Operációs rendszerek BSc

9. Gyak.

2022. 04. 04.

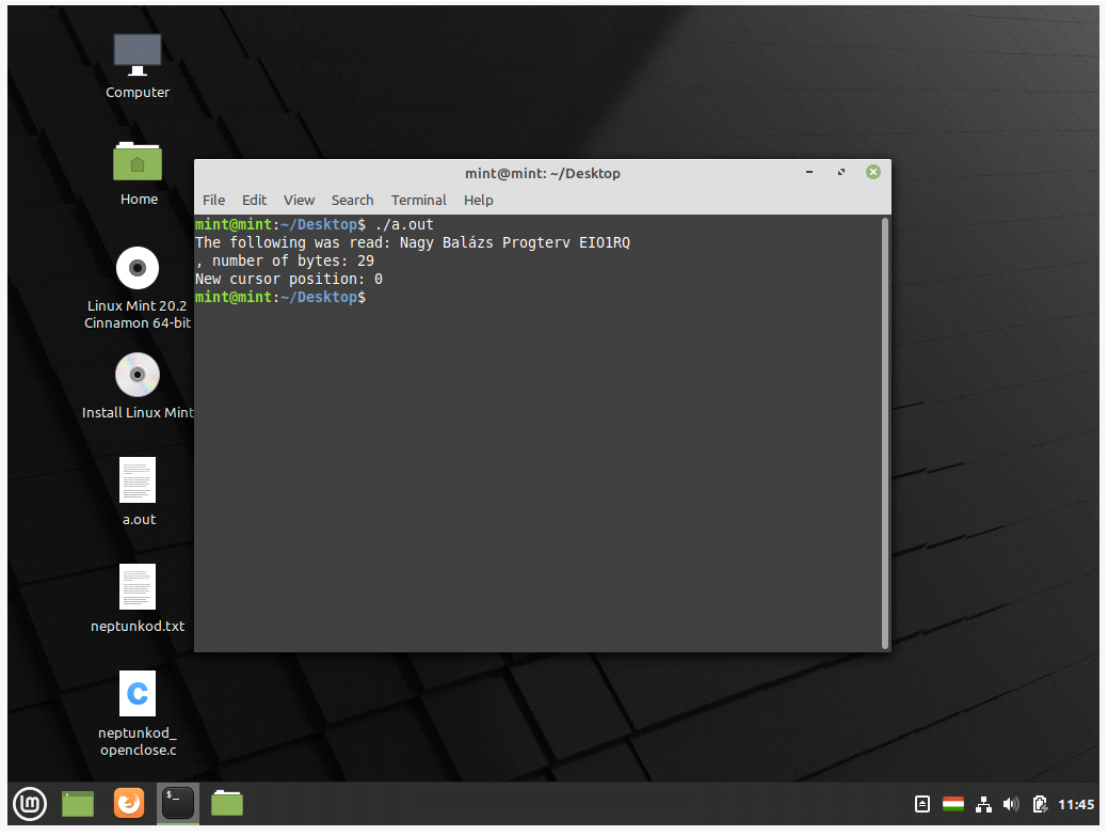
Készítette:

Nagy Balázs Bsc
Programtervező
informatikus
EIO1RQ

Miskolc, 2022

1. feladat – A tanult rendszerhívásokkal (open(), read()/write(), close() - ők fogják a rendszerhívásokat tovább hívni - írjanak egy neptunkod_openclose.c programot, amely megnyit egy fájlt – neptunkod.txt, tartalma: hallgató neve, szak , neptunkod.

