

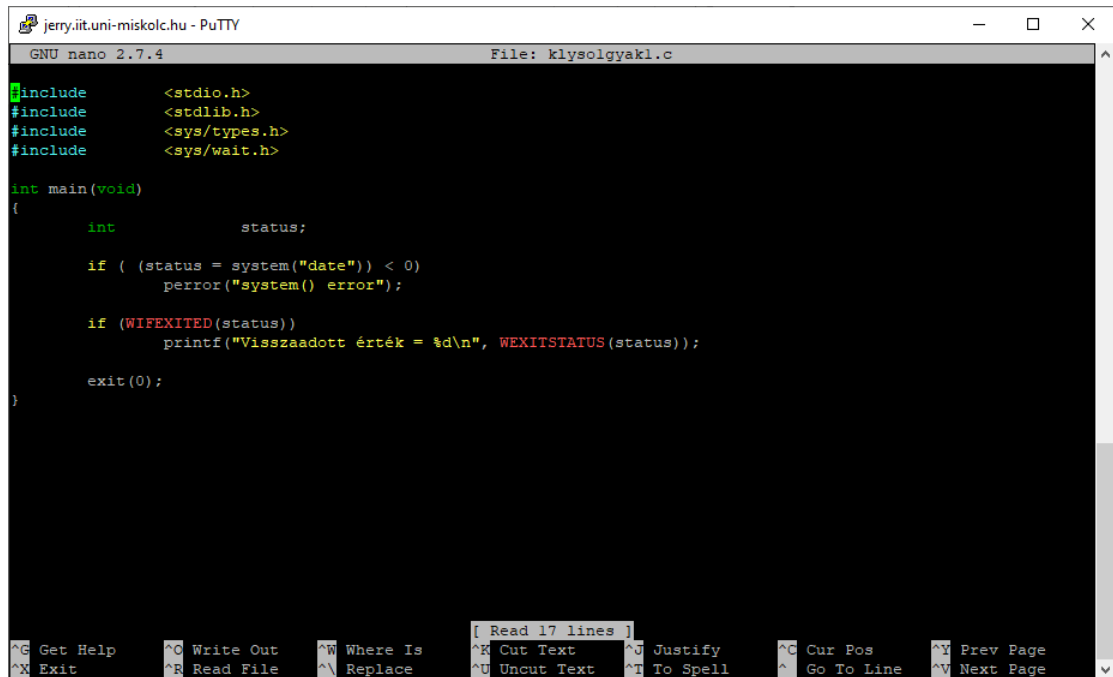
Linux OS – Rendszerhívások

(Repoban megtalálhatóak a file-ok)

