

Operációs Rendszerek

7.gyak

2021.03.24.

Készítette:

Palkó Patrik Dávid

Programtervező – informatikus

ZW7DOR

Miskolc, 2021

1. feladat - Adott négy processz a rendszerbe, melynek beérkezési sorrendje: A, B, C és D. Minden processz USER módban fut és mindegyik processz futásra kész. Kezdetben mindegyik processz: $p_uspri = 60$. Az A, B, C processz: $p_nice = 0$, a D processz: $p_nice = 5$. Mindegyik processz: $p_cpu = 0$, az óraütés 1 indul, a befejezés legyen 201. óraütés-ig

a.) Határozza meg az ütemezést RR nélkül és az ütemezést RR-nal - külön-külön táblázatba.

b.) Minden óraütem esetén határozza meg a processzek sorrendjét óraütés előtt/után.

c.) Igazolja a számítással a tanultak alapján.

RR nélkül:

	A process		B process		C process		D process		Reschedule	
Clock tick	p_uspri	p_cpu	p_uspri	p_cpu	p_uspri	p_cpu	p_uspri	p_cpu	running before	running after
Starting point	60	0	60	0	60	0	60	0	A	A
1	60	1	60	0	60	0	60	0	A	A
...	A	A
99	60	99	60	0	60	0	60	0	A	A
100	73	50	60	0	60	0	60	0	A	B
101	73	50	60	1	60	0	60	0	B	B
...		
199	73	50	60	99	60	0	60	0	B	B
200	66	25	73	50	60	0	60	0	B	C
201	66	25	73	50	60	1	60	0	C	C

$$p_cpu = 100/0,5 = 50$$

$$p_uspri(1) = P_USER + 50 / 4 + 2 * p_nice = 73$$

$$p_uspri(2) = P_USER + 25 / 4 + 2 * p_nice = 66$$

RR-el:

Clock tick	A process		B process		C process		D process		Reschedule	
	p_uspri	p_cpu	p_uspri	p_cpu	p_uspri	p_cpu	p_uspri	p_cpu	running before	running after
Starting point	60	0	60	0	60	0	60	0	A	A
1	60	1	60	0	60	0	60	0	A	A
...	A	A
9	60	9	60	0	60	0	60	0	A	A
10	60	10	60	0	60	0	60	0	A	B
...
19	60	10	60	9	60	0	60	0	B	B
20	60	10	60	10	60	0	60	0	B	C
...
29	60	10	60	10	60	9	60	0	C	C
30	60	10	60	10	60	10	60	0	C	D
...
39	60	10	60	10	60	10	60	9	D	D
40	60	10	60	10	60	10	60	10	D	A
50	60	20	60	10	60	10	60	10	A	B
60	60	20	60	20	60	10	60	10	B	C
70	60	20	60	20	60	20	60	10	C	D
80	60	20	60	20	60	20	60	20	D	A
90	60	30	60	20	60	20	60	20	A	B
100	67	26	67	26	64	17	64	27	B	C
...
199	67	46	67	46	64	37	64	46	D	D
200	70	39	70	39	68	31	70	40	D	A
201	70	40	70	39	68	31	70	40	A	A

100. óráütés:

$$KF = 2 * FK / (2 * FK + 1) = (2 * 3) / (2 * 3 + 1) = 0,85$$

$$A \text{ p_cpu} = 30 * 0,85 = 26$$

$$A \text{ p_uspri} = 60 + (26/4) = 67$$

$$B \text{ p_cpu} = 30 * 0,85 = 26$$

$$B \text{ p_uspri} = 60 + (26/4) = 67$$

$$C \text{ p_cpu} = 20 * 0,85 = 17$$

$$C \text{ p_uspri} = 60 + (17/4) = 64$$

$$D \text{ p_cpu} = 20 * 0,85 = 17$$

$$D \text{ p_uspri} = 60 + (17/4) + 10 = 74$$

200. óráütés

$$KF = 2 * FK / (2 * FK + 1) = (2 * 3) / (2 * 3 + 1) = 0,85$$

$$A \text{ p_cpu} = 30 * 0,85 = 39 \quad A \text{ p_uspri} = 60 + (26/4) = 70$$

$$B \text{ p_cpu} = 30 * 0,85 = 39 \quad B \text{ p_uspri} = 60 + (26/4) = 70$$

$$C \text{ p_cpu} = 20 * 0,85 = 31 \quad C \text{ p_uspri} = 60 + (17/4) = 68$$

$$D_{p_cpu} = 20 * 0,85 = 40 \quad D_{p_uspri} = 60 + (17/4) + 10 = 70$$

2. feladat - A tanult rendszerhívásokkal (open(), read()/write(), close()) - ők fogják a rendszerhívásokat tovább hívni.) írjanak egy neptunkod_openclose.c programot, amely megnyit egy fájlt – neptunkod.txt, tartalma: hallgató neve, szak, neptunkod.

A program következő műveleteket végezze:

- olvassa be a neptunkod.txt fájlt, melynek attribútuma: O_RDWR hiba ellenőrzést,
- write() - mennyit ír ki a konzolra.
- read() - kiolvassa a neptunkod.txt tartalmát és mennyit olvasott ki (byte), és kiírja konzolra.
- lseek() – pozícionálja a fájl kurzor helyét, ez legyen a fájl eleje: SEEK_SET, és kiírja a konzolra.

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

#define FILE "ZW7D0R.txt"

int main() {
    int fileHandle = open(FILE, O_RDWR);
    if(fileHandle == -1)
    {
        perror("Nem sikerulta fajlt megnyitni!");
        return 1;
    } else
    {
        printf("Fajl megnyitva!");
        char tartalom[64];

        int olvasott = read(fileHandle, tartalom, sizeof(tartalom));
        printf("beolvasott tartalom: \"%s\" osszesen \"%i\" byte.\n", tartalom, olvasott);

        lseek(fileHandle, 0, SEEK_SET);

        char szoveg[] = "keksz";
        int irt = write(fileHandle, szoveg, sizeof(szoveg));
        printf("A fajlba irtuk a(z) \"%s\" szoveget. osszesen \"%i\" byte.\n", szoveg, irt);
        close(fileHandle);
        return 0;
    }
}
```