

Operációs rendszerek BSc

5. Gyak.

2022. 03. 08.

Készítette:

Sziráczki Soma

Bsc

Programtervező

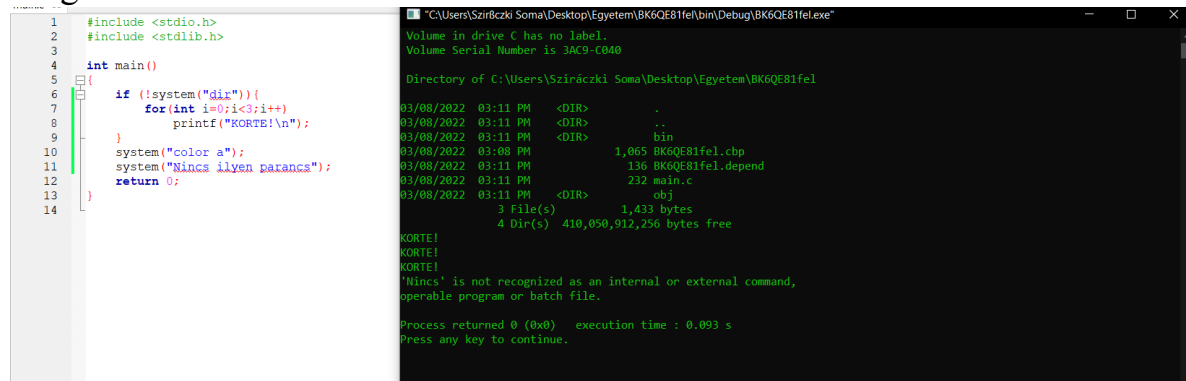
informatikus

BK6QE8

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. Mentés: neptunkod1fel.c

Megvalósítás:



The image shows a C program in a text editor on the left and its execution output in a terminal window on the right. The C program, named BK6QE81fel.c, includes <stdio.h> and <stdlib.h>. It defines a main function that checks if the system command 'dir' is successful. If successful, it prints 'KORTE!' and calls system('color a'). If not successful, it prints 'Nincs ilyen parancs'. The terminal window shows the output of the program, including the directory listing of the current directory and the error message for the 'Nincs' command.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     if (!system("dir")){
7         for(int i=0;i<3;i++){
8             printf("KORTE!\n");
9         }
10        system("color a");
11        system("Nincs ilyen parancs");
12        return 0;
13    }
14 }
```

```
"C:\Users\Sziraczki Soma\Desktop\Egyetem\BK6QE81fel\bin\Debug\BK6QE81fel.exe"
Volume in drive C has no label.
Volume Serial Number is 3AC9-C040

Directory of C:\Users\Sziraczki Soma\Desktop\Egyetem\BK6QE81fel

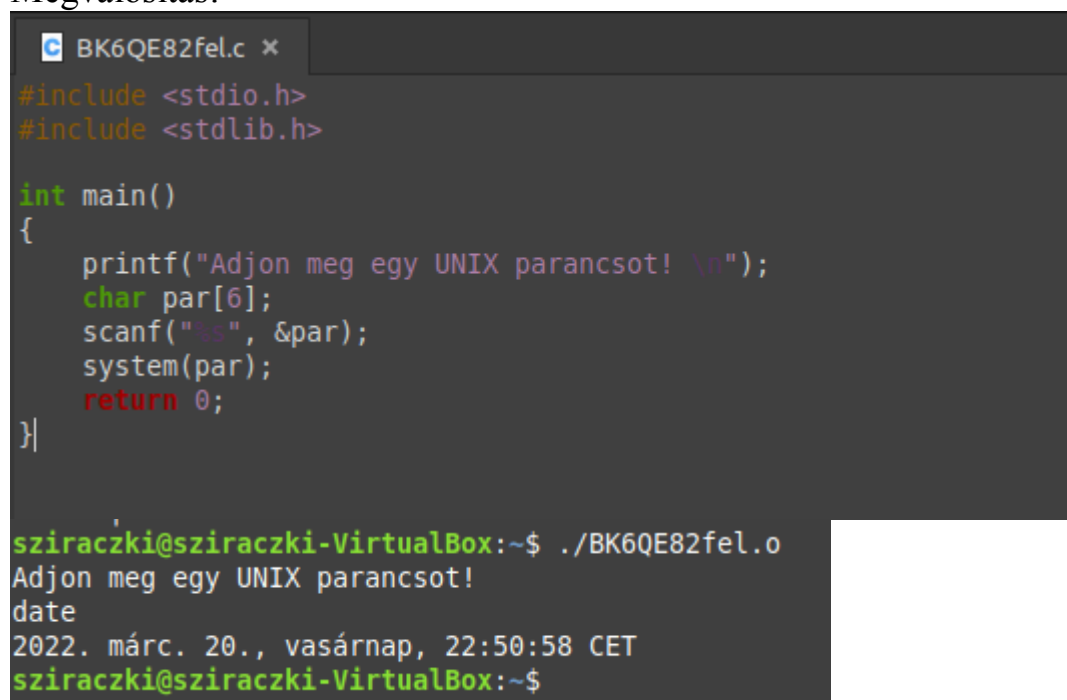
03/08/2022  03:11 PM    <DIR>          .
03/08/2022  03:11 PM    <DIR>          ..
03/08/2022  03:11 PM    <DIR>          bin
03/08/2022  03:08 PM             1,065 BK6QE81fel.chp
03/08/2022  03:11 PM             136 BK6QE81fel.depend
03/08/2022  03:11 PM             232 main.c
03/08/2022  03:11 PM    <DIR>          obj
               3 File(s)            1,433 bytes
               4 Dir(s)  410,050,912,256 bytes free