1. A `system()` rendszerhívással hajtson végre létező és nem létező parancsot, és vizsgálja a visszatérési értéket!

Mentés: `neptunkodgyak1.c`

Kód(létező command):



```
GNU nano 2.7.4 File: klysolgyak1.c
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/wait.h>

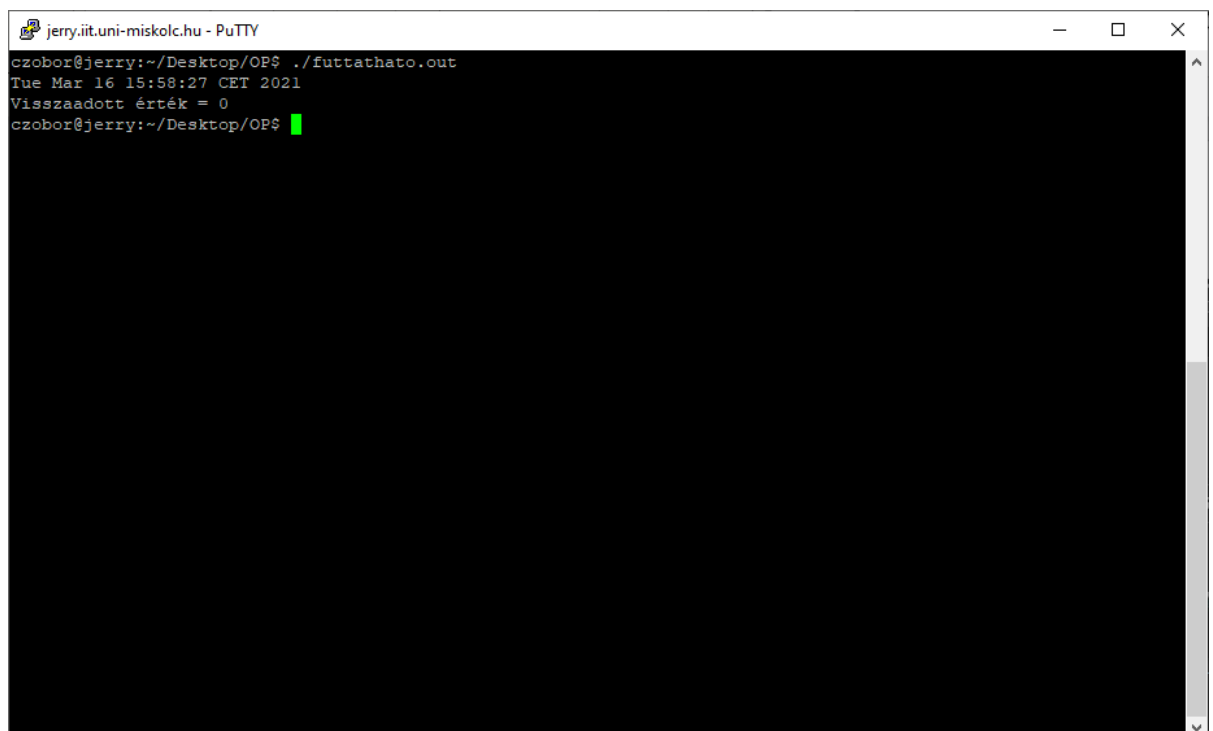
int main(void)
{
    int status;

    if ( (status = system("date")) < 0)
        perror("system() error");

    if (WIFEXITED(status))
        printf("Visszaadott érték = %d\n", WEXITSTATUS(status));

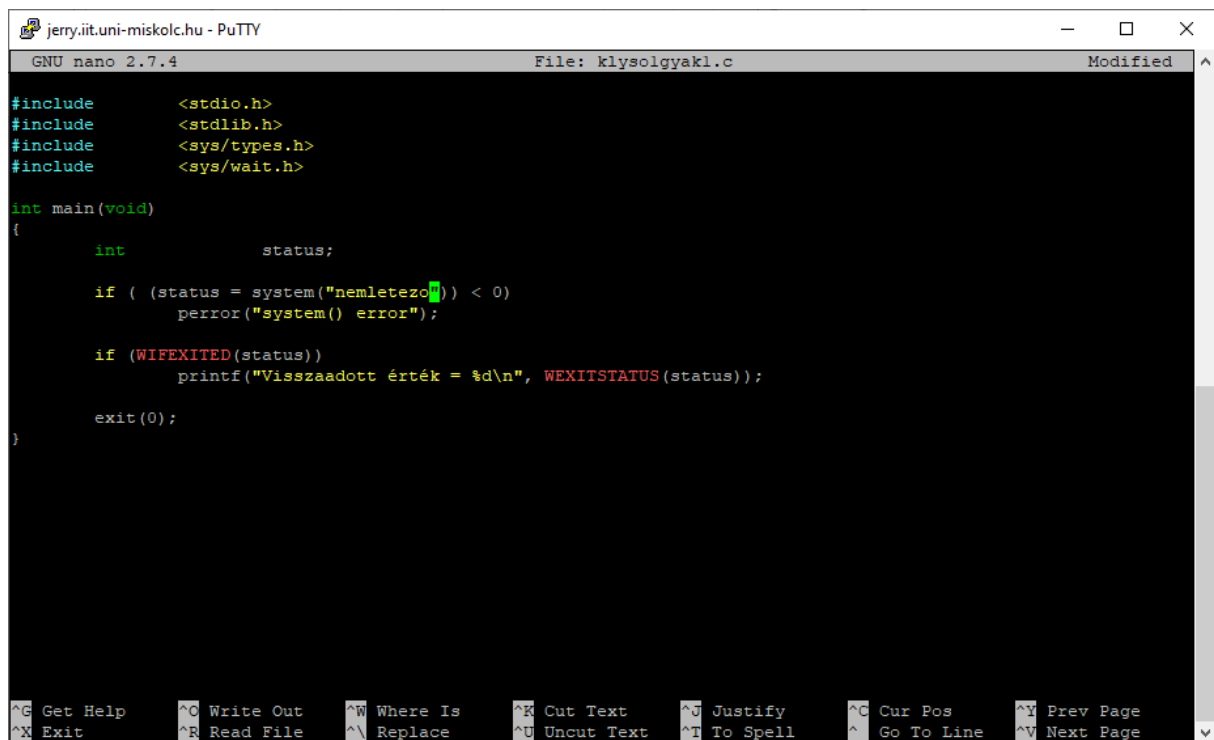
    exit(0);
}
```

