

Operációs rendszerek BSc

10. Gyak.

2022. 04. 11.

Készítette:

Kórád György Bsc
Programtervező Informatikus
ZF440N

Miskolc, 2022

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?

Az összes osztály -erőforrások száma: (10,5,7)											
Kiinduló állapot											
1. Lépés				2. Lépés				3. Lépés			
MAX IGÉNY				FOGLAL				IGÉNY = MAX IGÉNY - FOGLAL			
	R1	R2	R3		R1	R2	R3		R1	R2	R3
P0	7	5	3	P0	0	1	0	P0	7	4	3
P1	3	2	2	P1	2	0	0	P1	1	2	2
P2	9	0	2	P2	3	0	2	P2	6	0	0
P3	2	2	2	P3	2	1	1	P3	0	1	1
P4	4	3	3	P4	0	0	2	P4	4	3	1
MAXr = (10, 5, 7)											
Szabad = (10, 5, 7) - (7, 2, 5) = (3, 3, 2)											
P4 Szabad = [3, 3, 2] + [3, 3, 0] = [6, 6, 2]											
P0 Szabad = [6, 6, 2] + [0, 2, 0] = [6, 8, 2]											

A P4 processzre igaz, hogy kevesebb erőforrást kér, mint ami szabad. A P0 processz azonban nem teljesül, ezért a rendszer nem lesz biztonságos.

