

Operációs rendszerek BSc

5. Gyak.

2022. 03. 09.

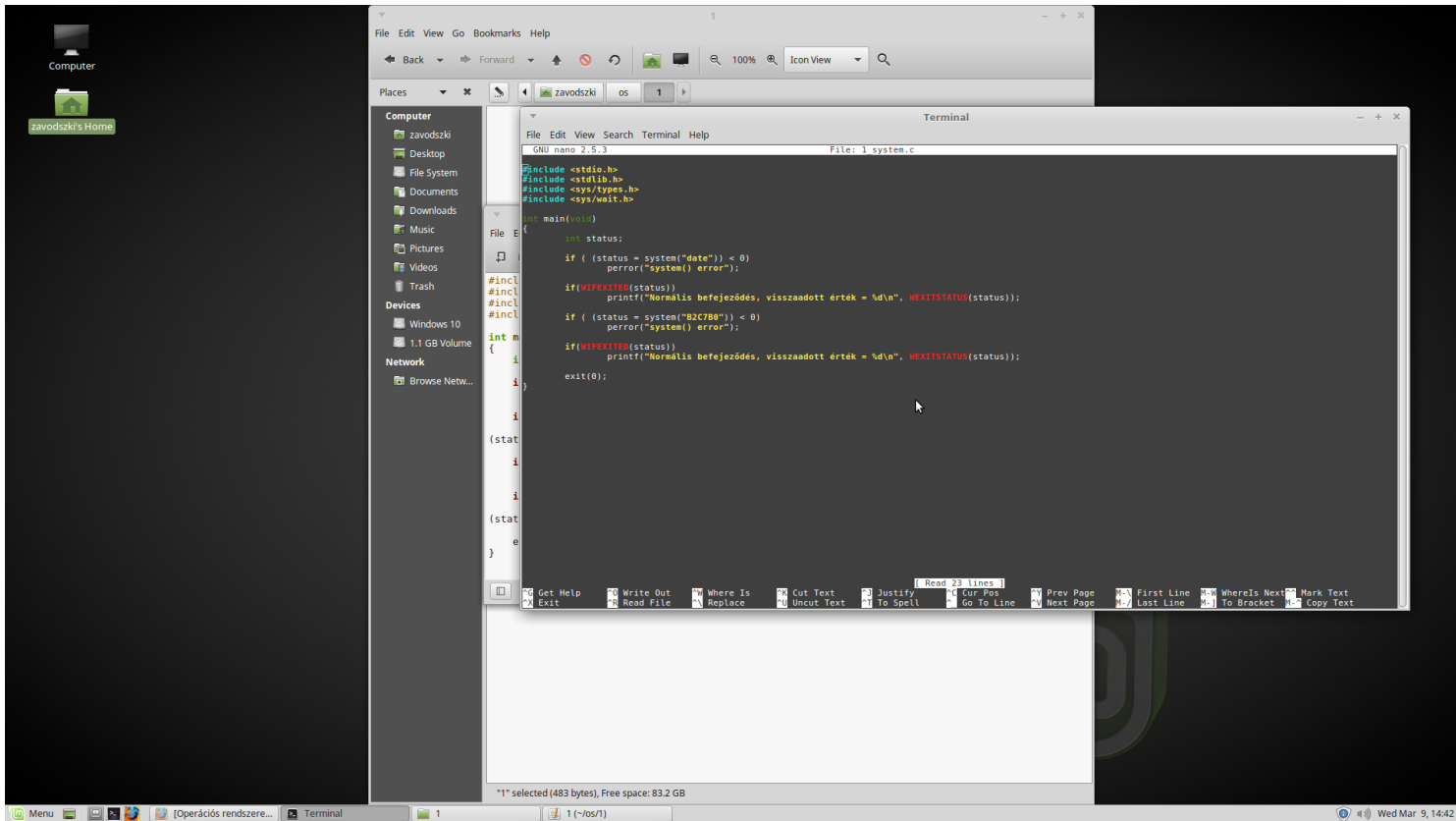
Készítette:

Závodszki Máté
Mérnökinformatikus
B2C7B0

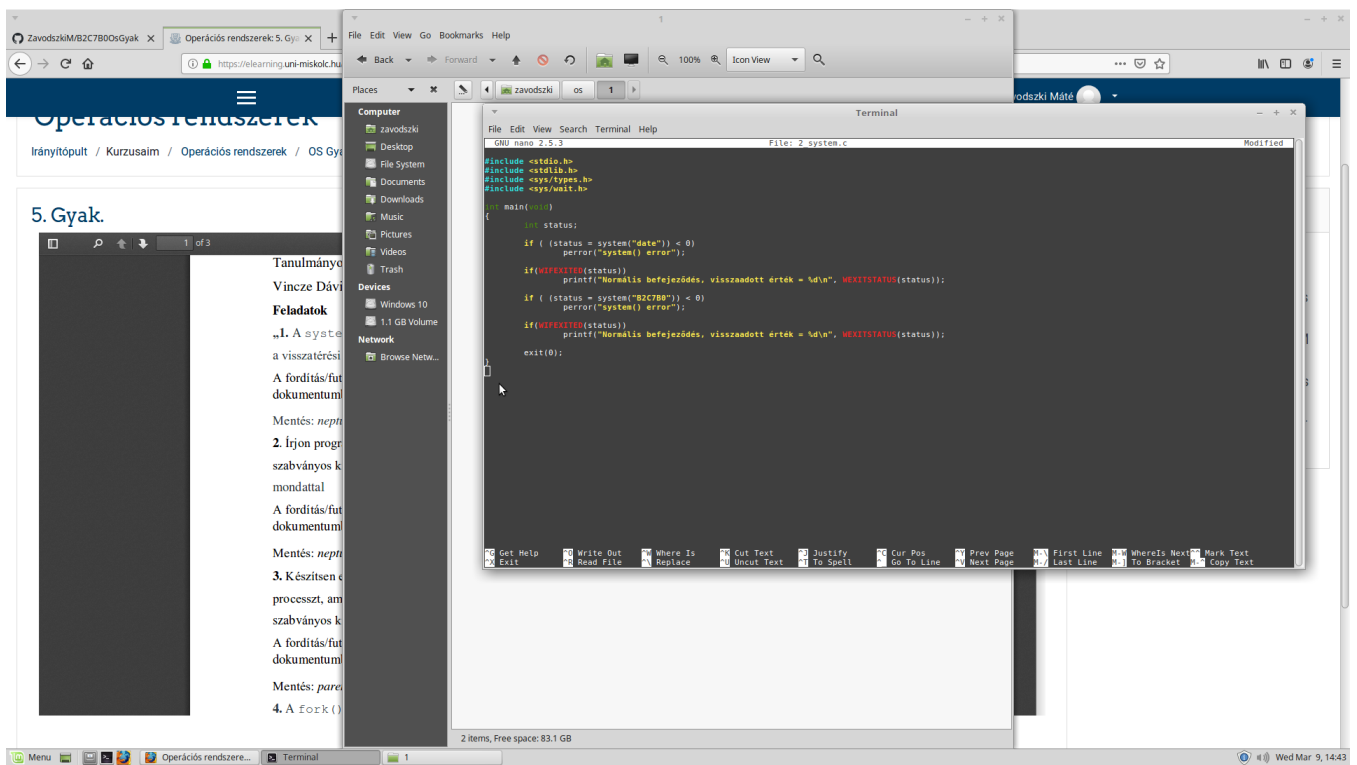
Miskolc, 2022

A feladat

1, A **system()** rendszerhívással hajtson végre létező és nem létező parancsot, és vizsgálja a visszatérési értéket, magyarázza egy-egy mondattal!



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



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)!

The screenshot shows a web browser window with the URL <https://elearning.uni-miskolc.hu/zar/mod/resource/view.php?id=71148>. The page title is "Operációs rendszerek" (Operating Systems). The navigation bar includes "Irányítópult / Kurzusaim / Operációs rendszerek / OS Gyakorlat / 5. Gyak.". The main content area is titled "5. Gyak." and contains the following text:

A fordítás/futtatás után készítsen egy képernyőképet dokumentumba.

Mentés: *neptunkod1felc*

2. Írjon programot, amely billentyűzetről bekér U szabványos kimenetre. (pl.: amit bekér: date, p mondattal

A fordítás/futtatás után készítsen egy képernyőképet dokumentumba.

Mentés: *neptunkod2felc*

3. Készítsen egy *parent.c* és a *child.c* programot, ami különbözik a szülőtől. A szülő me szabványos kimenetre (10-ször) (pl. a hallgató n A fordítás/futtatás után készítsen egy képernyőképet dokumentumba.

Mentés: *parent.c*, ill. *child.c*

4. A `fork()` rendszerhívással hozzon létre egy családbeli rendszerhívást (pl. `exec1p`). A szülő várja meg a gyerek futását! - magyarázza egy-egy mondattal.

