

Operációs rendszerek BSc

12. Gyak.

2022. 04. 25.

Készítette:

Bodnár LászlóBsc
Mérnökinformatikus
D1H8VP

Miskolc, 2022

1. feladat – Adott egy rendszer (foglalási stratégiák), melyben a következő

- Szabad területek: 30k, 35k, 15k, 25k, 75k, 45k és
- Foglalási igények: 39k, 40k, 33k, 20k, 21k állnak rendelkezésre.

A rendszerben a memória 4 kbyte-os blokkokban kerül nyilvántartásra, ennél kisebb méretű töredék igény esetén a teljes blokk lefoglalásra kerül. Határozza meg változó méretű partíció

[illegible]

feladat – A feladat megoldásához először tanulmányozza Vadász Dénes: Operációs rendszer jegyzet, a témához kapcsolódó fejezetét (6.4)., azaz Írjon C nyelvű programokat, ahol

- kreál/azonosít szemafor készletet, benne N szemafor-t. A kezdő értéket 0-ra állítja – semset.c,

```

1  ✓ #include <stdio.h>
2    #include <sys/types.h>
3    #include <sys/ipc.h>
4    #include <sys/sem.h>
5    #include <stdlib.h>
6    #define KEY 123456L
7
8  ✓ union semun {
9      int val;                /* Value for SETVAL */
10     struct semid_ds *buf;    /* Buffer for IPC_STAT, IPC_SET */
11     unsigned short *array;   /* Array for GETALL, SETALL */
12     struct seminfo *__buf;   /* Buffer for IPC_INFO (Linux-specific) */
13 };
14
15 ✓ void main() {
16     union semun arg;
17
18     int n = 5;
19     int semID = semget(KEY, n, IPC_CREAT | 0666);
20
21 ✓   if (semID == -1)
22   {
23       perror("Nem sikerult szemaforokat létrehozni");
24       exit(-1);
25   }
26
27     arg.array = (short *)calloc(n, sizeof(int));
28
29 ✓   if (semctl(semID, 0, SETALL, arg))
30   {
31       perror("Nem sikerult beallitani az erteket\n");
32       exit(-1);
33   }
34

```

- kérdezze le és írja ki a pillanatnyi szemafor értéket – semval.c

```

1  #include <stdio.h>
2  #include <sys/types.h>
3  #include <sys/ipc.h>
4  #include <sys/sem.h>
5  #include <stdlib.h>
6  #define KEY 123456L
7
8  union semun {
9      int val; /* Value for SETVAL */
10     struct semid_ds *buf; /* Buffer for IPC_STAT, IPC_SET */
11     unsigned short *array; /* Array for GETALL, SETALL */
12     struct seminfo *__buf; /* Buffer for IPC_INFO (Linux-specific) */
13 };
14
15 void main() {
16
17     int semID = semget(KEY, 0, 0);
18     int n = 5;
19     if (semID == -1)
20     {
21         perror("Nem sikerult szemaforokat lekerdezni\n");
22         exit(-1);
23     }
24
25     union semun arg;
26
27     printf("Szemaforok tartalma: \n");
28     arg.array = (short *)calloc(n, sizeof(int));
29
30     semctl(semID, 0, GETALL, arg);
31
32     for (int i = 0; i < n; i++)
33     {
34         printf("%d \n", arg.array[i]);
35     }

```

- szüntesse meg a példácskák szemafor készletét – semkill.c

```

#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdlib.h>
#define KEY 123456L

void main() {
    int n = 5;
    int semID = semget(KEY, 0, 0);
    if (semID == -1) {
        perror("Nem sikerult szemaforokat lekerdezni\n");
        exit(-1);
    }

    for (int i = 0; i < n; i++)
        semctl(semID, i, IPC_RMID);
}

