

Operációs rendszerek BSc

5.Gyak

2022.03.07.

Készítette:

Urbán Olivér BSc

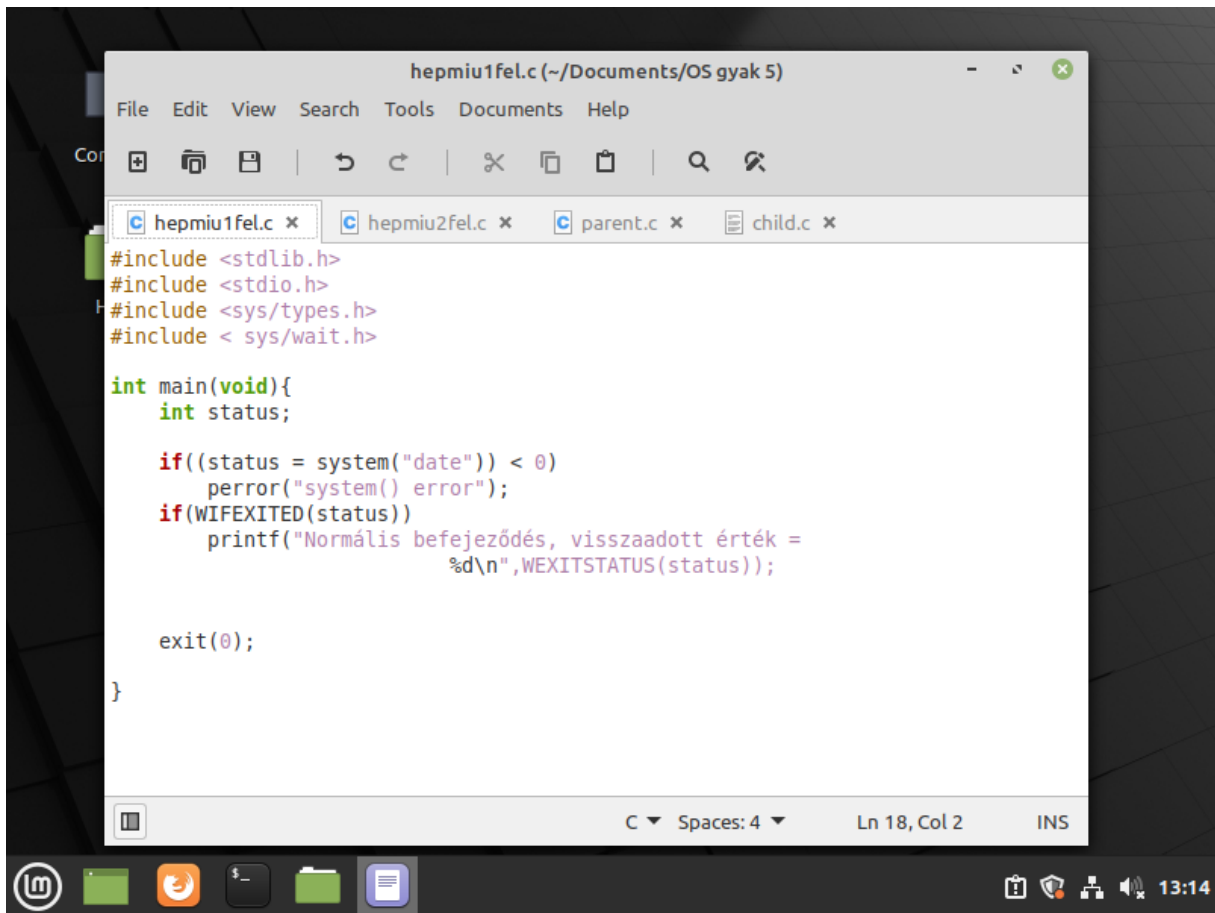
Szak:

Mérnökinformatikus

Neptunkód: HEPMIU

2022.03.07.

