Operációs rendszerek BSc

Beadandó

2022. 05. 11.

Készítette:
Pogácsás Benedek Bsc
mérnökinformatikus
FM4Z3B

Miskolc, 2022

Az én algoritmus feladatom: 12.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1															
2	FIFO	6	8	3	8	6	0	3	6	3	5	3	5	6	
3	1. lap	6	6	6			0		0		0	3			
4	2. lap		8	8			8		6		6	6			
5	3. lap			3			3		3		5	5			
6	Laphiba	*	*	*			*		*		*	*			
7	Össz. Laphiba: 7														
8															
9															
10	SC	6	8	3	8	6	0	3	6	3	5	3	5	6	
11	1. lap	6 (1)	6 (1)	6 (1)	6 (1)	6 (1)	6 (0)	6 (0)	6 (1)	6 (1)	6 (0)	6 (0)	6 (0)	6 (1)	
12	2. lap		8 (1)	8 (1)	8 (1)	8 (1)	8 (0)	3 (1)	3 (1)	3 (1)	3 (0)	3 (1)	3 (1)	3 (1)	
13	3. lap			3 (1)	3 (1)	3 (1)	0 (1)	0 (1)	0 (1)	0 (1)	5 (1)	5 (1)	5 (1)	5 (1)	
14	Laphiba	*	*	*			*	*			*				
15	Össz. Laphiba: 6														
16															
17															
18	FIFO	6	8	3	8	6	0	3	6	3	5	3	5	6	
19	1. lap	6	6	6			6				5			5	
20	2. lap		8	8			8				8			6	
21	3. lap			3			3				3			3	
22	4. lap						0				0			0	
23	Laphiba	*	*	*			*				*			*	
24	Össz. Laphiba: 6														
25															
26															
27	SC	6	8	3	8	6	0	3	6	3	5	3	5	6	
28	1. lap	6 (1)	6 (1)	6 (1)	6 (1)	6 (1)	6 (1)	6 (1)	6 (1)	6 (1)	6 (0)	6 (0)	6 (0)	6 (1)	
29	2. lap		8 (1)	8 (1)	8 (1)	8 (1)	8 (1)	8 (1)	8 (1)	8 (1)	5 (1)	5 (1)	5 (1)	5 (1)	
30	3. lap			3 (1)	3 (1)	3 (1)	3 (1)	3 (1)	3 (1)	3 (1)	3 (0)	3 (1)	3 (1)	3 (1)	
31	4. lap						0 (1)	0 (1)	0 (1)	0 (1)	0 (0)	0 (0)	0 (0)	0 (0)	
32	Laphiba	*	*	*			*				*				
33	Össz. Laphiba: 5														
34															

Az én ipc feladatom: 25.

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>

#define SHMKEY 42073

void shmCreate(int *mem, int size);
void shmRead(int *mem, int size);
void shmWrite(int *mem, int size);
void shmDelete(int *mem);

int main(int argc, char *argv[])
{
    int count = 1;
    char *tomb[] = {"create", "read", "write", "delete"};
    int mem;
    int size = 512;

    while (count < argc) {
        if (strcmp(tomb[0], argv[count]) == 0){
            printf("- Create -\n");
            shmCreate(&mem, size);
        }
        else if (strcmp(tomb[1], argv[count]) == 0) {
            printf("- Read -\n");
            shmCreate(&mem, size);
            shmRead(&mem, size);
        }
        else if (strcmp(tomb[2], argv[count]) == 0) {
            printf("- Write -\n");
            shmCreate(&mem, size);
            shmWrite(&mem, size);
        }
        else if (strcmp(tomb[3], argv[count]) == 0) {
            printf("- Delete -\n");
            if ((mem=shmget(SHMKEY, size, 0)) < 0)
                printf("Ez a szegmens nem is letezik! \n");
            else
                shmDelete(&mem);
        }
        else {
            printf("- Nincs ilyen parancs -\n");
        }
        count++;
    }

    exit(0);
}
```

```c
void shmCreate(int *mem, int size) {

    key_t key;
    int shmflg;

    key = SHMKEY;

    shmflg = 0;
    if ((*mem=shmget(key, size, shmflg)) < 0) {
       printf("A szegmens meg nem letezik! El kell keszitени! \n");
       shmflg = 01747 | IPC_CREAT;

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

void shmRead(int *mem, int size) {

    int shmflg;
    struct readstruct {
        int  hossz;
        char szoveg[size-sizeof(int)];
    } *segm;

    shmflg = 00327 | SHM_RND;
    segm = (struct readstruct *)shmat(*mem, NULL, shmflg);

    if (strlen(segm->szoveg) > 0)
        printf("A memory-n levo szoveg: %s\n",segm->szoveg);
    else
        printf("A memory-n nincs szoveg\n");

    shmdt(segm);
}
```

```c
void shmWrite(int *mem, int size) {

    int shmflg;
    struct writestruct {
        int  hossz;
        char szoveg[size-sizeof(int)];
    } *segm;

    shmflg = 0327 | SHM_RND;
    segm = (struct writestruct *)shmat(*mem, NULL, shmflg);

    printf("A memoriara iras: \n");
    gets(segm->szoveg);
    printf("Az uj szoveg: %s\n",segm->szoveg);
    segm->hossz=strlen(segm->szoveg);

    shmdt(segm);
}

void shmDelete(int *mem) {

    shmctl(*mem, IPC_RMID, NULL);
    printf("Szegmens torolve.\n");
}
```