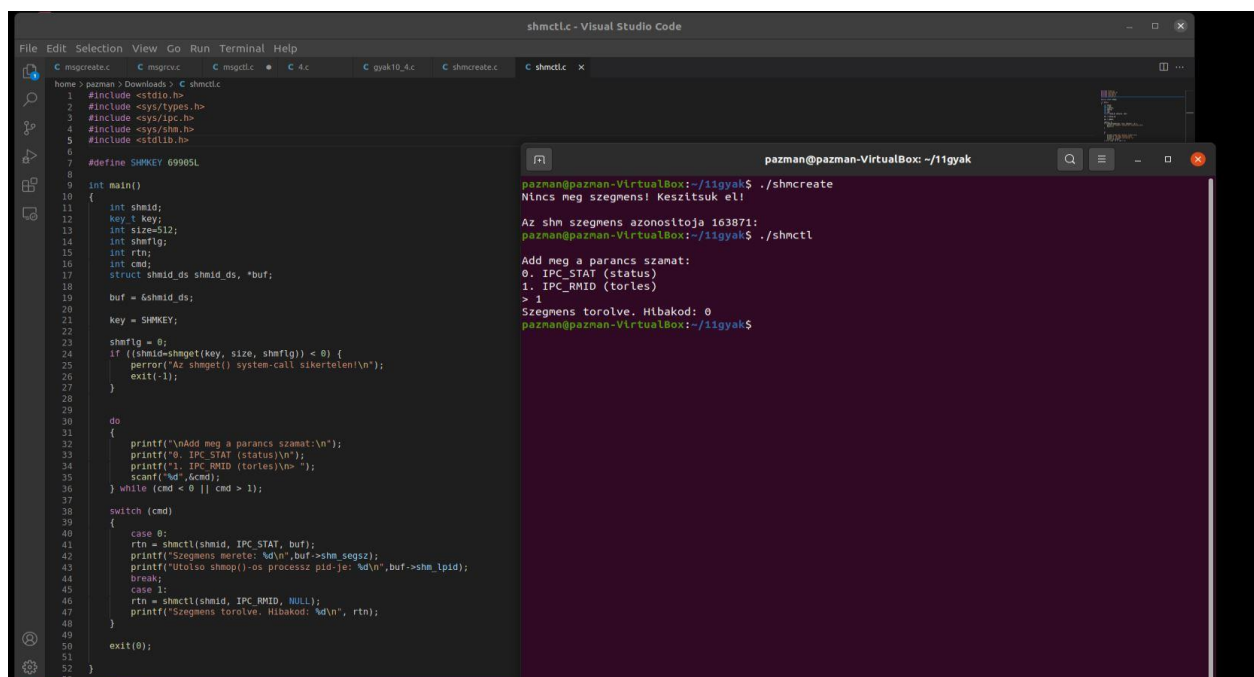
home > pazman > Downloads > C shmcreate.c
1 #include <stdio.h>
2 #include <sys/types.h>
3 #include <sys/ipc.h>
4 #include <sys/shm.h>
5
6 #define SHMKEY 55555L
7
8 int main()
9 {
10     int shmid;
11     key_t key;
12     int size=512;
13     int shmflg;
14
15     key = SHMKEY;
16
17     shmflg = 0;
18     if ((shmid=shmget(key, size, shmflg)) < 0) {
19         printf("Nincs meg szegmens! Készítsuk el!\n");
20         shmflg = 0666 | IPC_CREAT;
21         if ((shmid=shmget(key, size, shmflg)) < 0) {
22             perror("Az shmget() system-call sikertelen!\n");
23             exit(-1);
24         }
25     } else printf("Van már ilyen shm szegmens!\n");
26     printf("\nAz shm szegmens azonosítója %d: \n", shmid);
27     exit(0);
28 }

pazman@pazman-VirtualBox: ~/11gyak$ ./shmcreate
Nincs meg szegmens! Készítsuk el!
Az shm szegmens azonosítója 131131:
pazman@pazman-VirtualBox: ~/11gyak$ lpcsh

----- Message Queues -----
key      msgid   owner    perms   used-bytes  messages
0x00009fbf1 0      pazman   666     13         1
0x0000d903 1      pazman   666     26         2
0xffffffff 4      pazman   664     336        3