Output:



```
jerry.iit.uni-miskolc.hu - PuTTY
czobor@jerry:~/Desktop/OP$ ./futtathato.out
Tue Mar 16 15:58:27 CET 2021
Visszaadott érték = 0
czobor@jerry:~/Desktop/OP$
```

Kód(nem létező command):



```

jerry.iit.uni-miskolc.hu - PuTTY
GNU nano 2.7.4                                File: klysolgyakl.c                                Modified
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/wait.h>

int main(void)
{
    int status;

    if ( (status = system("nemletozo")) < 0)
        perror("system() error");

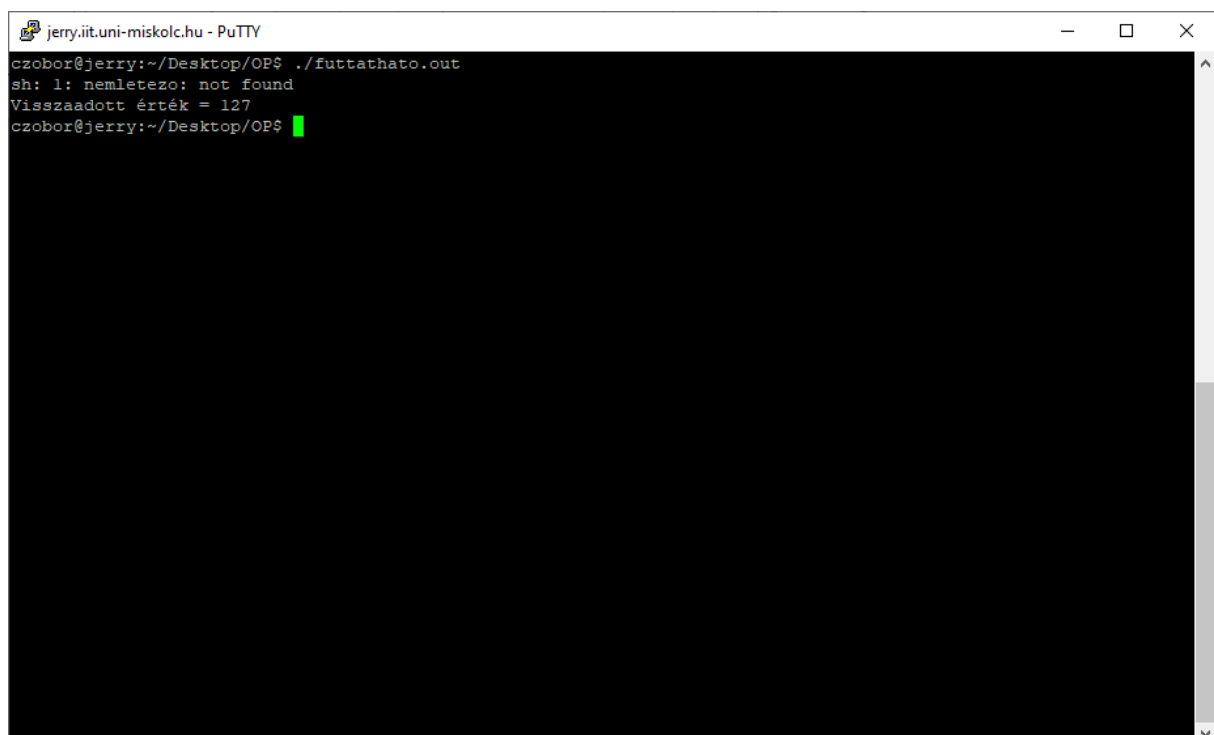
    if (WIFEXITED(status))
        printf("Visszaadott érték = %d\n", WEXITSTATUS(status));

    exit(0);
}

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos      ^Y Prev Page
^X Exit          ^R Read File    ^_ Replace      ^U Uncut Text   ^T To Spell     ^_ Go To Line    ^V Next Page