A fordítás/futtatás után készítsen egy képernyőképet (minden parancs esetén) és illessze be a dokumentumba.

Mentés: *neptunkod4felc*

Overlaid on the bottom right of the page is a terminal window titled "Terminal". It shows the execution of a C program named `child.c`. The code includes `<stdio.h>`, `<stdlib.h>`, `<sys/types.h>`, and `<sys/wait.h>`. The `main` function contains a loop that prints "Závodszi Máté, B2C780" 10 times. The terminal output shows the program running successfully.

The screenshot shows a web browser window at the top with the address bar displaying "https://elearning.uni-miskolc.hu/zart/mod/resource/view.php?id=71148". The page title is "Operációs rendszerek" and the breadcrumb trail indicates the user is in the "5. Gyak." section. Below the header, there's a sidebar with navigation links like "Irányítópult / Kurzusaim / Operációs rendszerek / OS Gyakorlat / 5. Gyak.". The main content area displays a document titled "5. Gyak." which contains instructions in Hungarian about creating a process tree using fork() and wait(). The document mentions files named neptunkod1fel.c, neptunkod2fel.c, parent.c, and child.c. Overlaid on the bottom right of the browser window is a terminal application. The terminal shows the GNU nano 2.5.3 editor editing a file named parent.c. The code in the terminal includes standard headers, defines pid_t, and implements a main function that forks a child process and waits for it to complete. Comments in the code correspond to the steps described in the document.

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!

The screenshot shows a web browser window with the URL <https://elearning.uni-miskolc.hu/zart/mod/resource/view.php?id=71148>. The page is titled "Operációs rendszerek" and contains a document titled "5. Gyak.". The document text is as follows:

A fordítás/futtatás után készítsen egy képernyőképét a dokumentumba.
Mentés: `neptunkod1fel.c`

2. Írjon programot, amely billentyűzetről bekér U...
szabványos kimenetre. (pl.: amit bekér: date, pwd...
mondattal)
A fordítás/futtatás után készítsen egy képernyőképét a dokumentumba.
Mentés: `neptunkod2fel.c`

3. Készítsen egy `parent.c` és a `child.c` pro...
processzt, ami különbözik a szülőtől. A szülő me...
szabványos kimenetre (10-ször) (pl. a hallgató ne...
A fordítás/futtatás után készítsen egy képernyőképét a dokumentumba.
Mentés: `parent.c`, ill. `child.c`

4. A `fork()` rendszerhívással hozzon létre egy...
családbeli rendszerhívást (pl. `execlp`). A szülő várja meg a gyerek futását! - magyarázza egy-egy...
mondattal.
A fordítás/futtatás után készítsen egy képernyőképét (minden parancs esetén) és illessze be a dokumentumba.
Mentés: `neptunkod4fel.c`

The terminal window shows the following C code:

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

int main(void)
{
    if ((pid = fork()) < 0)
        perror("fork error");
    else if (pid == 0)
        exit(7);
    if (wait(&status) != pid)
        perror("wait hiba");
    if (WIFEXITED(status))
        printf("Normális befejeződés, visszaadott érték = %d\n", WEXITSTATUS(status));

