

Operációs rendszerek BSc

5.gyak.

2021. 03. 14.

Készítette:

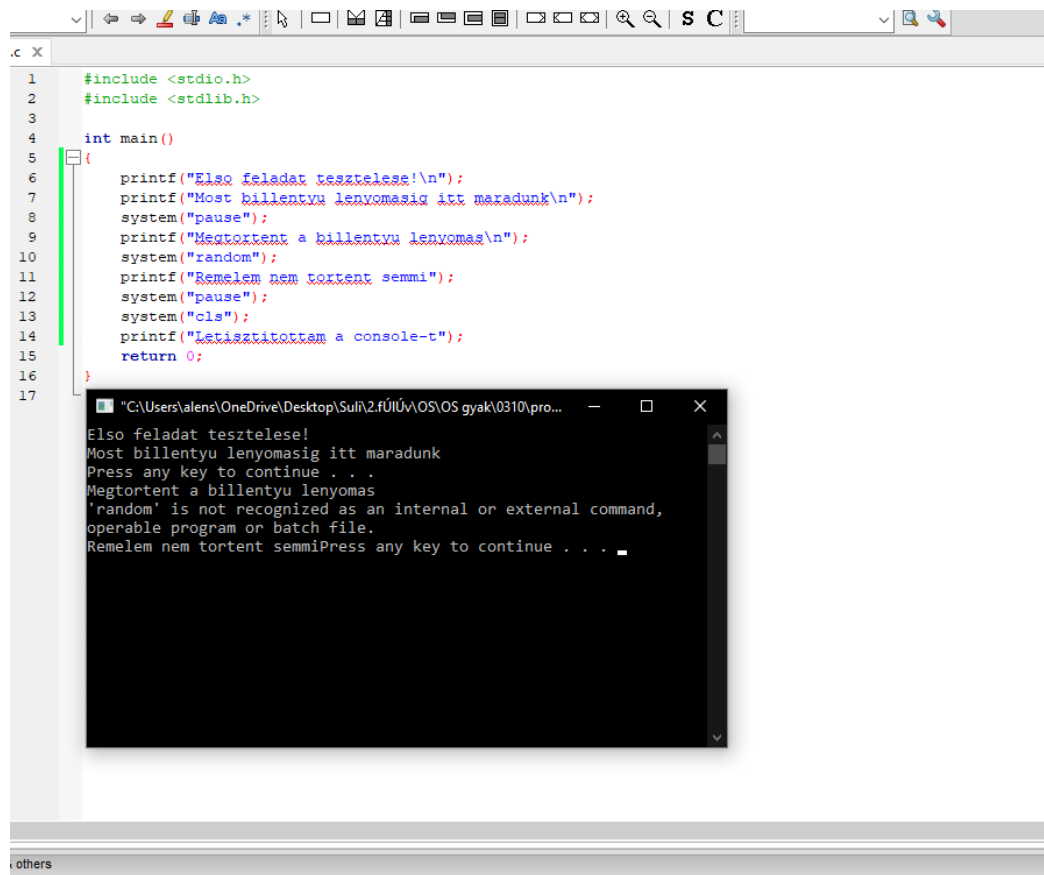
Szabó Alen Bsc

Progterv. info.