```

- sembuf.sem_op=1 értékkel inkrementálja a szemaforot – semup.c

```

1  #include <stdio.h>
2  #include <sys/types.h>
3  #include <sys/ipc.h>
4  #include <sys/sem.h>
5  #include <stdlib.h>
6  #define KEY 123456L
7
8  void main() {
9      int semID = semget(KEY, 0, 0);
10     if (semID == -1) {
11         perror("Nem sikerult szemaforokat lekerdezni\n");
12         exit(-1);
13     }
14
15     struct sembuf buffer;
16
17     buffer.sem_num = 4;
18     buffer.sem_op = 1;
19     buffer.sem_flg = 0666;
20
21     if (semop(semID, &buffer, 1)) {
22         perror("Sikertelen\n");
23         exit(-1);
24     }
25 }

```

2a. feladat – a. Írjon egy C nyelvű programot, melyben

- egyik processz létrehozza a szemafort (egyetlen elemi szemafort; inicializálja 1-re, vagy x-re, ha még nem létezik),
- másik processz használja a szemafort, belépési szakasz (down), a kritikus szakaszban alszik 2-3 sec-et, m pid-et kiír, kilépési szakasz (up), ezt ismételve 2x-

3x (és a hallgató egyszerre indítson el 2-3 ilyen processzt),

```
4  #include <sys/sem.h>
5  #include <stdlib.h>
6  #include <errno.h>
7  #include <unistd.h>
8
9  #define KEY 77777L
10
11 union semun {
12     int val; /* Value for SETVAL */
13     struct semid_ds *buf; /* Buffer for IPC_STAT, IPC_SET */
14     unsigned short *array; /* Array for GETALL, SETALL */
15     struct seminfo *__buf; /* Buffer for IPC_INFO (Linux-specific) */
16 };
17
18 void main() {
19     union semun arg;
20
21     int semID = semget(KEY, 0, 0);
22     if (errno == ENOENT)
23     {
24         semID = semget(KEY, 1, IPC_CREAT | 0666);
25         printf("Szam: ");
26         scanf("%d", &(arg.val));
27     }
28     else
29     {
30         arg.val = 1;
31     }
32
33     semctl(semID, 0, SETVAL, arg);
34
35     printf("A szemafor erteke (1) : %d\n", semctl(semID, 0, GETVAL));
36
37 }
```

```

void main()
{
    int semID = semget(KEY, 0, 0);

    if (semID == -1)
    {
        perror("Nem sikerult megnyitni\n");
        exit(-1);
    }

    //belepesi szakasz
    printf("Kritikus szakasz\n");
    down(semID);
    sleep(3);
    printf("pid : %d\n", getpid());
    printf("%d \n", semctl(semID, 0, GETVAL));
    up(semID);
    printf("kritikus szakasz vege\n");
}

void up(int semId) {
    struct sembuf buffer;
    buffer.sem_num = 0;
    buffer.sem_op = 1;
    buffer.sem_flg = 0;

    semop(semId, &buffer, 1);
}

void down(int semId) {
    struct sembuf buffer;
    buffer.sem_num = 0;
    buffer.sem_op = -1;
    buffer.sem_flg = 0;

```

- harmadik processzben, ha létezik a szemafor, akkor megszünteti”.

```
1  ✓ #include <stdio.h>
2    #include <sys/types.h>
3    #include <sys/ipc.h>
4    #include <sys/sem.h>
5    #include <stdlib.h>
6    #include <errno.h>
7    #include <unistd.h>
8
9    #define KEY 77777L
10
11  ✓ void main() {
12      int semID = semget(KEY, 0, 0);
13
14  ✓    if (semID == -1)
15      {
16          perror("Nem sikerult megnyitni\n");
17          exit(-1);
18      }
19
20  ✓    if (semctl(semID, 0, IPC_RMID) == -1)
21      {
22          perror("Nem sikerult torolni\n");
23          exit(-1);
24      }
25
26      printf("Torolve\n");
27
28  }
29
```