2. Készítsen C nyelvű programot, ahol egy szülő processz létrehoz egy csővezetékét, 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

```
int main() {
    int fd[2];
    int child;

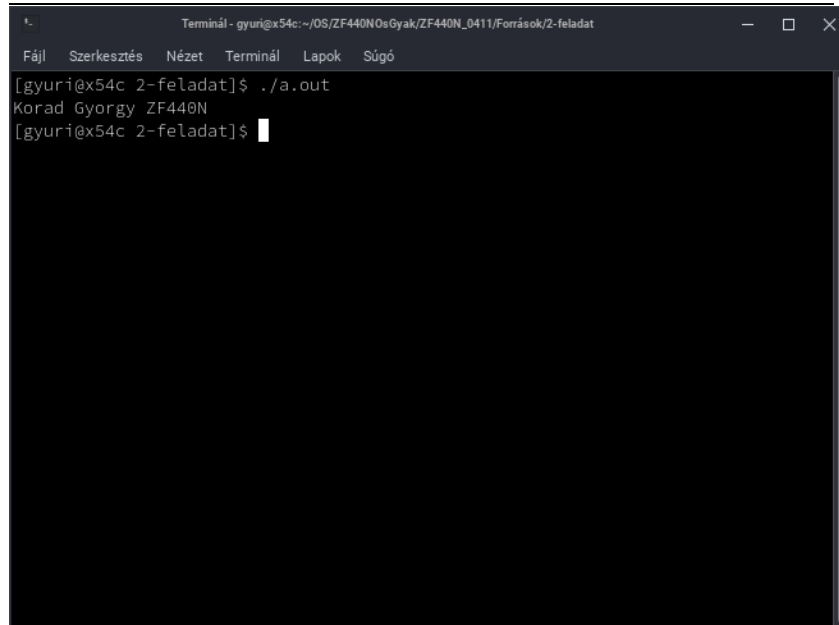
    if(pipe(fd)) {
        perror("pipe");
        return 1;
    }

    child = fork();

    if(child > 0) {
        char s[1024];
        close(fd[1]);
        read(fd[0], s, sizeof(s));
        printf("%s", s);

        close(fd[0]);
    } else if (child == 0) {
        close(fd[0]);
        write(fd[1], "Korad Gyorgy ZF440N\n", 21);
        close(fd[1]);
    }

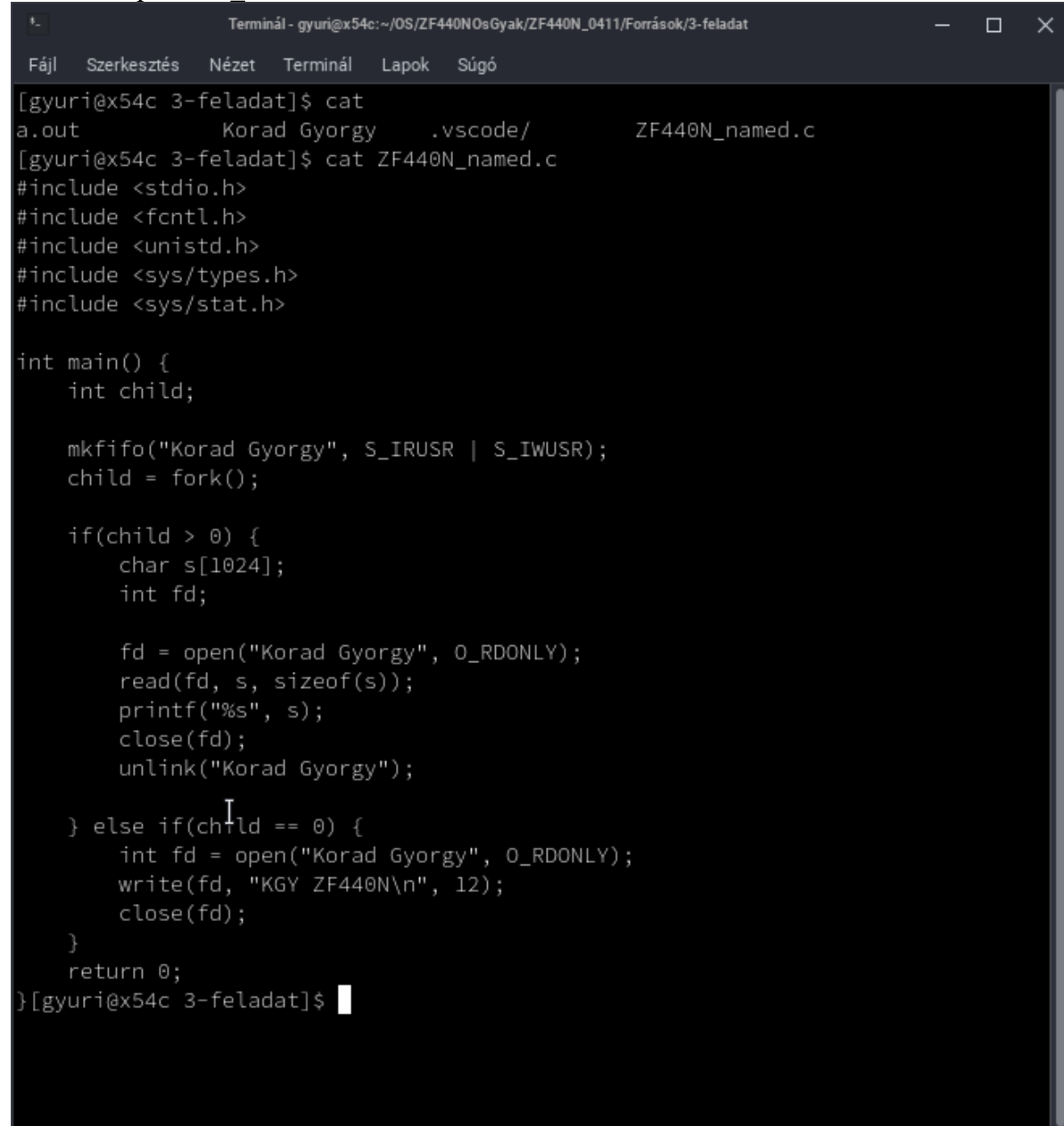
    return 0;
}
```



```
Terminál - gyuri@x54c: ~/OS/ZF440N0sGyak/ZF440N_0411/Forrasok/2-feladat
Fájl Szerkesztés Nézet Terminál Lapok Súgó
[gyuri@x54c 2-feladat]$ ./a.out
Korad Gyorgy ZF440N
[gyuri@x54c 2-feladat]$
```

3. Készítsen C nyelvű programot, ahol egy szülő processz létrehoz egy nevesített csővezetékét (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: neptunkod_named.c



```
Terminál - gyuri@x54c:~/OS/ZF440N0sGyak/ZF440N_0411/Forrasok/3-feladat
Fájl Szerkesztés Nézet Terminál Lapok Súgó

[gyuri@x54c 3-feladat]$ cat
a.out          Korad Gyorgy      .vscode/      ZF440N_named.c
[gyuri@x54c 3-feladat]$ cat ZF440N_named.c
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>

int main() {
    int child;

    mkfifo("Korad Gyorgy", S_IRUSR | S_IWUSR);
    child = fork();

    if(child > 0) {
        char s[1024];
        int fd;

        fd = open("Korad Gyorgy", O_RDONLY);
        read(fd, s, sizeof(s));
        printf("%s", s);
        close(fd);
        unlink("Korad Gyorgy");
    } else if(child == 0) {
        int fd = open("Korad Gyorgy", O_WRONLY);
        write(fd, "KGY ZF440N\n", 12);
        close(fd);
    }
    return 0;
}[gyuri@x54c 3-feladat]$
```