```

Output:



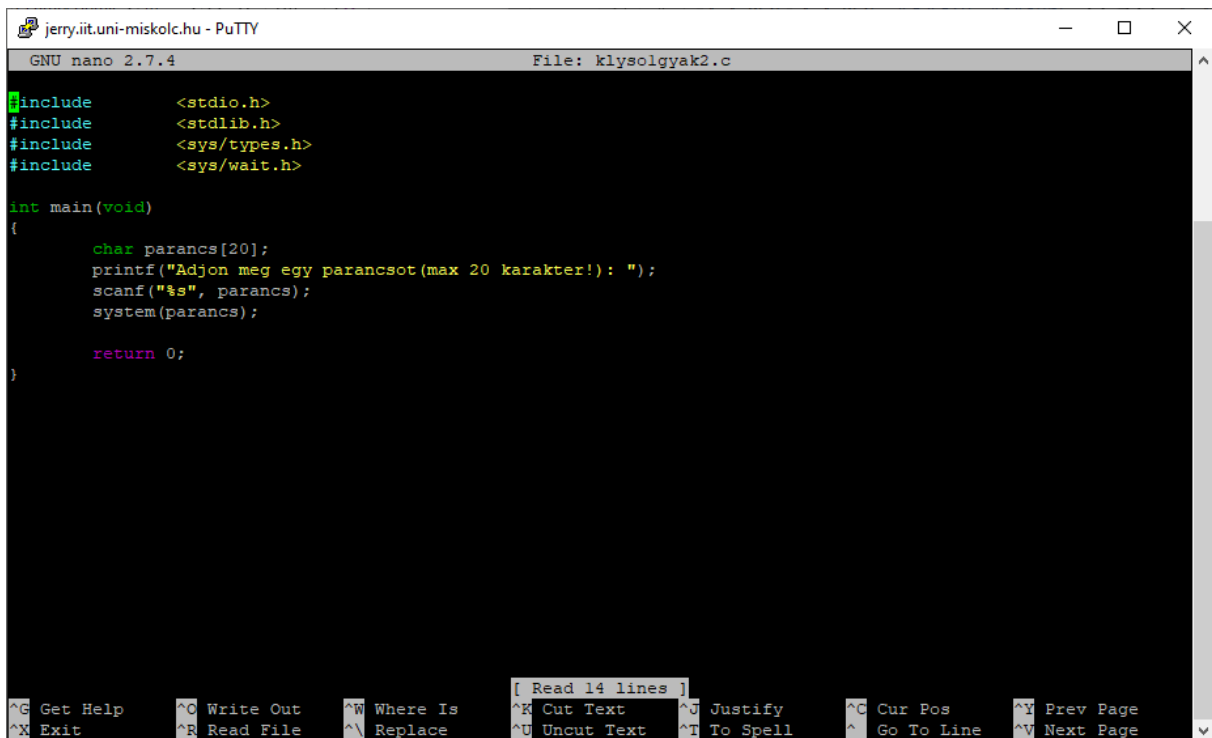
```

jerry.iit.uni-miskolc.hu - PuTTY
czobor@jerry:~/Desktop/OP$ ./futtathato.out
sh: 1: nemletozo: not found
Visszaadott érték = 127
czobor@jerry:~/Desktop/OP$

```

2. Írjon programot, amely billentyűzetről bekér Unix parancsokat és végrehajtja őket, majd kiírja a szabványos kimenetre. Mentés: neptunkodgyak2.c

Megjegyzés: printf() függvényhívással kiírja, hogy mit csinál ez a kis program, aztán scanf()-et meghívva beolvasson egy max 20 karakteres commandot.

