Operációs rendszerek BSc

10.Gyak

2022.04.11.

Készítette:

Bodnár László BSc

Szak:
Mérnökinformatikus

Neptunkód: D1H8Vp

2022.04.11.

Feladatok

„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?”

	Maximális igény:			Kielegítéshez szükséges igények		
	R1	R2	R3	R1	R2	R3
P0	7	5	3	7	4	3
P1	3	2	2	1	2	2
P2	9	0	2	6	0	0
P3	2	2	2	0	1	1
P4	4	3	3	4	3	1

	R1	R2	R3	
Igény:	-4	-1	-1	P0
	2	1	0	P1
	-3	3	2	P2
	3	2	1	P3
	-1	0	1	P4

	Foglalas		
	R1	R2	R3
	0	1	0
	2	0	0
	3	0	2
	2	1	1
	0	0	2
foglalt:	7	2	5
Összes erőf	10	5	7
Szabad erőf	3	3	2

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.

Mentés: neptunkod_unnamed.c

```c
#include <sys/wait.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>

int main() {
    int pipefd[2];
    pid_t cpid;
    char buf;
    char szoveg[32];
    int ret;

    if (pipe(pipefd) == -1) {
      perror("pipe");
      exit(-1);
    }

    printf("%d: fd1: %d, fd2: %d\n",getpid(),pipefd[0],pipefd[1]);

    cpid = fork();
    if (cpid == -1) {
      perror("fork");
      exit(-1);
    }

    if (cpid == 0) {
      printf("%d: children\n",getpid());
      close(pipefd[1]);

      printf("%d: what's in the pipe\n%d: ",getpid(),getpid());
      while (read(pipefd[0], &buf, 1) > 0) {
        printf("%c",buf);
      }
      printf("\n%d:the pipes other side is closed\n",getpid());
```

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. Mentés:

C: > Users > Bodnár László > Desktop > os2 > C D1H8VP_named.c > main()

```c

#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/file.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>

int main()
{
    int fd, ret;
    char buf[32];

    buf[0]=0;


        ret=mkfifo("fifo",00666);
    if (ret == -1) {
        perror("mkfifo()");
        exit(-1);
    }

    fd=open("fifo",O_RDWR);
    if (fd == -1) {
        perror("open() error!");
        exit(-1);
    }

    strcpy(buf," in  fifo\0");
    printf(" fifo: %s:%d\n",buf,strlen(buf));
    write(fd,buf,strlen(buf));
```

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.

C: > Users > Bodnár László > Desktop > os2 > C msgcreate.c > main()

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

    key = MSGKEY;
    msgflg = 00666 | IPC_CREAT;
    msgid = msgget( key, msgflg);
    if ( msgid == -1) {
                perror("\n A msgget rendszer hivas sikertelen!");
                exit(-1);
        }
    printf("\n Az msgid %d, %x : ", msgid,msgid);

    msgp        = &sndbuf;
    msgp->mtype = 1;
    strcpy(msgp->mtext," uzenet1");
    msgsz       = strlen(msgp->mtext) + 1;
    /* es elkuldom: */
        rtn = msgsnd(msgid,(struct msgbuf *) msgp, msgsz, msgflg);
    printf("\n Az 1. msgsnd visszaadott %d-t", rtn);
    printf("\n Az uzenet:%s", msgp->mtext);

    strcpy(msgp->mtext," uzenet2");
    msgsz       = strlen(msgp->mtext) + 1;
    rtn = msgsnd(msgid,(struct msgbuf *) msgp, msgsz, msgflg);
    printf("\n A 2.  msgsnd visszaadott %d-t", rtn);
    printf("\n Az uzenet:%s", msgp->mtext);
    printf("\n");

    exit (0);
}
```

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

main()
{
    int msgid, msgflg,  rtn;
    key_t key;
    key = MSGKEY;
    msgflg = 00666 | IPC_CREAT;
    msgid = msgget( key, msgflg);

    rtn = msgctl(msgid, IPC_RMID, NULL);
    printf ("\n Vissza: %d", rtn);

    exit (0);
}
```

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

struct msqid_ds ds, *buf;


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

    key = MSGKEY;
    msgflg = 00666 | IPC_CREAT | MSG_NOERROR;

    msgid = msgget( key, msgflg);
    if ( msgid == -1) {
            perror("\n A msget rendszer hivas sikertelen volt!");
            exit(-1);
        }
    printf("\n Az msgid: %d",msgid);

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

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.

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#define KEY 123456L



main()
{
    int hallgato;
    key_t key;
    int size=512;
    int shmflg;

    key = KEY;

    shmflg = 0;
    if ((hallgato=shmget( key, size, shmflg)) < 0) {
       printf("\n Nincs szegmens! ");
       shmflg = 00666 | IPC_CREAT;
       if ((hallgato=shmget( key, size, shmflg)) < 0) {
          perror("\n Az shmget system call sikertelen!");
          exit(-1);
       }
    } else printf("\n Van  szegmens!");

    printf("A hallgato azonositoja a kovetkezo %d: \n", hallagato);

    exit (0);
}
```

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).

```c
int hallagto;
key_t key;
int size=512;
int shmflg;
int rtn;
int cmd;
struct hallgato_ds hallgato_ds, *buf;

buf = &hallgato_ds;

key = KEY;
shmflg = 0;
if ((hallgato=shmget( key, size, shmflg)) < 0) {
    perror("\n Az shmget system call sikertelen");
    exit(-1);
}


do {
printf("\n Add meg a parancs szamat ");
printf("\n 0 IPC_STAT  ");
printf("\n 1 IPC_RMID  ");
scanf("%d",&cmd);
} while (cmd < 0 && cmd > 1);

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

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)”

A futtatás eredményét is tartalmazza a jegyzőkönyv.

```c
int hallgato;
key_t key;
int size=512;
int shmflg;
struct vmi {
    int  hossz;
    char szoveg[512-sizeof(int)];
} *segm;

key = KEY;
shmflg = 0;
if ((hallgato=shmget( key, size, shmflg)) < 0) {
    perror("\n Az shmget system call sikertelen");
    exit(-1);
}


shmflg = 00666 | SHM_RND;
segm = (struct vmi *)shmat(hallgato, NULL, shmflg);
if (segm == (void *)-1) {
    perror(" Sikertelen attac");
    exit (-1);
}


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

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