JEGYZŐKÖNYV

Operációs rendszerek BSc

2022. tavasz féléves feladat

Készítette: Nyíri Dániel

Neptunkód: AUGHMI

Feladat leírása (IPC):

Írjon C nyelvű programokat, ami:

SIGUSR1 signal érkeztekor növeli egy változó értékét.

SIGUSR2 signal hatására pedig kiírja a képernyőre az aktuális értéket.

A másik program pedig: futtatáskor signal-t küld az adott processznek.

Feladat elkészítésének lépései:

- 1. pid lekérdezése
- 2. error ellenőrzés
- 3. amíg nem billen ált a flag
- 4. signal ellenőrzés
 - a. ha SIGUSR1, változót növelem
 - b. ha SIGUSR2, kiírom a változót és átbillentem a flag-et

```
void handlerSignals(int sig);
int flag = 0;
int signals = 0;
int main()
{
    printf("PID = %d\n", getpid());
    if(signal(SIGUSR1, handlerSignals) == SIG_ERR) {
        printf("[SIGUSR1] : ERROR\n");
        return 1;
    }
    if(signal(SIGUSR2, handlerSignals) == SIG_ERR) {
        printf("[SIGUSR2] : ERROR\n");
        return 1;
    }
    while(flag != 1) {
        pause();|
    }
    return 0;
}

void handlerSignals(int sig) {
    if(sig == SIGUSR1) {
        signals++;
        printf("SIGUSR2 signal recieved!\n");
    }
    else if(sig == SIGUSR2) {
        printf("NSIGUSR2 signal: %d\n", signals);
        signal(SIGUSR2, SIG_DFL);
        flag = 1;
    }
}
```

Futtatás eredménye:

```
nydani@LinuxMint ~/Dokumentumok/OS/Beadando - © ©

Fájl Szerkesztés Nézet Keresés Terminál Súgó

nydani@LinuxMint ~/Dokumentumok/OS/Beadando $ ./AUGHMI_IPC
PID = 3348
SIGUSR1 signal recieved!
SIGUSR2 signal: 5
nydani@LinuxMint ~ $ kill -SIGUSR1 3348
nydani@LinuxMint ~ $ kill -SIGUSR2 3348
nydani@LinuxMint ~ $ kill -SIGUSR1 348
nydani@LinuxMint ~ $ kill
```

Feladat leírása (Algoritmus):

Adott egy *igény szerinti lapozást* használó számítógéprendszer, melyben futás közben egy processz számára a következő laphivatkozással lehet hivatkozni: 6, 8, 3, 8, 6, 0, 3, 6, 3, 5, 3, 6.

Memóriakeret: 3, illetve 4 memóriakeret.

Készítse el a laphivatkozások betöltését külön-külön táblázatba 3, ill. 4 memóriakeret esetén.

Mennyi laphiba keletkezik az alábbi algoritmusok esetén: FIFO, SC?

Feladat elkészítésének lépései:

FIFO: a behozott lapok számát egy FIFO sorban tároljuk, laphiba esetén a FIFO elején álló lapot lecseréljük, a behozott lapot a FIFO végére írjuk.

SC: minden laphoz tartozik egy hivatkozási bit (*0 vagy 1*). Lapcsere esetén azt a lapot cseréljük ki, amelyik a legrégebben van bent, de ha a kicserélendő lap hivatkozási bitje 1, akkor adunk neki még egy esélyt, azaz a bitet 0-ra állítjuk, betesszük a FIFO végére és új lapot választunk.

Futtatás eredménye:

FIFO	Laphivatkozások											
Memóriakeret	6	8	3	8	6	0	3	6	3	5	3	6
1. lap	6	6	6			0		0		0	3	
2. lap		8	8			8		6		6	6	
3. lap			3			3		3		5	5	
Laphiba:	*	*	*			*		*		*	*	
FIFO:	6	8	3	0	6	5	3					
Hibák:	3 + 4											

SC		Laphivatkozások																	
Memóriaker	et 6	, [8	3	8	6	0				3	6	3	5				3	6
1. l	ap 6	1	6 ¹	6 ¹			6 ⁰	6 ⁰	6 ⁰	01	01	01		0 0	0 0	0 0	5 ¹	5 ¹	5 ¹
2. I	ар		8 ¹	8 ¹			8 ¹	8 ⁰	8 ⁰	8 ⁰	8 0	6 ¹		6 ¹	6 °	6 ⁰	6 º	6 ⁰	6 ¹
3. I	ар			3 ¹			3 ¹	3 ¹	3 ⁰	3 ⁰	3 ¹	3 ¹		3 ¹	3 ¹	3 ⁰	3 ⁰	3 ¹	3 ¹
Laphiba:	*		*	*			*							*					
FIFO:	6		8	3	6	8	3	0	3	6	0	6	3	5	3	6			
Hibák:	3 +	3																	

FIFO	Laphivatkozások														
Memóriakeret	6	8	3	8	6	0	3	6	3	5	3	6			
1. lap	6	6	6			6				5		5			
2. lap		8	8			8				8		6			
3. lap			3			3				3		3			
4. lap						0				0		0			
Laphiba:	*	*	*			*				*		*			
FIFO:	6	8	3	0	5	6									
Hibák:	4+3														

sc		Laphivatkozások														
Memóriakeret	6	8	3	8	6	0	3	6	3	5					3	6
1. lap	6 ¹	6 ¹	6 ¹			6 ¹				6 ⁰	6 °	6 ⁰	6 °	5 ¹	5 ¹	5 ¹
2. lap		8 ¹	8 ¹			8 ¹				8 ¹	8 ⁰	8 ⁰	8 ⁰	8 ⁰	8 °	6 ¹
3. lap			3 ¹			3 ¹				3 ¹	3 ¹	3 0	3 ⁰	3 0	3 ¹	3 ¹
4. lap						0 1				01	01	01	0 0	0 0	0 0	0 0
Laphiba:	*	*	*			*				*						*
FIFO:	6	8	3	0	6	8	3	0	5	3	6					
Hibák:	4 + 2															