

Operációs rendszerek BSc

5.gyak

2021. 03. 10.

Készítette:

Rontó Eszter Bsc

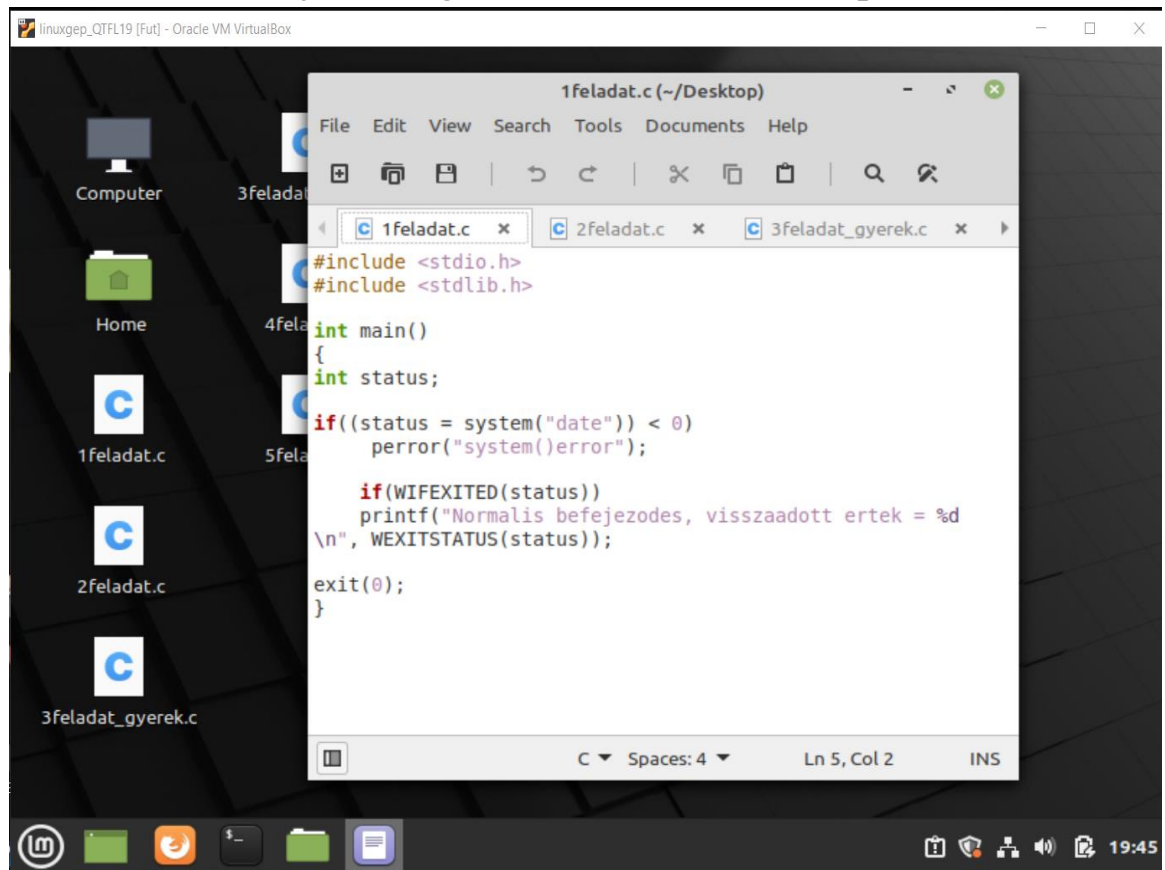
Programtervező inf

QTFL19

Miskolc, 2021

Linux OS - Rendszerhívások, processz ütemezés

1. A `system()` rendszerhívással hajtson végre létező és nem létező parancsot, és vizsgálja a visszatérési értéket!



```
1feladat.c (~/.Desktop)
File Edit View Search Tools Documents Help
1feladat.c 2feladat.c 3feladat_gyerek.c
#include <stdio.h>
#include <stdlib.h>

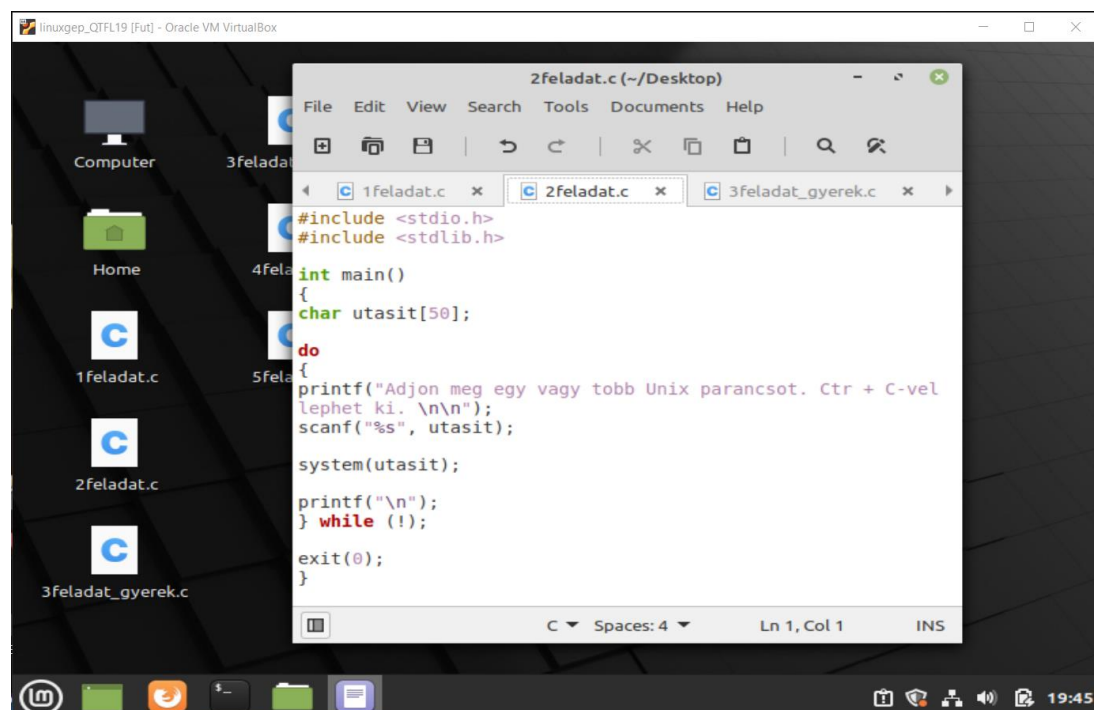
int main()
{
    int status;

    if((status = system("date")) < 0)
        perror("system()error");

    if(WIFEXITED(status))