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



```
hepmiu1fel.c (~/.Documents/OS gyak 5)
File Edit View Search Tools Documents Help
hepmiu1fel.c x hepmiu2fel.c x parent.c x child.c x
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>

int main(void){
    int status;

    if((status = system("date")) < 0)
        perror("system() error");
    if(WIFEXITED(status))
        printf("Normális befejeződés, visszaadott érték =
               %d\n", WEXITSTATUS(status));

    exit(0);
}
```

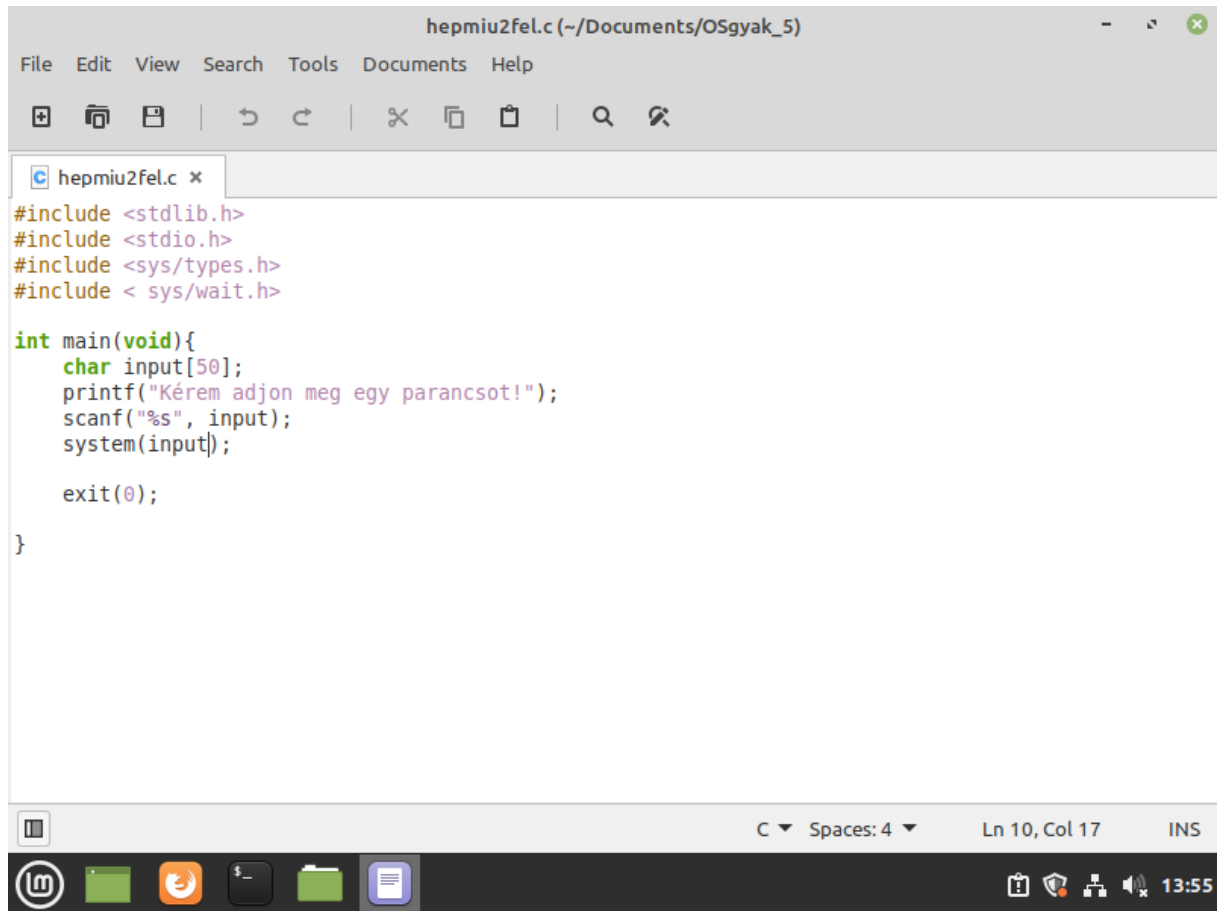
C Spaces: 4 Ln 18, Col 2 INS

13:14

Ha a parancs létezik , végrehajtódik , ha nem akkor egy értékkel tér vissza.

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



```
hepmiu2fel.c (~/.Documents/OSgyak_5)
File Edit View Search Tools Documents Help
+ - | ↶ ↷ | ✂ | 🔍
hepmiu2fel.c x
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>

int main(void){
    char input[50];
    printf("Kérem adjon meg egy parancsot!");
    scanf("%s", input);
    system(input);

    exit(0);
}
```

C Spaces: 4 Ln 10, Col 17 INS 13:55

A parancsot a scanf –el kérjük be majd végrehajtódik.

