

Operációs rendszerek BSc

5. Gyak.

2022. 03. 07.

Készítette:

Kórád György Bsc

Programtervező Informatikus

ZF440N

Miskolc, 2022

1. A `system()` rendszerhívással hajtson végre létező és nem létező parancsot, és vizsgálja a visszatérési értéket, magyarázza egy-egy mondattal A fordítás/futtatás után készítsen egy képernyőképet (minden parancs esetén) és illessze be a dokumentumba.



The first screenshot shows a nano editor window titled 'források : nano — Konsole' editing a file 'ZF440N1fel.c'. The code inside is:

```
GNU nano 6.2                                ZF440N1fel.c                                Módosítva
#include <stdio.h>
#include <stdlib.h>

int main()
{
    system("ls");
    return 0;
}
```

The second screenshot shows a bash terminal window titled 'források : bash — Konsole'. It shows the execution of the compiled program:

```
[gyuri@gyuri-asusX54C források]$ ./a.out
a.out  kod_jo.png  ZF440N1fel.c
[gyuri@gyuri-asusX54C források]$
```

Leírás: A `system()` függvénynek átadtam egy létező UNIX parancsot, futtatva hiba nélkül lefut, a paraméterként kapott parancsot sikeresen átadja az operációs rendszernek, majd az végrehajtja.



The image consists of two terminal window screenshots. The top window is titled 'források: nano — Konsole' and shows the GNU nano 6.2 editor editing a file named 'ZF440N1fel.c'. The code in the file is as follows:

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    system("sl");
    return 0;
}
```

The bottom window is titled 'források: bash — Konsole' and shows the execution of the compiled program. The prompt is '[gyuri@gyuri-asusX54C források]\$' and the command entered is './a.out'. The output is:

```
sh: sor: 1: sl: parancs nem található
[gyuri@gyuri-asusX54C források]$
```

The error message 'sh: sor: 1: sl: parancs nem található' indicates that the 'sl' command is not found, which is the parameter passed to the system() function in the C program.

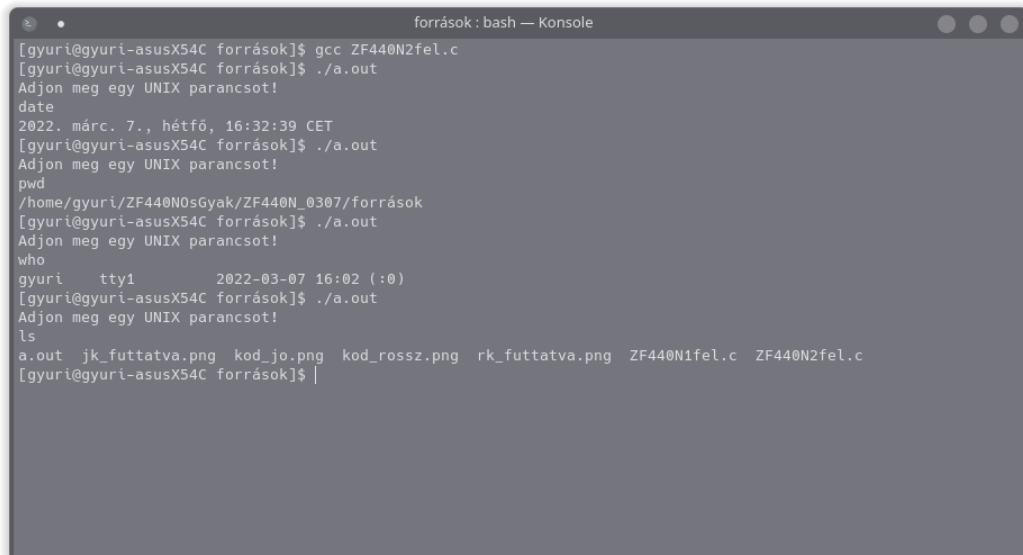
Leírás: Hibásan megadott parancs esetén, a system() függvény ugyanúgy átadja a rendszernek a paraméterét, a program hiba nélkül lefut, viszont mégis hibaüzenetet kapunk vissza a rendszertől.

2. Írjon programot, amely billentyűzetről bekér Unix parancsokat és végrehajtja őket, majd kiírja a szabványos kimenetre.



