

Operációs rendszerek BSc

10. Gyak.

2022. 04. 13.

Készítette:

Závodszki Máté
Mérnökinformatikus
B2C7B0

Miskolc, 2022

Feladatok

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

MAX. IGÉNY				FOGLAL				KIELÉGÍTETLEN IGÉNYEK			
	R1	R2	R3		R1	R2	R3	R1	R2	R3	
P0	7	5	3		0	3	0	7	2	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	
				foglaltak	7	4	5	készlet-giény			
				összesen	10	5	7	R1	R2	R3	
				Szabad erő	3	1	2	-5	1	-2	--
								1	3	0	Runnable
								-4	2	-1	--
								2	0	-1	--
								-2	3	0	--
MAX. IGÉNY				FOGLAL				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		3	0	2	0	2	0	
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	
				foglaltak	8	2	7	készlet-giény			
				összesen	10	5	7	R1	R2	R3	
				Szabad erő	2	3	0	-5	1	0	--
								2	3	0	Runnable
								-4	2	-1	--
								2	0	-1	--
								-2	3	0	--
MAX. IGÉNY				FOGLAL				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		3	3	2	1	0	1	
				foglaltak	10	5	5	készlet-giény			
				összesen	10	5	7	R1	R2	R3	
				Szabad erő	0	0	2	-5	1	-2	--
								1	3	0	Runnable
								-4	2	-1	--
								2	3	-1	--
								1	3	0	Runnable

2, 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.

```
main.c X
18 int msgid;
19 key_t key;
20 int msgflg;
21 int rtn, msgsz;
22
23 key = MSGKEY;
24 msgflg = 00666 | IPC_CREAT;
25 msgid = msgget(key, msgflg);
26 if (msgid == -1) {
27     perror("\n Az msgget rendszerhívás sikertelen!");
28     exit(-1);
29 }
30 printf("\n Az msgid %d, %x : ", msgid, msgid);
31
32
33
34
35 msgp = msgndbuf;
36 msgp->mtype = 1;
37 strcpy(msgp->mtext, "Az első üzenet");
38 msgsz = strlen(msgp->mtext) + 1;
39
40
41
42 rtn = msgsnd(msgid, (struct msgbuf *) msgp, msgsz, msgflg);
43 printf("\n Az 1. msgsnd visszaszolgált %d-t", rtn);
44 printf("\n A kiküldött üzenet: %s", msgp->mtext);
45
46
47 strcpy(msgp->mtext, "Másik üzenet");
48 msgsz = strlen(msgp->mtext) + 1;
49 rtn = msgsnd(msgid, (struct msgbuf *) msgp, msgsz, msgflg);
50 printf("\n Az 2. msgsnd visszaszolgált %d-t", rtn);
51 printf("\n A kiküldött üzenet: %s \n", msgp->mtext);
52
53 exit (0);
54
55
```

```
main.c X
10 char mtext[512];
11 } rcvbuf, "msgp;
12
13 struct msgid_ds ds, *buf;
14
15 int main()
16 {
17     int msgid;
18     key_t kulcs;
19     int tipus, msgflg;
20     int rtn, meret;
21
22     kulcs = MSGKEY;
23     msgflg = 00666 | IPC_CREAT | MSG_NOERROR;
24
25     msgid = msgget(kulcs, msgflg);
26     if (msgid == -1) {
27         perror("\n Az msgget rendszerhívás sikertelen!");
28         exit(-1);
29     }
30     printf("\n Az üzenet ID: %d", msgid);
31
32     msgp = rcvbuf;
33     buf = ds;
34     meret = 20;
35     tipus = 0;
36     rtn = msgctl(msgid, IPC_STAT, buf);
37     printf("\n Az üzenetek száma: %d", buf->msg_qnum);
38
39     while (buf->msg_qnum) {
40
41
42         rtn = msgrcv(msgid, (struct msgbuf *) msgp, meret, tipus, msgflg);
43         printf("\n Visszaszolgált: %d, A fogadott üzenet: %s \n", rtn, msgp->mtext);
44         rtn = msgctl(msgid, IPC_STAT, buf);
45
46     }
47     exit (0);
48
```

```
main.c X
1 #include <stdio.h>
2 #include <sys/types.h>
3 #include <sys/ipc.h>
4 #include <sys/msg.h>
5 #include <stdlib.h>
6 #define MSGKEY 654321L
7
8 int main()
9 {
10     int msgid, msgflg, rtn;
11     key_t kulcs;
12     kulcs = MSGKEY;
13     msgflg = 00666 | IPC_CREAT;
14     msgid = msgget(kulcs, msgflg);
15
16     rtn = msgctl(msgid, IPC_RMID, NULL);
17     printf ("\n Visszaszolgált: %d", rtn);
18
19     exit (0);
20
21
```