        printf("Normalis befejezodes, visszaadott ertekek = %d\n", WEXITSTATUS(status));

    exit(0);
}
```

2. Írjon programot, amely billentyűzetről bekér Unix parancsokat és végrehajtja őket, majd kiírja a szabványos kimenetre. (pl.: amit bekér: date, pwd, who etc.; kilépés: CTRL-\\)



```
2feladat.c (~/.Desktop)
File Edit View Search Tools Documents Help
1feladat.c 2feladat.c 3feladat_gyerek.c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    char utasit[50];

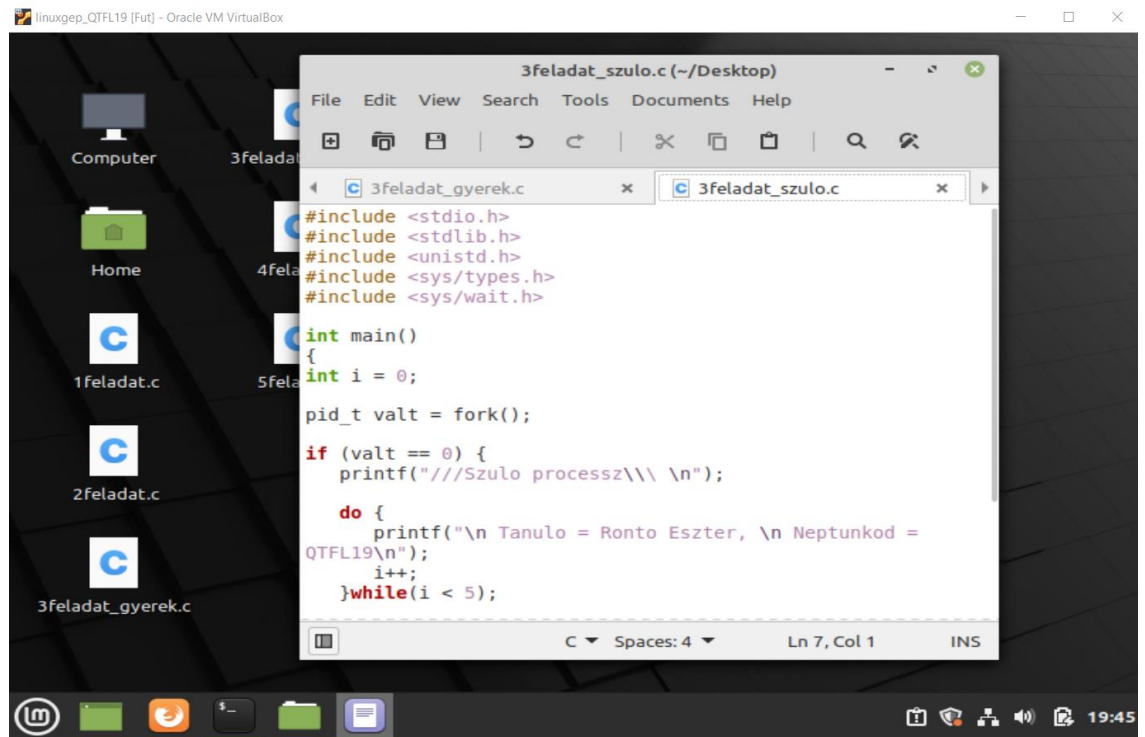
    do
    {
        printf("Adjon meg egy vagy tobb Unix parancsot. Ctr + C-vel lehet ki. \n\n");
        scanf("%s", utasit);

        system(utasit);

        printf("\n");
    } while (!);

    exit(0);
}
```

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 (5-ször) (pl. a hallgató neve és a neptunkód)!



The screenshot shows a Linux desktop environment with a dark theme. In the background, there are several desktop icons labeled 'Computer', 'Home', '1feladat.c', '2feladat.c', and '3feladat_gyerek.c'. A code editor window titled '3feladat_szulo.c (~/.Desktop)' is open in the foreground. The editor shows the following C code:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

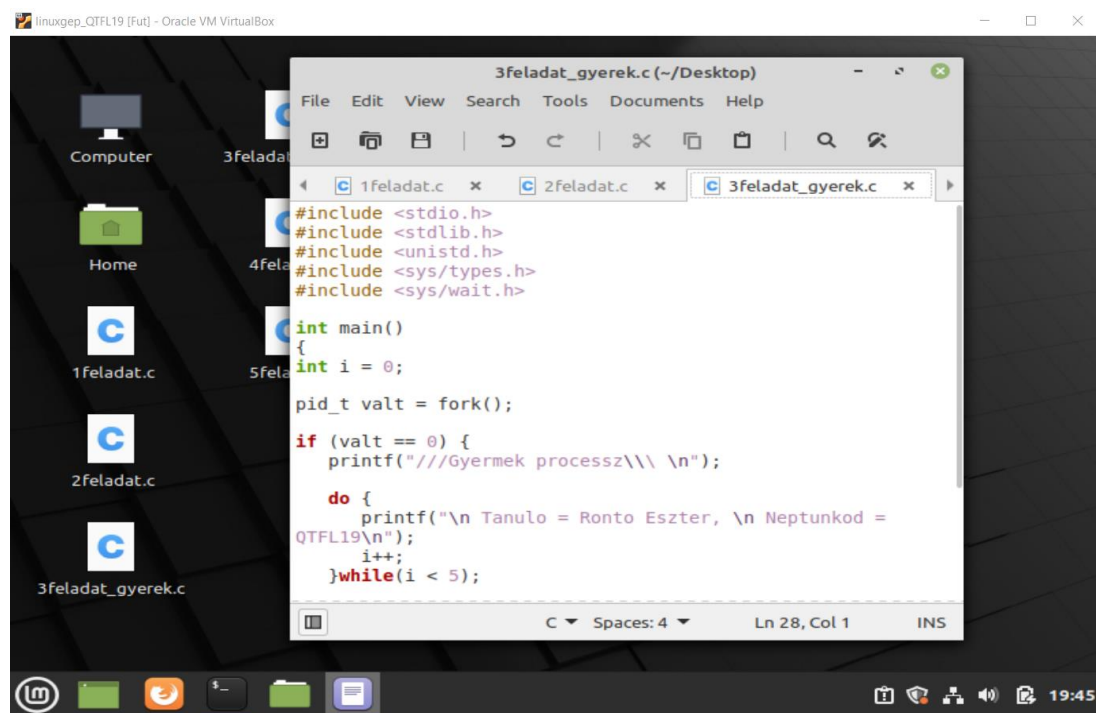
int main()
{
    int i = 0;

    pid_t valt = fork();