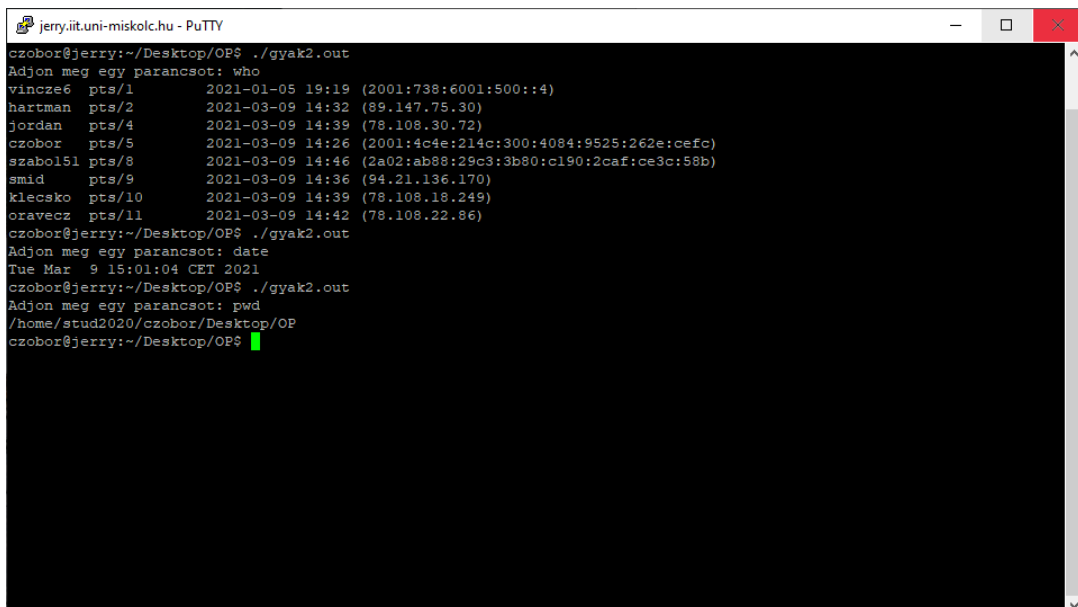


```
GNU nano 2.7.4 File: klysolgyak2.c

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

int main(void)
{
    char parancs[20];
    printf("Adjon meg egy parancsot(max 20 karakter!): ");
    scanf("%s", parancs);
    system(parancs);

    return 0;
}
```



```
czobor@jerry:~/Desktop/OP$ ./gyak2.out
Adjon meg egy parancsot: who
vincze6 pts/1      2021-01-05 19:19 (2001:738:6001:500::4)
hartman pts/2      2021-03-09 14:32 (89.147.75.30)
jordan pts/4      2021-03-09 14:39 (78.108.30.72)
czobor pts/5      2021-03-09 14:26 (2001:4c4e:214c:300:4084:9525:262e:cefc)
szabo151 pts/8      2021-03-09 14:46 (2a02:ab88:29c3:3b80:c190:2caf:ce3c:58b)
smid pts/9      2021-03-09 14:36 (94.21.136.170)
klecsko pts/10     2021-03-09 14:39 (78.108.18.249)
oravecz pts/11     2021-03-09 14:42 (78.108.22.86)
czobor@jerry:~/Desktop/OP$ ./gyak2.out
Adjon meg egy parancsot: date
Tue Mar  9 15:01:04 CET 2021
czobor@jerry:~/Desktop/OP$ ./gyak2.out
Adjon meg egy parancsot: pwd
/home/stud2020/czobor/Desktop/OP
czobor@jerry:~/Desktop/OP$
```

Készítsen egy parent.c és egy 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 (5-ször) (pl. a hallgató neve és a neptunkód)! Mentés: parent.c, ill. child.c

és (Mivel exel()-t használtam a negyedik feladatot is teljesítettem)

A fork() rendszerhívással hozzon létre egy gyerek processzt és abban hívjon meg egy

exec családbeli rendszerhívást (pl. `execvp`) egy unix-paranccsal. A szülő várja meg a gyerek

futását! Mentés: neptunkodgyak4.c

[illegible]

5.

A `fork()` rendszerhívással hozzon létre gyerekeket, várja meg és vizsgálja a befejeződési

állapotokat (gyerekekben: `exit`, `abort`, nullával való osztás)!

