Operációs rendszerek BSc

5.gyak.

2021. 03. 14.

Készítette:
Szabó Alen Bsc
Progterv. info.
MX6WLR

Miskolc, 2021

1. feladat system rendszerhívás

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

int main()
{
    printf("Elso feladat tesztelese!\n");
    printf("Most billentyu lenyomasig itt maradunk\n");
    system("pause");
    printf("Megtortent a billentyu lenyomas\n");
    system("random");
    printf("Remelem nem tortent semmi");
    system("pause");
    system("cls");
    printf("Letisztitottam a console-t");
    return 0;
}
```

"C:\Users\alens\OneDrive\Desktop\Suli\2.fÚIÚv\OS\OS gyak\0310\pro...

```
Elso feladat tesztelese!
Most billentyu lenyomasig itt maradunk
Press any key to continue . . .
Megtortent a billentyu lenyomas
'random' is not recognized as an internal or external command,
operable program or batch file.
Remelem nem tortent semmiPress any key to continue . . .
```

2. feladat unix parancsok

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

int main(void){

char bekert[100];
    printf("Enter the command:");
    scanf("%s", read);

        system(read);

return 0;
}
```

"C:\Users\alens\OneDrive\Desktop\Suli\2.fÚIÚv\OS\OS gy...

```
Enter the command:date
The current date is: 2021. 03. 14.
Enter the new date: (yy-mm-dd)
```

3. feladat

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

int main (void) {
        pid_t pid;

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

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main()
{
    for(int i = 0; i < 5; i++){
        printf("SzA MX6WLR\n");
    }
    return 0;
}
```

4. feladat fork ()

ain.c

```c
#include  <stdio.h>
#include  <sys/types.h>

#define   MAX_COUNT  200

void  ChildProcess(void);                /* child process prototype  */
void  ParentProcess(void);               /* parent process prototype */

void  main(void)
{
     pid_t  pid;

     pid = fork();
     if (pid == 0)
          ChildProcess();
     else
          ParentProcess();
}

void  ChildProcess(void)
{
     int   i;

     for (i = 1; i <= MAX_COUNT; i++)
          printf("   This line is from child, value = %d\n", i);
     printf("   *** Child process is done ***\n");
}

void  ParentProcess(void)
{
     int   i;

     for (i = 1; i <= MAX_COUNT; i++)
          printf("This line is from parent, value = %d\n", i);
     printf("*** Parent is done ***\n");
}
```

5. feladat gyerekek létrehozása fork()

```
int main()
{
    int pid, pid1, pid2;

    pid = fork();

    if (pid == 0) {

        sleep(3);

        printf("child[1] --> pid = %d and ppid = %d\n",
            getpid(), getppid());
    }

    else {
        pid1 = fork();
        if (pid1 == 0) {
            sleep(2);
            printf("child[2] --> pid = %d and ppid = %d\n",
                getpid(), getppid());
        }
        else {
            pid2 = fork();
            if (pid2 == 0) {
                printf("child[3] --> pid = %d and ppid = %d\n",
                    getpid(), getppid());
            }

            else {
                sleep(3);
                printf("parent --> pid = %d\n", getpid());
            }
        }
    }

    return 0;
}
```