    if (valt == 0) {
        printf("///Szulo processz\\\ \n");

        do {
            printf("\n Tanulo = Ronto Eszter, \n Neptunkod = QTFL19\n");
            i++;
        }while(i < 5);
    }
```

The status bar at the bottom of the editor indicates 'C', 'Spaces: 4', 'Ln 7, Col 1', and 'INS'. The system tray at the bottom right shows the time as 19:45.



The screenshot shows the same Linux desktop environment. A code editor window titled '3feladat_gyerek.c (~/.Desktop)' is open in the foreground. The editor shows the following C code:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

int main()
{
    int i = 0;

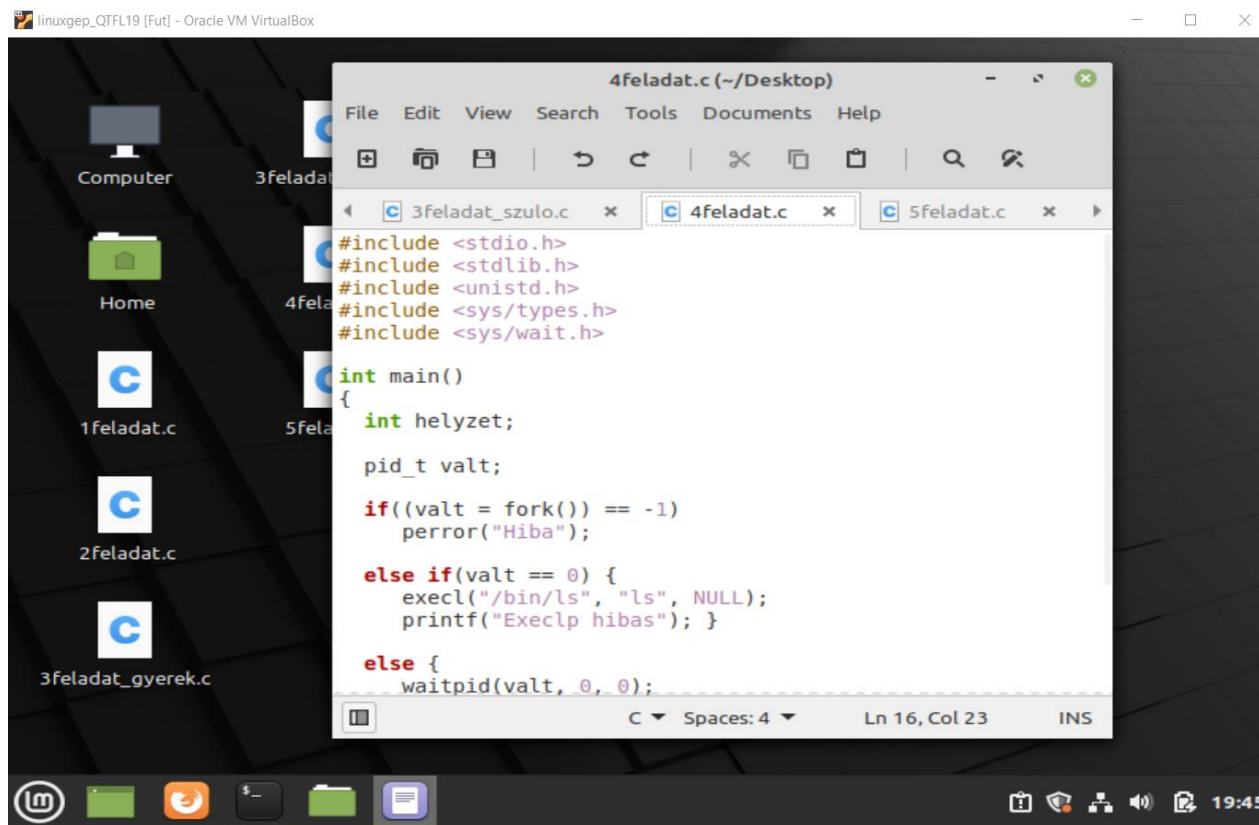
    pid_t valt = fork();

    if (valt == 0) {
        printf("///Gyerek processz\\\ \n");

        do {
            printf("\n Tanulo = Ronto Eszter, \n Neptunkod = QTFL19\n");
            i++;
        }while(i < 5);
    }
```

The status bar at the bottom of the editor indicates 'C', 'Spaces: 4', 'Ln 28, Col 1', and 'INS'. The system tray at the bottom right shows the time as 19:45.