3, 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.

```
*main.c X
16 struct msqid_ds ds, *buf;
17
18
19
20
21 int main()
22 {
23     int msqid;
24     key_t key;
25     int msgflg;
26     int rtn, msgsz;
27
28     key = MSGKEY;
29     msgflg = 00666 | IPC_CREAT;
30     msqid = msgget(key, msgflg);
31     if (msqid == -1) {
32         perror("\n Az msqid rendszerhívás sikertelen!");
33         exit(-1);
34     }
35
36     msgp = sendbuf;
37     msgp->mtype = 1;
38     strcpy(msgp->mtext, "Az egyik üzenet");
39     msgsz = strlen(msgp->mtext) + 1;
40
41     rtn = msgsnd(msqid, (struct msgbuf *) msgp, msgsz, msgflg);
42
43     strcpy(msgp->mtext, "Másik üzenet");
44     msgsz = strlen(msgp->mtext) + 1;
45     rtn = msgsnd(msqid, (struct msgbuf *) msgp, msgsz, msgflg);
46
47     if (msgp->mtext == "exit") {
48         exit(0);
49     }
50
51
52     key_t kulcs;
53
```

4, Gyakorló feladat:

```
main.c X
1 #include <stdio.h>
2 #include <sys/types.h>
3 #include <sys/ipc.h>
4 #include <sys/shm.h>
5 #include <stdlib.h>
6 #define SHMKEY 13
7
8 int main()
9 {
10     int shmid;
11     key_t kulcs;
12     int meret = 512;
13     int flag;
14
15     kulcs = SHMKEY;
16     flag = 0;
17     if ((shmid = shmget(kulcs, meret, flag)) < 0)
18     {
19         flag = 00666 | IPC_CREAT;
20         if ((shmid = shmget(kulcs, meret, flag)) < 0)
21         {
22             perror("\n Sikertelen volt az shmget!\n");
23             exit(-1);
24         }
25     }
26     printf("Az azonosítók: %d\n", shmid);
27     return 0;
28 }
29
```

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <sys/types.h>
4  #include <sys/ipc.h>
5  #include <sys/shm.h>
6  #define SHMKEY 13
7
8  int main()
9  {
10     int shmid;
11     key_t kulcs;
12     int meret = 512;
13     int flag, rtn, sm;
14     struct shm_id ds shm_id, *buf;
15
16     buf = shm_id.ds;
17     kulcs = SHMKEY;
18     flag = 0;
19
20     if((shmid = shmget(kulcs, meret, flag)) < 0)
21     {
22         perror("\n Sikertelen volt az shmget!\n");
23         exit(-1);
24     }
25
26     |
27     rtn = shmctl(shmid, IPC_STAT, buf);
28     printf("\n Méret: %ld", buf->shm_segsize);
29     printf("\n PID: %d\n", buf->shm_lpid);
30
31     exit(0);
32 }

```

```

10  {
11     int shmid;
12     key_t kulcs;
13     int meret = 512;
14     int flag;
15
16     struct str {
17         int lng;
18         char text[512-sizeof(int)];
19     } *segment;
20
21     kulcs = SHMKEY;
22     flag = 0;
23
24     if((shmid = shmget(kulcs, meret, flag)) < 0)
25     {
26         perror("\n Sikertelen volt az shmget!\n");
27         exit(-1);
28     }
29
30     flag = 00666 | SHM_RND;
31     segment = (struct str *)shmat(shmid, NULL, flag);
32
33     if(segment == (void *) - 1)
34     {
35         perror("\n Sikertelen csatlakozás!");
36         exit(-1);
37     }
38
39     if(strlen(segment->text) > 0)
40     {
41         printf("\n A régi szöveg : %s", segment->text);
42     }
43
44     printf("\n Adja meg az új szöveget : ");
45     scanf("%s", segment->text);
46     printf("\n\n Az új szöveg : %s\n", segment->text);
47 }

```