4. Gyakorló feladat

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.

```
Terminál - gyuri@x54c:~/OS/ZF440N0sGyak/ZF440N_0411/Források/4-gyakorlati/4
Fájl Szerkesztés Nézet Terminál Lapok Súgó

[gyuri@x54c 4]$ cat msgctl.c
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <stdlib.h>

#define KEY 1
#define PERM 0666

void main()
{
    int id;
    struct msqid_ds buff, *buffp = &buff;
    if( (id = msgget(KEY, PERM | IPC_CREAT)) < 0) {
        perror(" Nem hozható létre az üzenetsor. \n");
        exit(-1);
    }

    msgctl(id, IPC_STAT, buffp);

    if(buff.msg_qnum > 0)
        perror(" Nem üres az üzenetsor. \n");
    else
        msgctl(id, IPC_RMID, NULL);
}
[gyuri@x54c 4]$
```

```
Terminál - gyuri@x54c:~/OS/ZF440N0sGyak/ZF440N_0411/Források/4-gyakorlati/4
Fájl Szerkesztés Nézet Terminál Lapok Súgó

[gyuri@x54c 4]$ cat msgrcv.c
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <stdlib.h>
#include <string.h>

#define KEY 1
#define PERM 0666

void main()
{
    int id;
    struct {long mtype;
            char mtext[80];
    } uzenet;
    int msgsize;

    if( (id = msgget(KEY, PERM | IPC_CREAT)) < 0) {
        perror(" Nem hozható létre az üzenetsor. \n");
        exit(-1);
    }
    puts(" Kérek egy sort >");
    gets(uzenet.mtext);
    uzenet.mtype = 1;
    msgsize = strlen(uzenet.mtext) + 1;

    msgsnd(id, (struct msgbuf *) &uzenet, msgsize, PERM);
}
[gyuri@x54c 4]$
```

```
Terminál - gyuri@x54c:~/OS/ZF440N0sGyak/ZF440N_0411/Forrasok/4-gyakorlati/4
Fájl Szerkesztés Nézet Terminál Lapok Súgó

.rwxr-xr-x 16k gyuri 17 apr 10:09 msgrcv
.rw-r--r-- 526 gyuri 17 apr 10:09 msgrcv.c
[gyuri@x54c 4]$ cat msgc
msgcreate msgcreate.c msgctl msgctl.c
[gyuri@x54c 4]$ cat msgcreate.c
#include <stdlib.h>
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/msg.h>

#define KEY 1
#define PERM 0666

void main()
{
    int id;
    struct msqid_ds buff, *buffp = &buff;
    struct {long mtype;
            char mtext[80];
        } uzenet;
    int msgsize = 32;
    int select = 0;

    if( (id = msgget(KEY, PERM | IPC_CREAT)) < 0) {
        perror(" Nem hozható létre az üzenetsor. \n");
        exit(-1);
    }
    msgctl(id, IPC_STAT, buffp);
    while(buff.msg_qnum) {
        msgrcv(id, (struct msgbuf *) &uzenet, msgsize, select, IPC_NOWAIT);
        puts(uzenet.mtext);
        msgctl(id, IPC_STAT, buffp);
    }
}
[gyuri@x54c 4]$

Terminál - gyuri@x54c:~/OS/ZF440N0sGyak/ZF440N_0411/Forrasok/4-gyakorlati/4
Fájl Szerkesztés Nézet Terminál Lapok Súgó

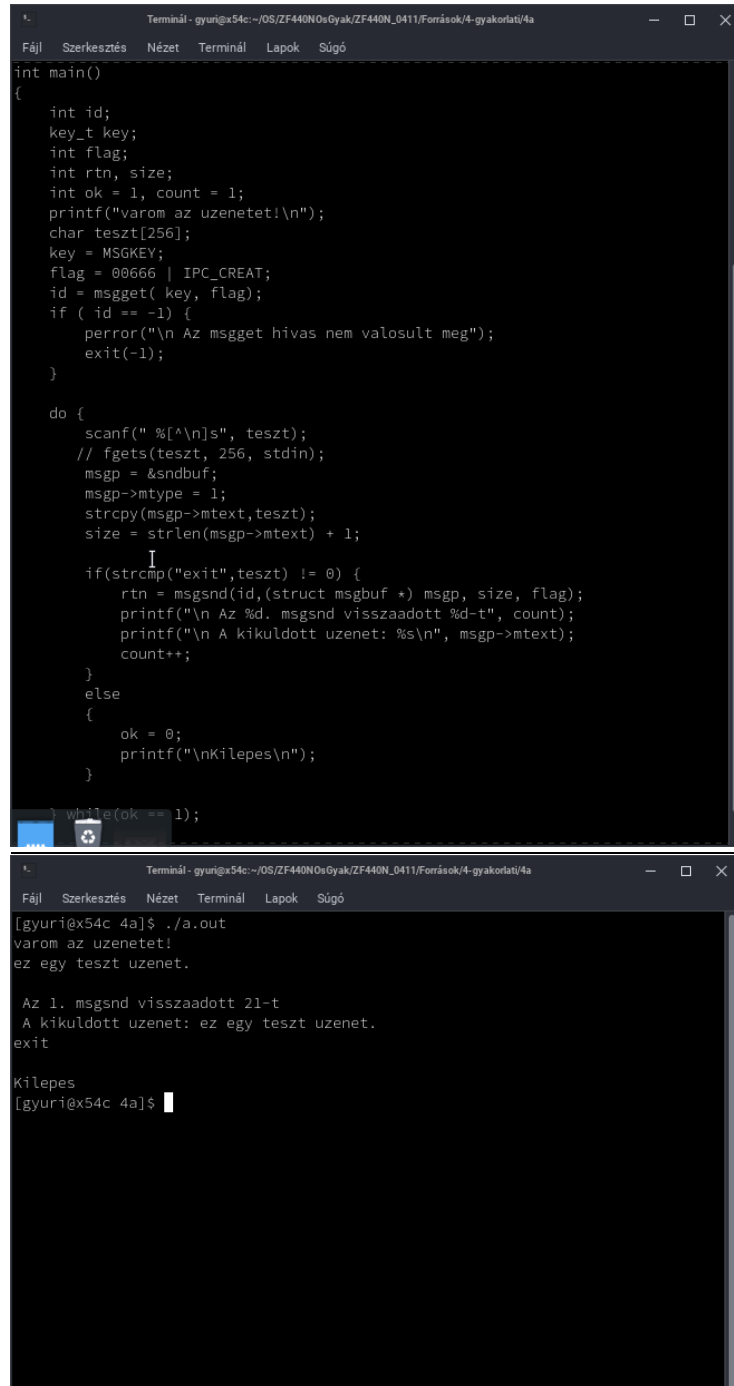
[gyuri@x54c 4]$ ./msgrcv
Kérek egy sort >
Ez egy teszt üzenet.
[gyuri@x54c 4]$ ./msgcreate
Ez egy teszt üzenet.
[gyuri@x54c 4]$ ./msgctl
[gyuri@x54c 4]$ ./msgcreate
[gyuri@x54c 4]$
```

Az msgrcv beolvas egy sort, amit az msgcreate vissza tud olvasni, amíg az msgctl futtatásával ki nem töröljük.

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



The image consists of two terminal window screenshots. The top screenshot shows the source code of a C program named 'gyak10_4.c'. The code defines a message queue, sends a test message, and enters a loop where it can receive messages or exit. The bottom screenshot shows the program's execution. It prompts for a message, receives 'ez egy teszt üzenet.', prints the message details (ID 1, size 21), and then prints 'Kilepes' before returning to the shell prompt.

```
int main()
{
    int id;
    key_t key;
    int flag;
    int rtn, size;
    int ok = 1, count = 1;
    printf("varom az uzenetet!\n");
    char teszt[256];
    key = MSGKEY;
    flag = 00666 | IPC_CREAT;
    id = msgget( key, flag);
    if ( id == -1) {
        perror("\n Az msgget hivas nem valosult meg");
        exit(-1);
    }

    do {
        scanf(" %[^\n]s", teszt);
        // fgets(teszt, 256, stdin);
        msgp = &sndbuf;
        msgp->mtype = 1;
        strcpy(msgp->mtext,teszt);
        size = strlen(msgp->mtext) + 1;

        if(strcmp("exit",teszt) != 0) {
            rtn = msgsnd(id,(struct msgbuf *) msgp, size, flag);
            printf("\n Az %d. msgsnd visszaadott %d-t", count);
            printf("\n A kikuldott uzenet: %s\n", msgp->mtext);
            count++;
        }
        else
        {
            ok = 0;
            printf("\nKilepes\n");
        }
    } while(ok == 1);
}
```

```
[gyuri@x54c 4a]$ ./a.out
varom az uzenetet!
ez egy teszt uzenet.

Az 1. msgsnd visszaadott 21-t
A kikuldott uzenet: ez egy teszt uzenet.
exit

Kilepes
[gyuri@x54c 4a]$
```

A program minden megadott üzenetnek visszaadja a sorszámát és a karakterszámát a '\0' lezáró karaktert beleszámítva, majd, ha exit üzenetet kap leáll a program futása.