KORTE!
KORTE!
KORTE!
'Nincs' is not recognized as an internal or external command,
operable program or batch file.

Process returned 0 (0x0)   execution time : 0.093 s
Press any key to continue.
```

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-\) - magyarázza egy-egy mondattal A fordítás/futtatás után készítsen egy képernyőképet (minden parancs esetén) és illessze be a dokumentumba. Mentés: neptunkod2fel.c

Megvalósítás:



The image shows a C program in a text editor on the left and its execution output in a terminal window on the right. The C program, named BK6QE82fel.c, includes <stdio.h> and <stdlib.h>. It defines a main function that prompts the user for a Unix command, reads it, and executes it using system(). The terminal window shows the output of the program, including the prompt 'Adjon meg egy UNIX parancsot!' and the execution of the 'date' command.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     printf("Adjon meg egy UNIX parancsot! \n");
7     char par[6];
8     scanf("%s", &par);
9     system(par);
10    return 0;
11 }
12 }
```

```
sziraczki@sziraczki-VirtualBox:~$ ./BK6QE82fel.o
Adjon meg egy UNIX parancsot!
date
2022. márc. 20., vasárnap, 22:50:58 CET
sziraczki@sziraczki-VirtualBox:~$
```

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)! - magyarázza egy-egy mondattal A fordítás/futtatás után készítsen egy képernyőképet (minden parancs esetén) és illessze be a dokumentumba. Mentés: parent.c, ill. child.c

Megvalósítás:

```
parent.c x
#include <stdlib.h>
#include <stdio.h>

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

child.c x
#include <stdlib.h>
#include <stdio.h>

int main()
{
    printf("Sziraczki Soma, BK6QE8 \n");
    return 0;
}
```

```

sziraczki@sziraczki-VirtualBox:~$ gcc parent.c -o parent.o
sziraczki@sziraczki-VirtualBox:~$ gcc child.c -o child.o
sziraczki@sziraczki-VirtualBox:~$ ./parent.o
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
Sziraczki Soma, BK6QE8
sziraczki@sziraczki-VirtualBox:~$

```

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! - magyarázza egy-egy mondattal. A fordítás/futtatás után készítsen egy képernyőképet (minden parancs esetén) és illessze be a dokumentumba. Mentés: neptunkod4fel.c

Megvalósítás:

```

BK6QE84fel.c x
#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 szulo processz. pid: %d\n",pid);
        wait(NULL);
    }
    exit(0);
}

sziraczki@sziraczki-VirtualBox:~$ ./BK6QE84fel.o
Ez a szulo processz. pid: 7507
Ez a gyerek processz. pid: 0
Asztal BK6QE84fel.c BK6QE84fel.o child.c child.o Dokumentumok Képek Letöltések nano.save Nyilvános parent.c parent.o Sablonok Videók Zenék
sziraczki@sziraczki-VirtualBox:~$

```

5. A fork() rendszerhívással hozzon létre gyerekeket, várja meg

és vizsgálja a befejezési állapotokat (gyereken: exit, abort, nullával való osztás)! - magyarázza egy-egy mondattal! A fordítás/futtatás után készítsen egy képernyőképet (minden parancs esetén) és illessze be a dokumentumba. Mentés: neptunkod5fel.c

Megvalósítás:

```
BK6QE85fel.c x
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

int main()
{
    pid_t p = fork();
    pid_t p2 = fork();
    if ( p == -1 )
    {
        perror("Fork failed");
        return EXIT_FAILURE;
    }
    else if ( p == 0 )
    {
        execl("/bin/sh", "bin/sh", "-c", "./error", "NULL");
        return EXIT_FAILURE;
    }

    if ( p2 == -1 )
    {
        perror("Fork failed");
        return EXIT_FAILURE;
    }
    else if ( p2 == 0 )
    {
        execl("/bin/sh", "bin/sh", "-c", "./dividezero", "NULL");
        return EXIT_FAILURE;
    }

    int status;
    if ( waitpid(p, &status, 0) == -1 )
    {
        perror("Waitpid failed");
        return EXIT_FAILURE;
    }

    if ( WIFEXITED(status) )
    {
        const int es = WEXITSTATUS(status);
        printf("Exit status was %d", es);
    }

    if ( waitpid(p2, &status, 0) == -1 )
    {
        perror("Waitpid failed");
        return EXIT_FAILURE;
    }
}
```

```
sziraczki@sziraczki-VirtualBox:~$ ./BK6QE85fel.o
NULL: 1: ./error: not found
NULL: 1: ./dividezero: not found
NULL: 1: ./error: not found
Exit status was 127
Exit status was 127
sziraczki@sziraczki-VirtualBox:~$
```