4. A `fork()` rendszerhívással hozzon létre egy gyerek processzt-t és abban hívjon meg egy `exec` családbeli rendszerhívást (pl. `execlp`). A szülő várja meg a gyerek futását!



```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

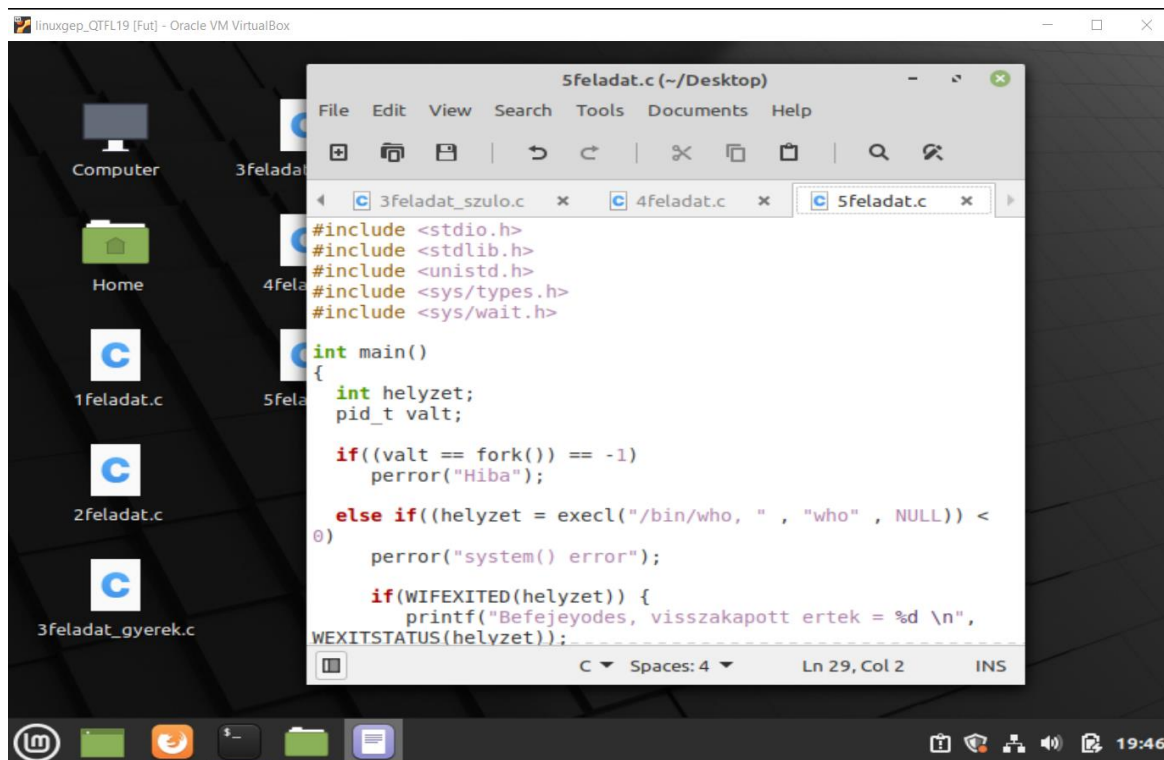
int main()
{
    int helyzet;
    pid_t valt;

    if((valt = fork()) == -1)
        perror("Hiba");

    else if(val == 0) {
        execl("/bin/ls", "ls", NULL);
        printf("Execlp hibas"); }

    else {
        waitpid(val, 0, 0);
    }
}
```

5. A `fork()` rendszerhívással hozzon létre gyerekeket, várja meg és vizsgálja a befejeződési állapotokat (gyerekben: `exit`, `abort`, nullával való osztás)!



```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

int main()
{
    int helyzet;
    pid_t valt;

    if((valt == fork()) == -1)
        perror("Hiba");

    else if((helyzet = execl("/bin/who", " ", "who", NULL)) < 0)
        perror("system() error");

    if(WIFEXITED(helyzet)) {
        printf("Befejezodes, visszkapott ertek = %d \n",
            WEXITSTATUS(helyzet));
    }
}
```

6. Adott a következő ütemezési feladat, ahol a RR ütemezési algoritmus használatával készítse el:

Határozza meg a :

- Ütemezze az adott időszelet alapján az egyes processzek paramétereit (ms)!
- A rendszerben lévő processzek végrehajtásának sorrendjét?
- Ábrázolja Gantt diagram segítségével az aktív/várakozó processzek futásának menetét!