----- Shared Memory Segments -----
key      shmid   owner    perms   bytes      nattch   status
0x00000000 131075   pazman   600     6553600    2        dest
0x00000000 4        pazman   600     16384      1        dest
0x00000000 7        pazman   600     8355840    2        dest
0x00000000 10       pazman   600     524288     2        dest
0x00000000 11       pazman   600     524288     2        dest
0x00000000 98316    pazman   606     6824664    2        dest
0x00000000 98318    pazman   606     6824664    2        dest
0x00000000 98321    pazman   600     7598080    2        dest
0x00000000 98322    pazman   600     7598080    2        dest
0x00000000 19       pazman   600     524288     2        dest
0x00000000 98324    pazman   600     20480      2        dest
0x00000000 21       pazman   600     2703360    2        dest
0x00000000 98326    pazman   600     20480      2        dest
0x00000000 98328    pazman   600     36864      2        dest
0x00000000 25       pazman   606     2880000    2        dest
0x00000000 26       pazman   606     2880000    2        dest
0x00000000 98331    pazman   600     36864      2        dest
0x00000000 131102   pazman   600     524288     2        dest
0x00000000 98337    pazman   600     524288     2        dest
0x00000000 131111   pazman   600     524288     2        dest
0x00000000 131115   pazman   600     712704     2        dest
0x00000000 131116   pazman   600     712704     2        dest
0x00000000 131121   pazman   600     2703360    2        dest
0x00000000 131123   pazman   600     8126464    2        dest
0x00000000 131128   pazman   600     524288     2        dest
0x00000000 131130   pazman   600     2359296    2        dest
0x00011111 131131   pazman   666     512        0
0x00000000 61       pazman   600     532480     2        dest
0x00000000 62       pazman   600     532480     2        dest

----- Semaphore Arrays -----
```



```
shmctl.c - Visual Studio Code

home > pazman > Downloads > C shmctl.c
1 #include <stdio.h>
2 #include <sys/types.h>
3 #include <sys/ipc.h>
4 #include <sys/shm.h>
5 #include <stdlib.h>
6
7 #define SHMKEY 69905L
8
9 int main()
10 {
11     int shmid;
12     key_t key;
13     int size=512;
14     int shmflg;
15     int rtn;
16     int cmd;
17     struct shmid_ds shmid_ds;
18     buf = &shmid_ds;
19
20     key = SHMKEY;
21
22     shmflg = 0;
23     if ((shmid=shmget(key, size, shmflg)) < 0) {
24         perror("Az shmget() system-call sikertelen!\n");
25         exit(-1);
26     }
27
28     do
29     {
30         printf("\nAdd meg a parancs számát:\n");
31         printf("0. IPC_STAT (status)\n");
32         printf("1. IPC_RMID (torles)\n");
33         printf("2. IPC_SET (set)\n");
34         scanf("%d",&cmd);
35     } while (cmd < 0 || cmd > 2);
36
37     switch (cmd)
38     {
39     case 0:
40         rtn = shmctl(shmid, IPC_STAT, buf);
41         printf("Szegmens merete: %d\n",buf->shm_segsz);
42         printf("Utoiso shmop()-os processz pid-je: %d\n",buf->shm_lpid);
43         break;
44     case 1:
45         rtn = shmctl(shmid, IPC_RMID, NULL);
46         printf("Szegmens torolve. Hibakod: %d\n", rtn);
47     }
48     exit(0);
49 }

pazman@pazman-VirtualBox: ~/11gyak$ ./shmcreate
Nincs meg szegmens! Készítsuk el!
Az shm szegmens azonosítója 163871:
pazman@pazman-VirtualBox: ~/11gyak$ ./shmctl

Add meg a parancs számát:
0. IPC_STAT (status)
1. IPC_RMID (torles)
2. IPC_SET (set)
> 1
Szegmens torolve. Hibakod: 0
pazman@pazman-VirtualBox: ~/11gyak$
```

```
File Edit Selection View Go Run Terminal Help
C msgcreate.c C msgrcv.c C msgctl.c C 4.c C gyak10_4.c C shmcreate.c C shmctl.c C shmop.c x
home > pazman > /11gyak > C shmop.c
9 int main()
10 {
11     int shmid;
12     key_t key;
13     int size=512;
14     int shmflg;
15     struct vmi {
16         int hossz;
17         char szoveg[512-sizeof(int)];
18     } *segm;
19     key = SHMKEY;
20     shmflg = 0;
21
22     if ((shmid=shmget(key, size, shmflg)) < 0) {
23         perror("Az shmget system-call sikertelen!\n");
24         exit(-1);
25     }
26
27     shmflg = 00666 | SHM_RND;
28     segm = (struct vmi *)shmat(shmid, NULL, shmflg);
29
30     if (segm == (void *)-1) {
31         perror("Sikertelen attach!\n");
32         exit(-1);
33     }
34
35     if (strlen(segm->szoveg) > 0)
36         printf("\n Regi szoveg: %s (%d hossz)", segm->szoveg, segm->hossz);
37
38     printf("\nUj szoveget kerek!\n");
39     gets(segm->szoveg);
40     printf("Az uj szoveg: %s\n", segm->szoveg);
41     segm->hossz=strlen(segm->szoveg);
42
43     shmctl(segm);
44     exit(0);
45 }
46
47
48
49
50
51
52
```

```
pazman@pazman-VirtualBox: ~/11gyak
pazman@pazman-VirtualBox:~/11gyak$ ./shmop
Az shmget system-call sikertelen!
: No such file or directory
pazman@pazman-VirtualBox:~/11gyak$ |
```

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: státusz lekérése (szegmens mérete, utolsó shmop-os proc. pid-je), osztott memória megszüntetése, kilépés (2. és 3. proc. lehet egyben is)”

A futtatás eredményét is tartalmazza a jegyzőkönyv.

Mentés: gyak10\_5.c