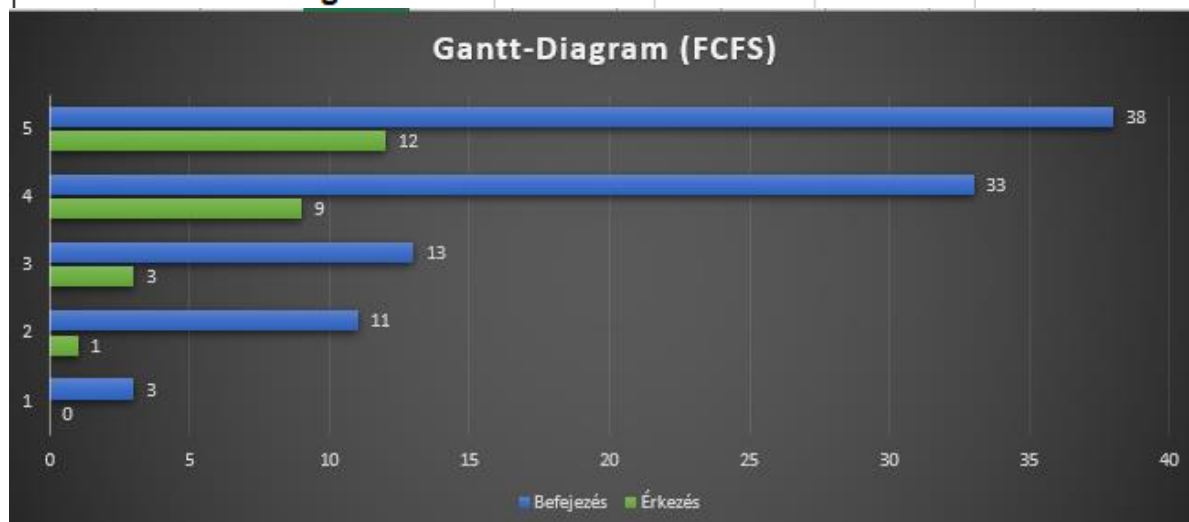
6. Adott a következő ütemezési feladat, amit a FCFS, SJF és Round Robin (RR) ütemezési algoritmus használatával készítsen el (külön-külön táblázatba): I. Határozza meg FCFS és SJF esetén

a.) A befejezési időt?

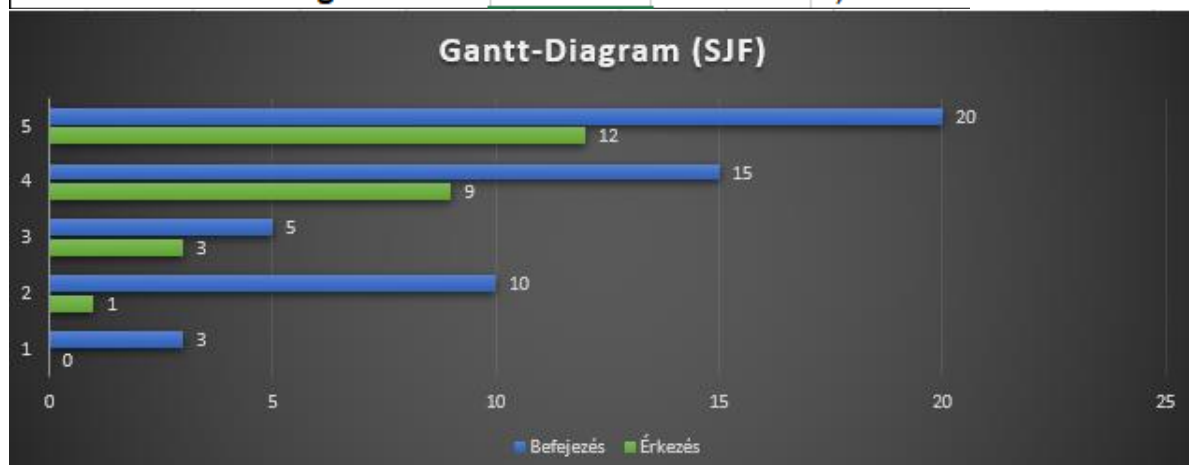
b.) A várakozási/átlagos várakozási időt?

c.) Ábrázolja Gantt diagram segítségével az aktív/várakozó processzek futásának menetét. Megj.: a Gantt diagram ábrázolása szerkesztő program segítségével vagy Excel programmal. Mentés: neptunkod6fel.pdf

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

a.) Ütemezze az adott időszel (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!” Megj.: a Gantt diagram ábrázolása szerkesztő program segítségével vagy Excel programmal. Mentés: neptunkod6fel pdf

Megvalósítás:

