Operációs rendszerek BSc

5. Gyak. 2022. 03. 09.

Készítette:

Holczer Vanda Bsc Programtervező informatikus COLLER **1. feladat** – Létező és nem létező parancs végrehajtása system() rendszerhívással

```
Terminal
                                                                                    - + \times
File Edit View Search Terminal Help
holczer@jerry:~/Desktop/OS C$ nano
holczer@jerry:~/Desktop/OS_C$ pwd
/home/stud2021/holczer/Desktop/OS C
holczer@jerry:~/Desktop/OS_C$ cc COLLER1fel.c
holczer@jerry:~/Desktop/OS_C$ cc COLLER1fel
/usr/bin/ld: cannot find COLLER1fel: File format not recognized collect2: error: ld returned 1 exit status
holczer@jerry:~/Desktop/OS C$ cc C0LLER1fel.c
holczer@jerry:~/Desktop/OS_C$ ls
a.out COLLER1fel COLLER1fel.c
holczer@jerry:~/Desktop/OS_C$ ./a.out
Wed Mar 9 13:03:47 CET 2022
Normalis befejezodes, visszaadott ertek = 0
holczer@jerry:~/Desktop/OS_C$ pico C0LLER1fel.c
holczer@jerry:~/Desktop/OS_C$ cc C0LLER1fel.c
holczer@jerry:~/Desktop/OS_C$ ls
a.out COLLER1fel COLLER1fel.c
holczer@jerry:~/Desktop/OS_C$ ./a.out
Wed Mar 9 13:08:01 CET 2022
Normalis befejezodes, visszaadott ertek = 0
sh: 1: COLLER: not found
Normalis befejezodes, visszaad<u>o</u>tt ertek = 127
holczer@jerry:~/Desktop/OS_C$
```

2. feladat – Parancs végrehajtása a bemenetről bekérve

```
#include <stdio.h>
 2
       #include <stdlib.h>
 3
       #include <stdbool.h>
 5
       int main()
     □ {
 6
 7
            char *command="";
 8
           char quit;
 9
10
           do
     \Box
11
                printf("Give a command: ");
12
                scanf ("%s", command);
13
14
                system(command);
15
                puts (command);
                scanf("%c", &quit);
16
17
                printf("%c",quit);
18
            }while(!(quit=='n'));
19
20
            return 0;
       }
21
"D:\Egyetem\C progarmozßs\C0LLER2fel.c\bin\Debug\C0LLER2fel.exe"
Give a command: pwd
Process returned -1073741819 (0xC0000005)
                                               execution time : 7.400 s
Press any key to continue.
```

3. feladat – Gyermek és szülő processz létrehozása

```
#include <stdio.h>
 2
       #include <stdlib.h>
 3
       #include <stdbool.h>
 4
       #include <unistd.h>
 5
       #include <sys/types.h>
       #include <sys/wait.h>
 6
 7
 8
       int main()
 9
           pid t pid = fork();
10
11
12
           if (pid == 0)
13
               execl("c", "Command-Line", NULL);
14
15
16
           waitpid(pid, NULL, 0);
17
18
           printf("The child-process has terminated!");
19
20
           return 0;
21
22
```

4. feladat – fork()-al gyerek processz létrehozása, exec családbeli rendszerhívás

```
1
       #include <stdio.h>
 2
       #include <stdlib.h>
 3
       #include <stdbool.h>
 4
       #include <unistd.h>
 5
       #include <sys/types.h>
 6
       #include <sys/wait.h>
 8
       int main()
 9
     ⊟{
           char *argv[3] = {"Command-line", ".", NULL};
10
11
12
           int pid = fork();
13
14
           if ( pid == 0 ) {
15
               execlp( "find", argv );
16
17
           wait(2);
18
           printf("Child process terminated\n");
19
20
21
           return 0;
22
23
```

5. feladat – fork()-al gyerek processz létrehozása, befejeződési állapotok

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
pid_t fork(void);
#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(void)
   pid_t pid;
    if((pid=fork())<0) perror("fork hiba");
        else if(pid==0)
                exit(7);
    if(wait(&status)!=pid) perror("wait hiba");
    if(WIFEXITED(status))
            printf("Normalis befejezodes, visszaadott ertek= %d\n", WEXITSTATUS(status));
     if((pid=fork())<0) perror("fork hiba");
        else if(pid==0)
                abort();
    if((wait()
}
```