```
File Edit Selection View Go Run Terminal Help
C msgcreate.c C msgrcv.c C msgctl.c C 4.c C gyak10_4.c C shmcreate.c C shmctl.c C shmop.c C gyak10_5.c x
home > pazman > /11gyak > C gyak10_5.c
1
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <sys/types.h>
5 #include <sys/ipc.h>
6 #include <sys/shm.h>
7
8 #define SHMKEY 555555L
9
10 int main()
11 {
12     int shmid;
13     key_t key = SHMKEY;
14     int size=512;
15     int shmflg;
16
17     shmflg = 0;
18     if ((shmid=shmget(key, size, shmflg)) < 0) {
19         shmflg = 00666 | IPC_CREAT;
20         if ((shmid=shmget(key, size, shmflg)) < 0) {
21             perror("shmget()hiba!\n");
22             exit(-1);
23         }
24         printf("A szegmens elkeszult!\n");
25     } else printf("A szegmens mar letezik!\n");
26
27     printf("\nAz shm szegmens azonositoja %d: \n", shmid);
28     exit(0);
29 }
30
```

```
pazman@pazman-VirtualBox: ~/11gyak
pazman@pazman-VirtualBox:~$ cd 11gyak
pazman@pazman-VirtualBox:~/11gyak$ gcc gyak10_5.c
pazman@pazman-VirtualBox:~/11gyak$ ./5.c
bash: ./5.c: No such file or directory
pazman@pazman-VirtualBox:~/11gyak$ gcc gyak10_5.c -o 5.c
pazman@pazman-VirtualBox:~/11gyak$ ./5.c
A szegmens elkeszult

Az shm szegmens azonositoja 196648:
pazman@pazman-VirtualBox:~/11gyak$ |
```



```
File Edit Selection View Go Run Terminal Help
C msgcreate.c C msgrcv.c C msgctl.c C 4.c C gyakorlat_4.c C shmcreate.c C shmctl.c C shmop.c C gyakorlat_5.c C gyakorlat_5a.c
home > pazman > 11gyak > C gyakorlat_5a.c
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <sys/types.h>
4 #include <sys/ipc.h>
5 #include <sys/shm.h>
6 #include <string.h>
7
8 #define SHMKEY 555555L
9
10 int main()
11 {
12     int shmid;
13     key_t key = SHMKEY;
14     int size=512;
15     int shmflg, rtn;
16     char c;
17     struct shmld_ds shmld_ds;
18     struct vmi {
19         int hossz;
20         char szoveg[size-sizeof(int)];
21     } *segm;
22
23     shmflg = 0;
24
25     if ((shmid=shmget(key, size, shmflg))
26         perror("Shmget hiba\n");
27         exit(-1);
28     }
29
30     shmflg = 0666 | SHM_RND;
31     segm = (struct vmi *)shmat(shmid, NULL,
32                                shmflg);
33     if (segm == (void *)-1) {
34         perror("Shmat hiba\n");
35         exit(-1);
36     }
37
38     do{
39         printf("1 - Kiolvasas es beiras\n");
40         scanf("%c", &c);
41         while(getchar()!='\n');
42         switch (c)
43         {
44             case '1':
45                 printf("Kiolvasas es beiras\n");
46                 break;
47             case '2':
48                 printf("Statusz lekerese\n");
49                 break;
50             case '3':
51                 printf("Szegmens megszuntes es kilepes\n");
52                 break;
53             default:
54                 printf("Hibas beiras\n");
55                 break;
56         }
57     } while(c != '\n');
```

pazman@pazman-VirtualBox: ~/11gyak

```
pazman@pazman-VirtualBox:~/11gyak$ gcc gyakorlat_5.c -o 5.c
pazman@pazman-VirtualBox:~/11gyak$ ./5.c
A szegmens elkeszult

Az shm szegmens azonositoja 196657:
pazman@pazman-VirtualBox:~/11gyak$ gcc gyakorlat_5a.c -o 5a.c
gyakorlat_5a.c: In function 'main':
gyakorlat_5a.c:57:30: warning: format '%d' expects argument of type 'int', but argument 2 has type 'size_t' {aka 'long unsigned int'} [-Wformat=]
57 |         printf("Szegmens merete: %d\n", shmld_ds.shm_segsz);
    |                                ^~
    |                                |
    |                                int      size_t {aka long unsigned int}
    |                                %ld
pazman@pazman-VirtualBox:~/11gyak$ ./5a.c
1 - Kiolvasas es beiras
2 - Statusz lekerese
3 - Szegmens megszuntes es kilepes
1
Uj szoveg:
asadkasdkas daokdop spodpasd pdkpaspo
Az uj szoveg: asadkasdkas daokdop spodpasd pdkpaspo
1 - Kiolvasas es beiras
2 - Statusz lekerese
3 - Szegmens megszuntes es kilepes
2
Szegmens merete: 512
Utolso shmop()-os processz pid-je: 21712
1 - Kiolvasas es beiras
2 - Statusz lekerese
3 - Szegmens megszuntes es kilepes
3
Szegmens torolve. Visszateresi ertek: 0
pazman@pazman-VirtualBox:~/11gyak$
```