Mentés: `neptunkodgyak5.c`

Megjegyzés: A `parent2.c` -ben megnézi, hogy az `i` egyenlő-e 2-vel, ha igen akkor 1-et ad vissza, ha nem akkor 0 a visszatérési érték. `Abort()`-al visszatérési érték nélkül lépne ki a gyerek programból, `exit(n)` -el pedig `n`-el térne vissza.

Kód(Visszatérési érték 1):



```
GNU nano 2.7.4 File: child2.c

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

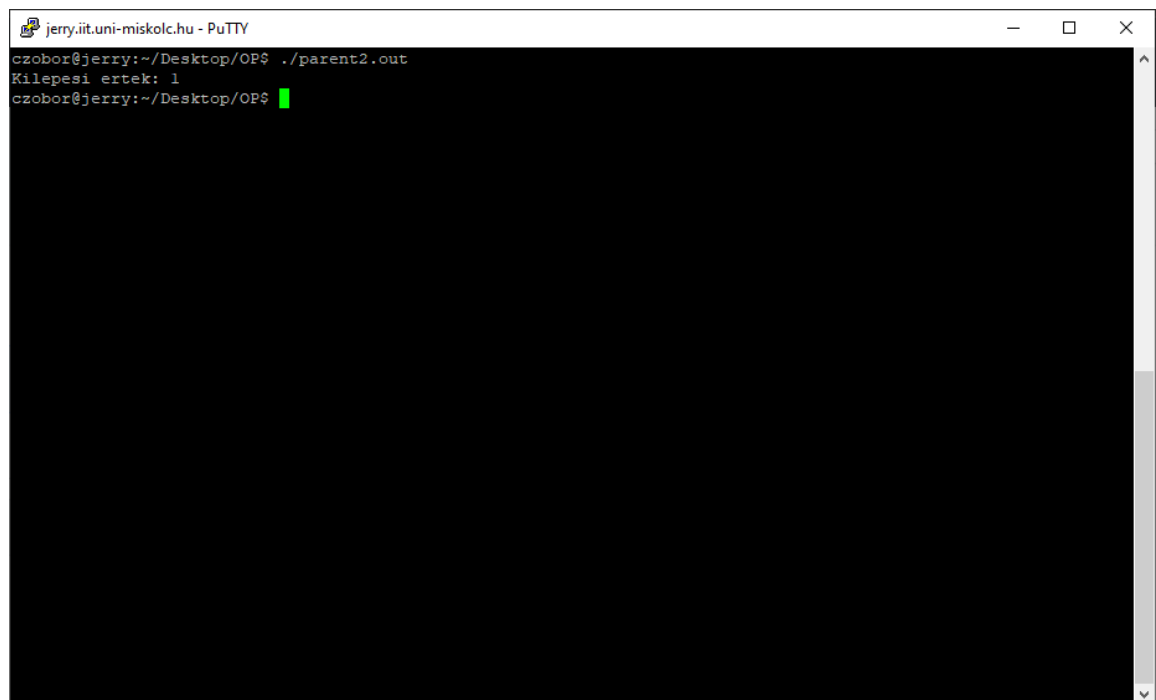
int main() {
    int i = 2;
    //int i = 3;

    if(i == 2) {
        return 1;
    } else {
        return 0;
    }
}

[Wrote 15 lines]
```

^G Get Help ^C Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos ^Y Prev Page
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line ^V Next Page

Output:



```
jerry.iit.uni-miskolc.hu - PuTTY
czobor@jerry:~/Desktop/OP$ ./parent2.out
Kilepesi ertek: 1
czobor@jerry:~/Desktop/OP$
```

Kód(Visszatérési érték 0):



```
jerry.iit.uni-miskolc.hu - PuTTY
GNU nano 2.7.4 File: child2.c


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

int main() {
    //int i = 2;
    int i = 3;

    if(i == 2) {
        return 1;
    } else {
        return 0;
    }
}

[ Wrote 15 lines ]
^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    ^Y Prev Page
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line  ^V Next Page
```

Output:

 jerry.iit.uni-miskolc.hu - PuTTY

```
czobor@jerry:~/Desktop/OP$ ./parent2.out  
Kilepesi ertek: 0  
czobor@jerry:~/Desktop/OP$
```