    if ((pid = fork()) < 0)
        perror("fork hiba");
    else if (pid == 0)
        abort();
    if (wait(&status) != pid)
        perror("wait hiba");
    if (WIFEXITED(status))
        printf("Abnormális befejeződés a szignál sorszáma = %d\n", WTERMSIG(status));
}
```

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

The screenshot shows a web browser window with the URL <https://elearning.uni-miskolc.hu/zart/mod/resource/view.php?id=71148>. The page is titled "Operációs rendszerek" and contains a document titled "5. Gyak.". The document text is as follows:

A fordítás/futtatás után készítsen egy képernyőképét a dokumentumba.
Mentés: `neptunkod1fel.c`

2. Írjon programot, amely billentyűzetről bekér U...
szabványos kimenetre. (pl.: amit bekér: date, pwd...
mondattal)
A fordítás/futtatás után készítsen egy képernyőképét a dokumentumba.
Mentés: `neptunkod2fel.c`

3. Készítsen egy `parent.c` és a `child.c` pro...
processzt, ami különbözik a szülőtől. A szülő me...
szabványos kimenetre (10-ször) (pl. a hallgató ne...
A fordítás/futtatás után készítsen egy képernyőképét a dokumentumba.
Mentés: `parent.c`, ill. `child.c`

4. A `fork()` rendszerhívással hozzon létre egy...
családbeli rendszerhívást (pl. `execlp`). A szülő várja meg a gyerek futását! - magyarázza egy-egy...
mondattal.
A fordítás/futtatás után készítsen egy képernyőképét (minden parancs esetén) és illessze be a dokumentumba.
Mentés: `neptunkod4fel.c`

The terminal window shows the following C code:

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

void pr_exit(int status)
{
    if (WIFEXITED(status))
        printf("Normális befejeződés a szignál sorszáma = %d\n", WEXITSTATUS(status));
    else
        printf("Abnormális befejeződés a szignál sorszáma = %d\n", WTERMSIG(status));
}

int main(void)
{
    pid_t pid;
    int status;

    if ((pid = fork()) < 0)
        perror("fork error");
    else if (pid == 0)
        exit(7);
    if (wait(&status) != pid)
        perror("wait hiba");
    pr_exit(status);

    if ((pid = fork()) < 0)
        perror("fork error");
    else if (pid == 0)
        exit(7);
    if (wait(&status) != pid)
        perror("wait hiba");
    pr_exit(status);

    if ((pid = fork()) < 0)
        perror("fork hiba");
    else if (pid == 0)
        abort();
    if (wait(&status) != pid)
        perror("wait hiba");
}
```

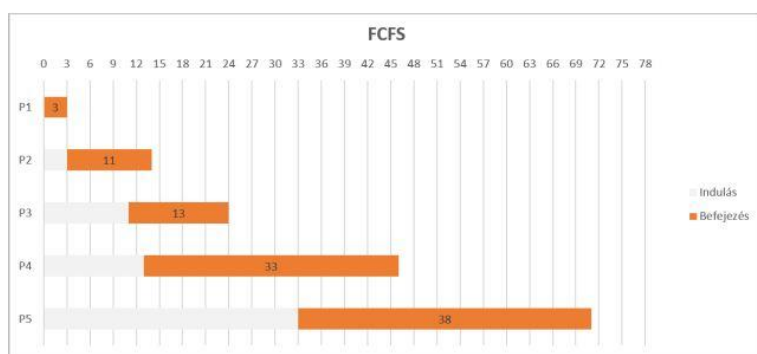
6, Határozza meg FCFS és SJF esetén

a.) A befejezési időt?

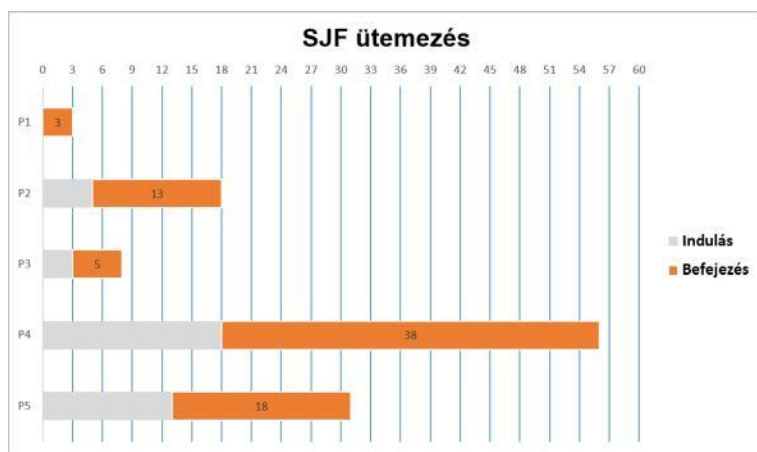
b.) A várakozási/átlagos várakozási időt?

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



SJF	Érkezés	CPU idő	Indulás	Befejezés	Várakozás
P1	0	3	0	3	0
P2	1	8	5	13	4
P3	3	2	3	5	0
P4	9	20	18	38	9
P5	12	5	13	18	1



B feladat – Round Robin

a.) Ütemezze az adott időszelvet (5ms) alapján az egyes processzek (befejezési és várakozási/átlagos várakozási idő) paramétereit (ms)!

b.) A rendszerben lévő processzek végrehajtásának sorrendjét?

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

RR: 5ms	Érkezés	CPU idő	Indulás	Befejezés	Várakozás	Várakozó processz
P1	0	3	0	3	0	P2
P2	1	8	3	8	2	P2, P3
P3	3	2	8	10	5	P2, P4
P4	9	20	13	18	4	P4, P5
P5	12	5	18	23	6	P4