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

```
parent.c (~/Documents/OSgyak_5)
File Edit View Search Tools Documents Help
+ - | < > | ✂ | | 🔍
parent.c x child.c x hepmiu1fel.c x

#include <unistd.h>
#include <sys/types.h>
#include < sys/wait.h>
pid_t wait(int *wstatus);
pid_t waitpid(pid_t pid, int *wstatus, int options);

int main(){
    pid_t pid;

    if((pid = fork()) < 0 ){
        perror("fork error");
    }
    else if(pid == 0){
        if(execl("./child", "child", (char *)NULL) < 0)
            perror("execl error");
    }
    if(waitpid(pid, NULL , 0) < 0 )
    {   perror("wait error");
    }

    exit(0);
}
```

Ln 15, Col 40 INS

```
child.c (~/Documents/OSgyak_5)
File Edit View Search Tools Documents Help
+ - | < > | ✂ | | 🔍
parent.c x child.c x hepmiu1fel.c x

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

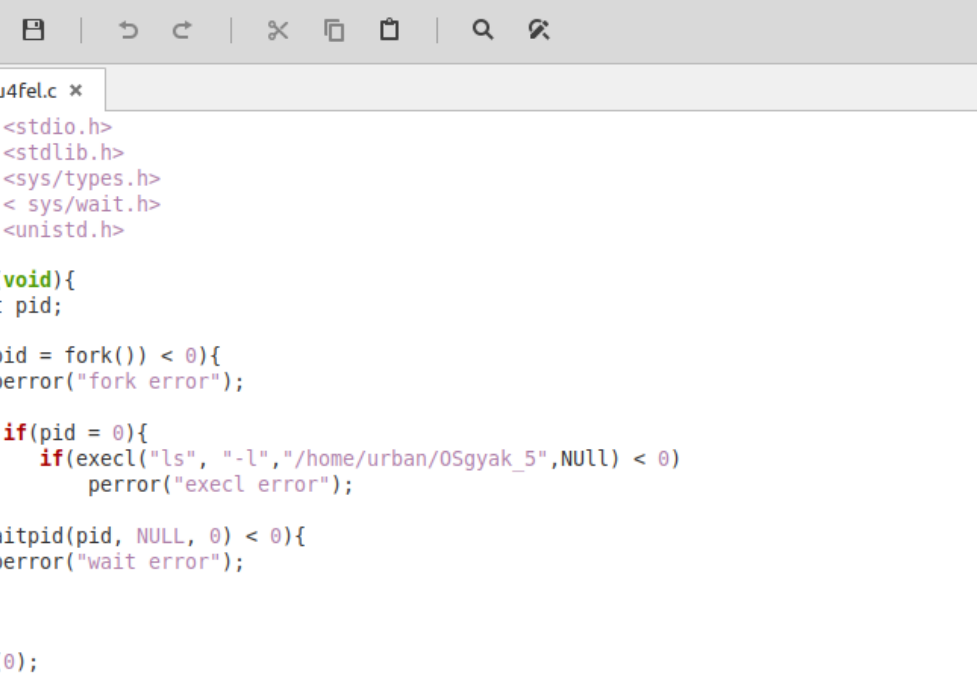
int main(void){

    for(int i = 0; i<10 ;i++){
        printf("Urbán Olivér HEPMIU\n");
        sleep(2);
    }

    exit(0);
}
```

Ln 11, Col 16 INS

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.



The screenshot shows a code editor window titled 'hepmiu4fel.c (~/Documents/OSgyak_5)'. The menu bar includes File, Edit, View, Search, Tools, Documents, and Help. The toolbar contains icons for file operations and search. The code is as follows:

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

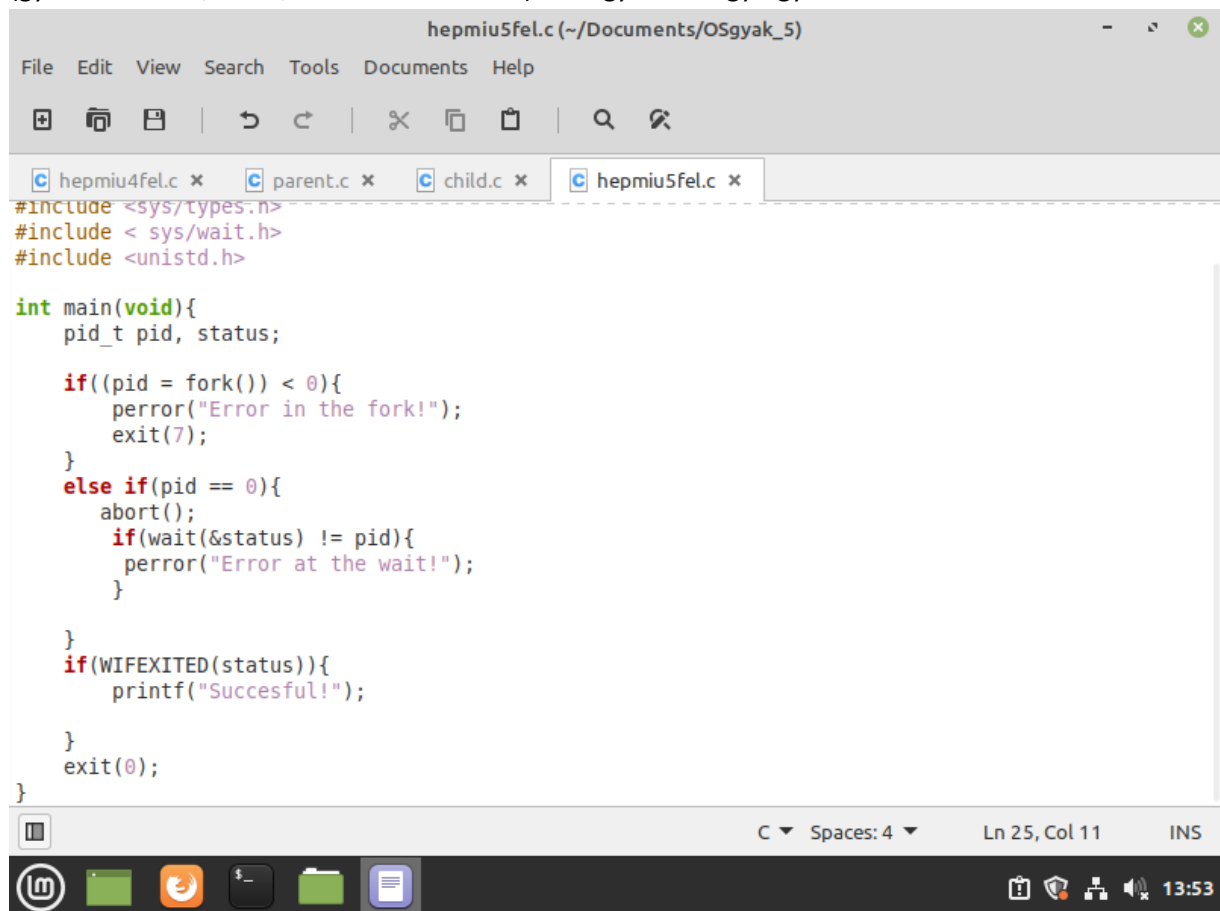
int main(void){
    pid_t pid;

    if((pid = fork()) < 0){
        perror("fork error");
    }
    else if(pid = 0){
        if(execl("ls", "-l", "/home/urban/OSgyak_5", NULL) < 0)
            perror("execl error");
    }
    if(waitpid(pid, NULL, 0) < 0){
        perror("wait error");
    }

    exit(0);
}
```

The status bar at the bottom indicates 'Loading file "/home/urban/Documents/OSgyak_5/hepmiu4fel.c"...', 'C' language, 'Spaces: 4', 'Ln 5, Col 20', and 'INS' mode. The system tray at the bottom right shows the time as 13:56.

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



```
hepmiu5fel.c (~/.Documents/OSgyak_5)
File Edit View Search Tools Documents Help
+ [ ] [ ] [ ] | [ ] [ ] [ ] | [ ] [ ] [ ] | [ ] [ ] [ ]
hepmiu4fel.c x parent.c x child.c x hepmiu5fel.c x
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>

int main(void){
    pid_t pid, status;

    if((pid = fork()) < 0){
        perror("Error in the fork!");
        exit(7);
    }
    else if(pid == 0){
        abort();
        if(wait(&status) != pid){
            perror("Error at the wait!");
        }
    }
    if(WIFEXITED(status)){
        printf("Successful!");
    }
    exit(0);
}
```

C Spaces: 4 Ln 25, Col 11 INS

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.