```
GNU nano 6.2 ZF440N2fel.c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("Adjon meg egy UNIX parancsot! \n");
    char par[6];
    scanf("%s", &par);
    system(par);
    return 0;
}
```



```
[gyuri@gyuri-asusX54C források]$ gcc ZF440N2fel.c
[gyuri@gyuri-asusX54C források]$ ./a.out
Adjon meg egy UNIX parancsot!
date
2022. márc. 7., hétfő, 16:32:39 CET
[gyuri@gyuri-asusX54C források]$ ./a.out
Adjon meg egy UNIX parancsot!
pwd
/home/gyuri/ZF440N0sGyak/ZF440N_0307/források
[gyuri@gyuri-asusX54C források]$ ./a.out
Adjon meg egy UNIX parancsot!
who
gyuri    tty1        2022-03-07 16:02 (:0)
[gyuri@gyuri-asusX54C források]$ ./a.out
Adjon meg egy UNIX parancsot!
ls
a.out  jk_futtatva.png  kod_jo.png  kod_rossz.png  rk_futtatva.png  ZF440N1fel.c  ZF440N2fel.c
[gyuri@gyuri-asusX54C források]$
```

Leírás: A program a scanf függvény segítségével beolvas egy parancsot, majd a system() függvénnyel átadja azt az operációs rendszernek, amennyiben helyes a megadott parancs, az OS lefuttatja azt, vagy ha hibás, egy hibaüzenettel tér vissza.

3. Készítsen egy parent.c és a child.c programokat. A parent.c elindít egy gyermek processzt, ami különbözik a szülőtől. A szülő megvárja a gyermek lefutását. A gyermek szöveget ír a szabványos kimenetre (10-ször) (pl. a hallgató neve és a neptunkód)!

```
források: nano — Konsole
GNU nano 6.2 XY_parent.c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    for(int i = 0; i < 10; i++)
    {
        system("./child.o");
    }
    return 0;
}
```

```
források: nano — Konsole
GNU nano 6.2 XY_child.c Módosítva
#include <stdlib.h>
#include <stdio.h>

int main()
{
    printf("Korad Gyorgy, ZF440N \n");
    return 0;
}
```

```
források: bash — Konsole
[gyuri@gyuri-asusX54C források]$ ./parent.o
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
Korad Gyorgy, ZF440N
[gyuri@gyuri-asusX54C források]$ |
```

Leírás: A parent program egy ciklusba ágyazott system() függvénnyel meghívja a child programot, amely kiírja a terminálra a nevem és neptunkódom.

4. A `fork()` rendszerhívással hozzon létre egy gyerek processzt és abban hívjon meg egy `exec` családbeli rendszerhívást (pl. `execlp`).

```
GNU nano 6.2 ZF440N4fel.c
#include <sys/wait.h>
#include <unistd.h>
#include <sys/types.h>
#include <stdlib.h>
#include <stdio.h>
int main(void)
{
    char command[] = "ls";
    char arg1[] = "-la";

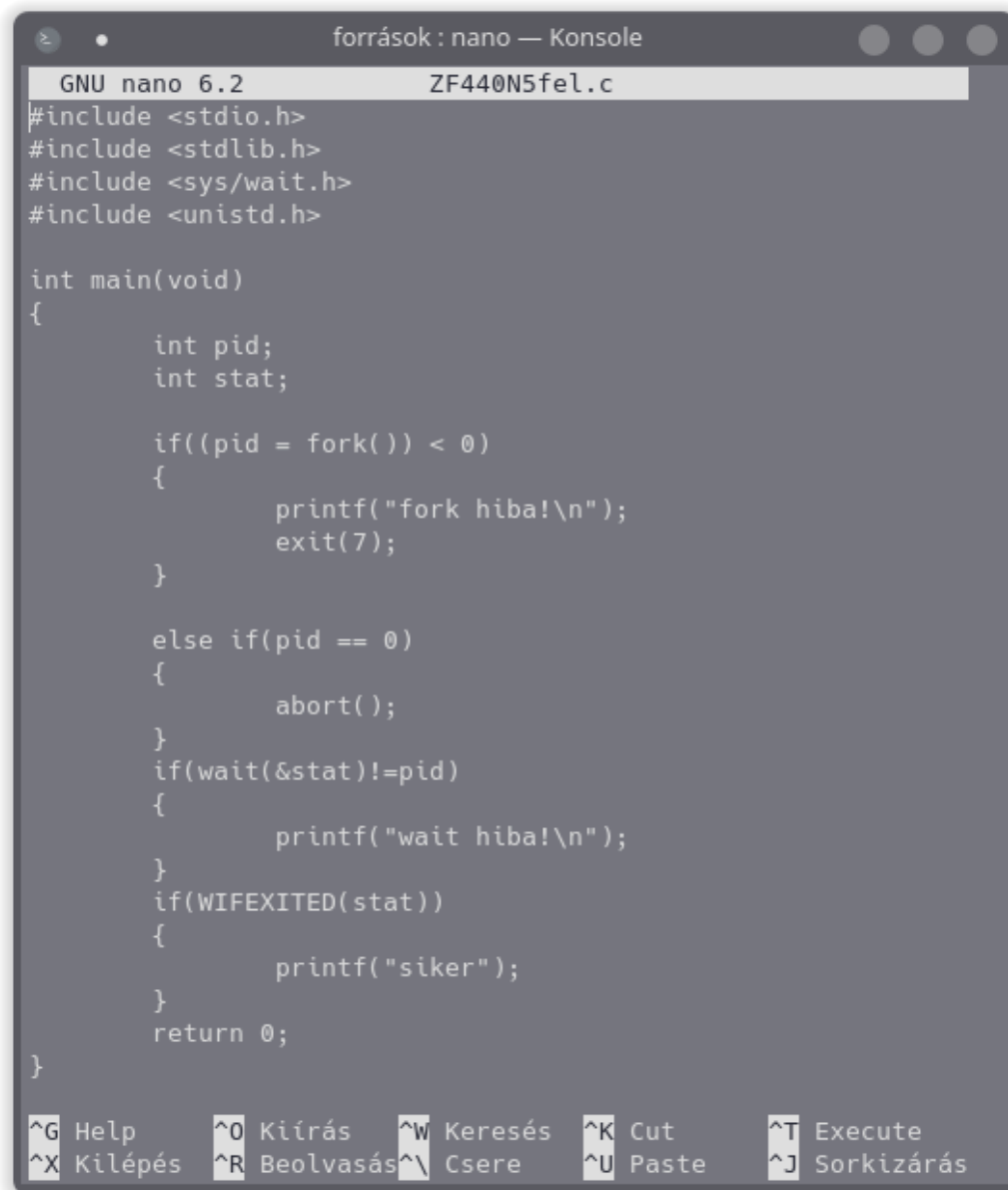
    pid_t pid;

    if ( (pid = fork()) < 0 )
        perror("fork error");
    else if (pid == 0) {
        printf("Ez a gyerek processz. pid: %d\n",pid);
        execlp(command, arg1, 0,0);
    } else {
        printf("Ez a szülő processz. pid: %d\n",pid);
        wait(NULL);
    }
    exit(0);
}
```

```
[gyuri@gyuri-asusX54C forrasok]$ ./a.out
Ez a szülő processz. pid: 15561
Ez a gyerek processz. pid: 0
a.out jk_futtatva.png masodik_kod.png negyedik_feladat-2.png parent-c_futtatva.png YX_child-c.png ZF440N2fel.c ZF440N5fel.c
child.c kod_jo.png masodik_kod.png negyedik_feladat.png parent.o YX_parent-c.png ZF440N3fel.c ZF440N5fel.c
child.o kod_rossz.png negyedik-1.png parent.c rk_futtatva.png ZF440N1fel.c ZF440N4fel.c
[gyuri@gyuri-asusX54C forrasok]$
```

Leírás: A program a `fork()` függvény segítségével létrehoz egy gyerek processzt nullás piddel, majd az `execlp` lefuttatja a paraméterként kapott parancsot. Ezután a program visszatér a szülőhöz, ami kiírja a kijelzőre a pidjét.

5. A `fork()` rendszerhívással hozzon létre gyerekeket, várja meg és vizsgálja a befejeződési állapotokat



```
források : nano — Konsole
GNU nano 6.2      ZF440N5fel.c
#include <stdio.h>
#include <stdlib.h>
#include <sys/wait.h>
#include <unistd.h>

int main(void)
{
    int pid;
    int stat;

    if((pid = fork()) < 0)
    {
        printf("fork hiba!\n");
        exit(7);
    }

    else if(pid == 0)
    {
        abort();
    }
    if(wait(&stat)!=pid)
    {
        printf("wait hiba!\n");
    }
    if(WIFEXITED(stat))
    {
        printf("siker");
    }
    return 0;
}

^G Help      ^O Kiírás    ^W Keresés   ^K Cut       ^T Execute
^X Kilépés   ^R Beolvasás ^\ Csere     ^U Paste     ^J Sorkizárás
```

Leírás: A program a `stat` változóban tárolja a gyerek processz visszatérési értékét, amely ha nem nulla, akkor hibaüzenetet ad vissza a felhasználónak.

6. Adott a következő ütemezési feladat, amit a FCFS, SJF és Round Robin (RR) ütemezési algoritmus használatával készítsen el (külön-külön táblázatba):

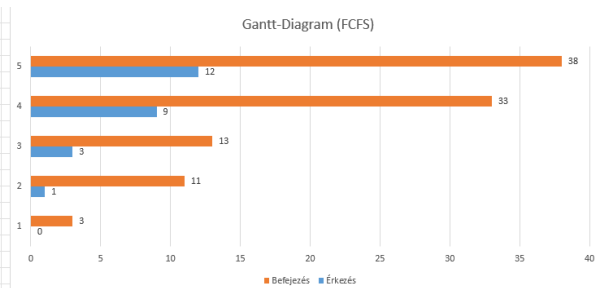
Határozza meg FCFS és SJF esetén

a.) A befejezési időt?

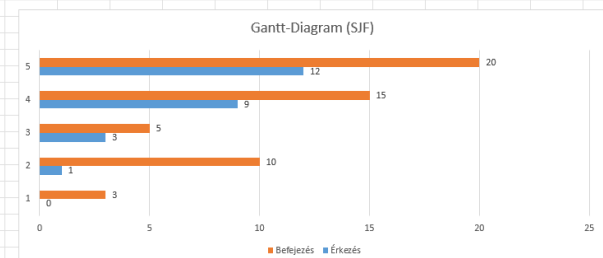
b.) A várakozási/átlagos várakozási időt?

c.) Ábrázolja Gantt diagram segítségével az aktív/várakozó processzek futásának menetét.

| FCFS | Érkezés | CPU idő | Indulás | Befejezés | Várakozás |
|-------------------------|---------|---------|---------|-----------|-----------|
| P1 | 0 | 3 | 0 | 3 | 0 |
| P2 | 1 | 8 | 3 | 11 | 2 |
| P3 | 3 | 2 | 11 | 13 | 8 |
| P4 | 9 | 20 | 13 | 33 | 4 |
| P5 | 12 | 5 | 33 | 38 | 21 |
| Várakozási idők átlaga: | | | | | 7 |

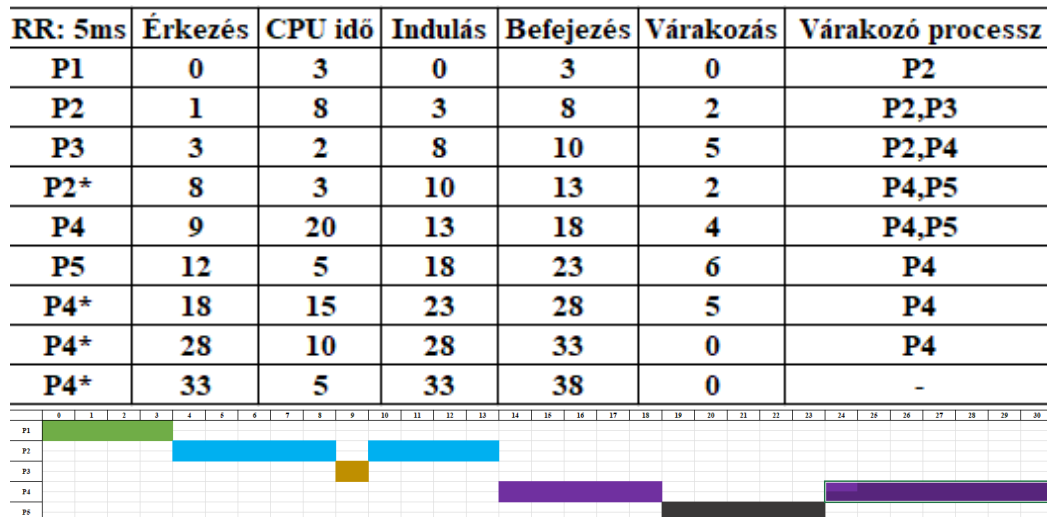


| SJF | Érkezés | CPU idő | Indulás | Befejezés | Várakozás |
|-------------------------|---------|---------|---------|-----------|-----------|
| P1 | 0 | 3 | 0 | 3 | 0 |
| P2 | 1 | 5 | 5 | 10 | 4 |
| P3 | 3 | 2 | 3 | 5 | 0 |
| P4 | 9 | 5 | 10 | 15 | 1 |
| P5 | 12 | 5 | 15 | 20 | 3 |
| Várakozási idők átlaga: | | | | | 1,6 |



II. Round Robin (RR) esetén

- Ütemezze az adott időszakot (5ms) alapján az egyes processzek (befejezési és várakozási/átlagos várakozási idő) paramétereit (ms)!
- A rendszerben lévő processzek végrehajtásának sorrendjét?
- Ábrázolja Gantt diagram segítségével az aktív/várakozó processzek futásának menetét!”



A programok elkészítéséhez a nano szövegszerkesztő programot használtam.
A dokumentumban használt ábrák megtalálhatóak a források mappában.