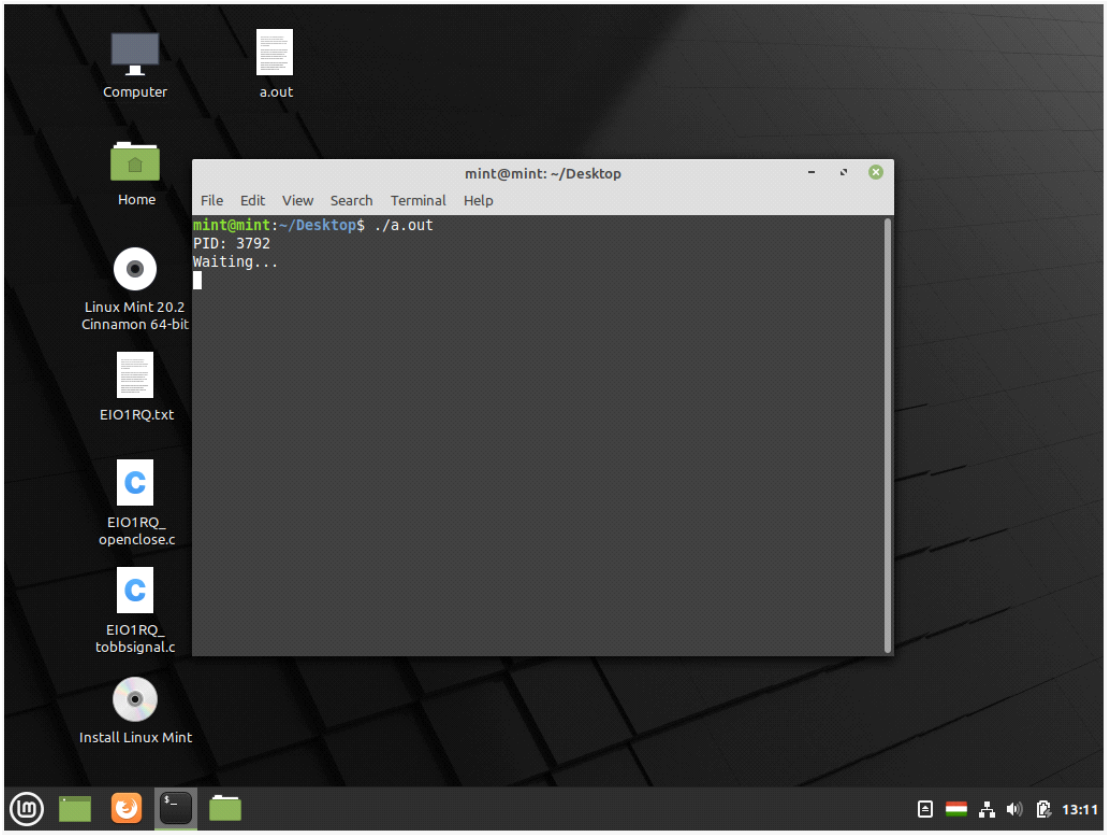
A program következő műveleteket végezze:
- olvassa be a neptunkod.txt fájlt, melynek attribútuma: O_RDWR
- hiba ellenőrzést,
- write() - mennyit ír ki a konzolra.
- read() - kiolvassa a neptunkod.txt tartalmát és mennyit olvasott ki (byte), és kiírja konzolra.
- lseek() – pozícionálja a fájl kurzor helyét, ez legyen a fájl eleje: SEEK_SET, és kiírja a konzolra.



2. feladat - Készítse el a következő feladatot, melyben egy szignálkezelő több szignált is tud kezelni:

a.) Készítsen egy szignál kezelőt (handleSignals), amely a SIGINT (CTRL + C) vagy SIGQUIT (CTRL + \) jelek fogására vagy kezelésére képes.
b.) Ha a felhasználó SIGQUIT jelet generál (akár kill paranccsal, akár billentyűzetről a CTRL + \) a kezelő egyszerűen kiírja az üzenetet visszatérési értékét – a konzolra.
c.) Ha a felhasználó először generálja a SIGINT jelet (akár kill paranccsal, akár billentyűzetről a CTRL + C), akkor a jelet úgy módosítja, hogy a következő alkalommal alapértelmezett műveletet hajtson végre (a SIG_DFL) – kiírás a konzolra.