MX6WLR

Miskolc, 2021

1. feladat system rendszerhívás



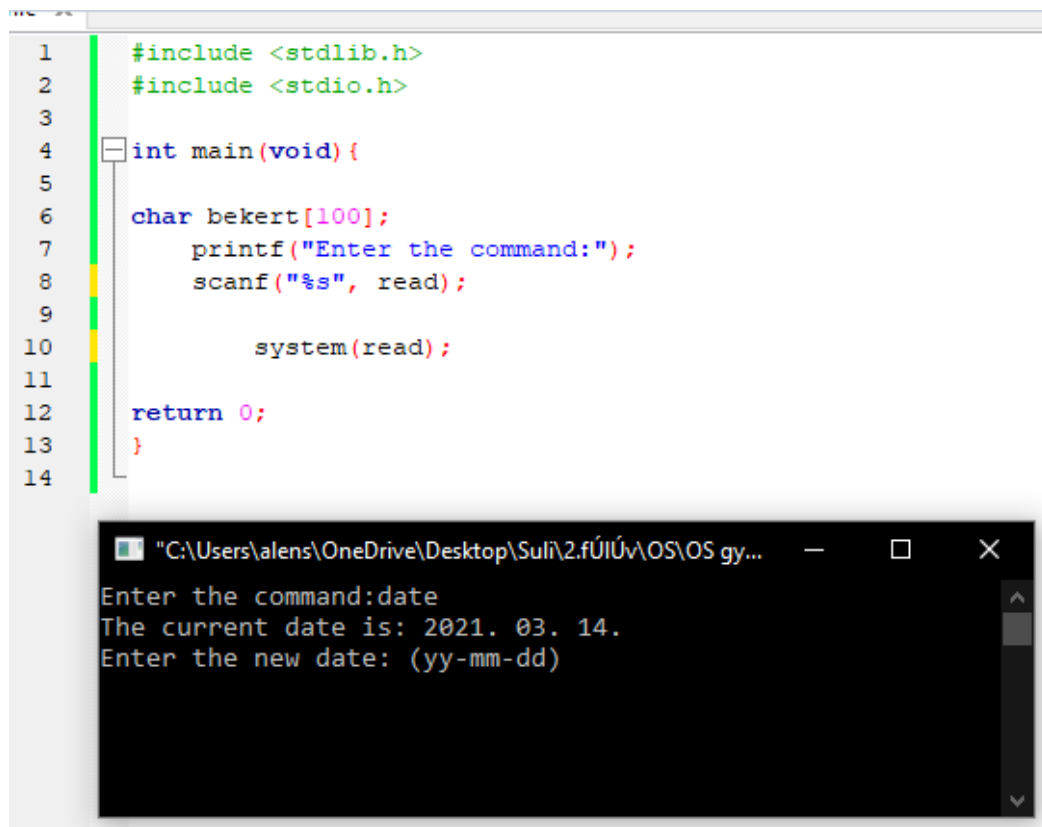
The image shows a C program in a code editor and its execution in a console window. The code is as follows:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     printf("Elso feladat tesztelese!\n");
7     printf("Most billentyu lenyomasig itt maradunk\n");
8     system("pause");
9     printf("Megtortent a billentyu lenyomas\n");
10    system("random");
11    printf("Remelem nem tortent semmi");
12    system("pause");
13    system("cls");
14    printf("Letisztitottam a console-t");
15    return 0;
16 }
17
```

The console window output is:

```
"C:\Users\alens\OneDrive\Desktop\Suli\2.fu\Uv\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



The image shows a C program in a code editor and its execution in a terminal window. The code is a simple program that prompts the user for a command and executes it using the `system()` function. The terminal output shows the user entering the command `date`, and the program displaying the current date and time.

```
1  #include <stdlib.h>
2  #include <stdio.h>
3
4  int main(void) {
5
6      char bekert[100];
7      printf("Enter the command:");
8      scanf("%s", read);
9
10     system(read);
11
12     return 0;
13 }
14
```

Terminal Output:

```
"C:\Users\alens\OneDrive\Desktop\Suli\2.füüüv\OS\OS gy...
Enter the command:date
The current date is: 2021. 03. 14.
Enter the new date: (yy-mm-dd)
```

3. feladat

```
start here X parent.c X
1  #include <unistd.h>
2  #include <stdlib.h>
3  #include <stdio.h>
4  #include <sys/types.h>
5  #include <sys/time.h>
6
7  int main (void) {
8      pid_t pid;
9
10     if ((pid = fork()) < 0){
11         perror("process error");
12     }else if (pid == 0){
13         if(execl("./child", "child", (char *) NULL) < 0){
14             perror("execl error");
15         }
16     }
17     if (waitpid(pid, NULL, 0) < 0){
18         perror("wait error");
19     }
20     return 0;
21 }
22
```

```
start here X child.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <unistd.h>
4
5  int main()
6  {
7      for(int i = 0; i < 5; i++){
8          printf("SzA MX6WLR\n");
9      }
10     return 0;
11 }
12
```

4. feladat fork ()

```
sin.c X
1  #include <stdio.h>
2  #include <sys/types.h>
3
4  #define MAX_COUNT 200
5
6  void ChildProcess(void);          /* child process prototype */
7  void ParentProcess(void);        /* parent process prototype */
8
9  void main(void)
10 {
11     pid_t pid;
12
13     pid = fork();
14     if (pid == 0)
15         ChildProcess();
16     else
17         ParentProcess();
18 }
19
20 void ChildProcess(void)
21 {
22     int i;
23
24     for (i = 1; i <= MAX_COUNT; i++)
25         printf("    This line is from child, value = %d\n", i);
26     printf("    *** Child process is done ***\n");
27 }
28
29 void ParentProcess(void)
30 {
31     int i;
32
33     for (i = 1; i <= MAX_COUNT; i++)
34         printf("This line is from parent, value = %d\n", i);
35     printf("*** Parent is done ***\n");
36 }
37
```

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;
}
```