JEGYZŐKÖNYV

Operációs rendszerek BSc

2021 tavasz féléves feladat

10. Gyakorlat

Készítette: **Érsek Norbert**

Neptunkód: **IIJU0Z**

1. **feladat leírása:** Adottak egy rendszerben az alábbi erőforrások: R (R1: 10; R2: 5; R3: 7)
   A rendszerben 5 processz van: P0, P1, P2, P3, P4.
   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.
   A feladat elkészítésének lépései: Semmi anyagot nem találtam róla

2. **feladat leírása:** 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 feladat elkészítésének lépései:

   - Létrehozzuk az első fájlt, aminek a fő parancsa a msgsnd() lesz, de erről majd később.
   - Szükségünk van egy struktúrára, ebben egy egész típusú és string típusú változóra. Ez fogja az üzenetet magában foglalni.
     ```
     struct msg_buffer{
             long msg_type;
             char msg_text[100];
     }message;
     ```
   - Létrehozzuk a szükséges változókat: A kulcsot, majd az üzenet azonosítóját. Az előbbivel fogjuk elérni a továbbiakban. Ezután az msgid megkapja az értékét.
     ```
     key_t key;
     int msgid;

     key = 2234;
     msgid = msgget(key, 0666 | IPC_CREAT);
     ```
   - Itt lesz a program fő része. msg_type megkapja az értékét, msg_text úgyszintén. Ezek után jön a msgsnd() függvény, aminek az első argumentuma az előbb létrehozott üzenet azonosító vagy msgid, Második maga az üzenet, majd annak a mérete és végül a flagek, ami jelenleg nulla.
     ```
     message.msg_type = 1;
     strcpy(message.msg_text,"hello");

     msgsnd(msgid, &message, sizeof(message), 0);
     ```
   - Létrehozzuk a második fájlt az üzenet olvasására.
   - Nagyrészt az elsőtől nem különbözik, viszont msgsnd() helyett msgrcv() lesz a használt függvényünk.
   - Ezért ebben fog különbözni.
     ```
     msgrcv(msgid, &rbuf, 100, rbuf.msg_type, 0);
     printf("%s\n", rbuf.msg_text);
     ```
     Láthatjuk, hogy az msgrcv() öt argumentummal rendelkezik, ezek közül az előzőhez hasonlóan az első az üzenet ID, majd az üzenet maga, ennek mérete

byte-ban, a típusa, ami egész szám és a flagek. Következő sorban pedig kiírjuk az üzenetet stringként.

- Végül az msgctl.c nevű fájlt készítjük el, ami jelentősen egyszerűbb a korábbiaknál, ugyanis négy sorból áll. Átadjuk neki az ID-t, majd az msgctl() -lel letöröljük a message queue-t. Ennek három argumentuma van: ID, parancs és egy buffer, ami jelenleg üres.

```
int msgid;
msgid= msgget(2234, 0666);
msgctl(msgid, IPC_RMID, NULL);
printf("Uzenet torolve\n");
```

A feladat futásának eredménye:

ipcs paranccsal megnézve egy message queue -t látunk, mielőtt lefuttatnánk az msgcreate programot.

```
------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages
0xffffffff 0          pi         666        312          3
```

Futás után kettőt:

```
------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages
0xffffffff 0          pi         666        312          3
0x000008ba 98305      pi         666        104          1
```

Az alsó a miénk.

Most jön az msgrcv, amivel megkapjuk az üzenetet.

```
pi@raspberrypi:~/Documents/OsGyak10 $ ./msgrcv
L▒▒~PL▒v▒▒~▒▒~
```

Valamit elírtam a kódban, vaslószínűleg, de megkapta az üzenetet. Message queue-ek száma változatlan.

Ezt követi a törlés a msgctl programmal. Futás után ismét csak egy message queue létezik.

```
pi@raspberrypi:~/Documents/OsGyak10 $ ./msgctl
Uzenet torolve
pi@raspberrypi:~/Documents/OsGyak10 $ ipcs

------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages
0xffffffff 0          pi         666        312          3
```

2. C
Két órája keresgélek és nem megy.

3.

Szinte biztos, hogy megbukok ebből a tárgyból.