d.) Ha a felhasználó másodszor generálja a SIGINT jelet, akkor végrehajt egy alapértelmezett műveletet, amely a program befejezése - kiírás a konzolra.
Mentés: neptunkod_tobbszignal.c



3. feladat - Adott a következő ütemezési feladat, amit a FCFS, SJF és Round Robin (RR: 4 ms) ütemezési algoritmus alapján határozza meg következő teljesítmény értékeket, metrikákat (külön-külön táblázatba):

FCFS	P1	P2	P3	P4
Érkezés	0	0	2	5
CPU idő	24	3	6	3
Indulás	0	24	27	33
Befejezes	24	27	33	36
Várakozás	0	24	25	28
Kör.ford	24	27	31	31
Válasz	0	24	25	28

Algoritmus neve	
Cpu kihasználtság	98,90%
Körülfordulási idő	28,25
Várakozási idők átlaga	19,25
Válaszidől átlaga	19,25

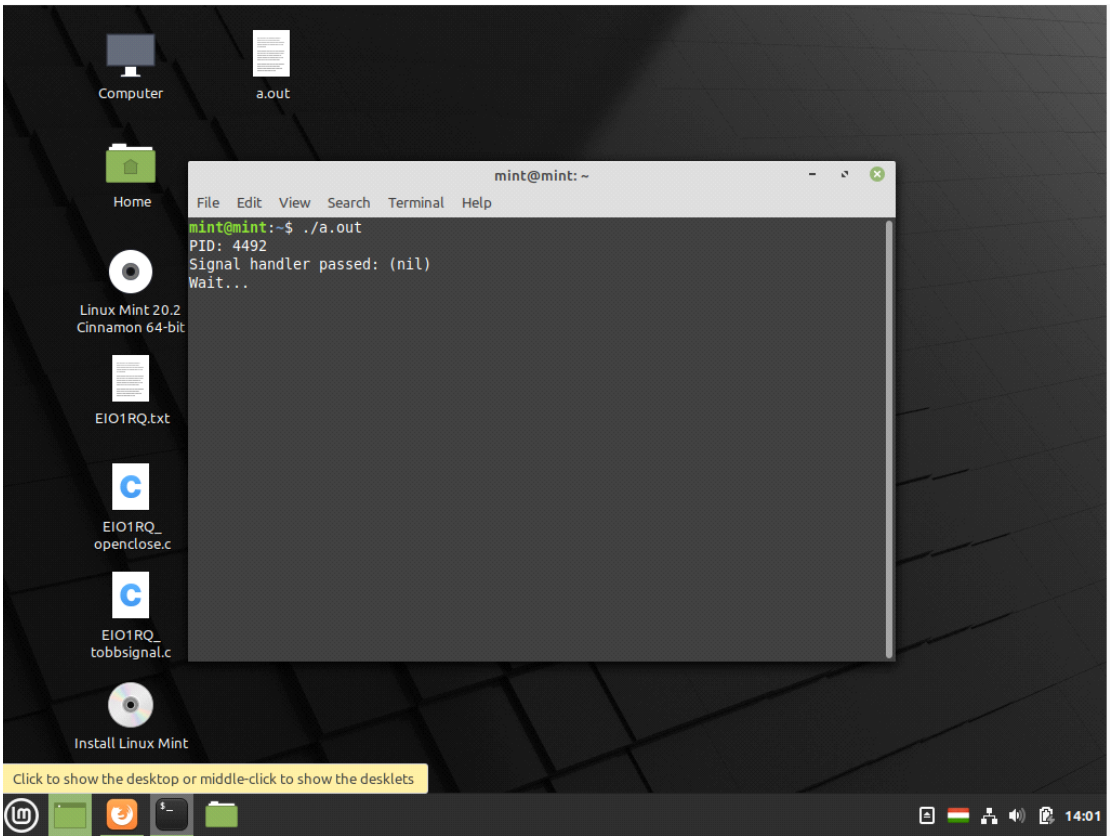
SJF	P1	P2	P3	P4
Érkezés	0	0	2	5
CPU idő	24	3	6	3
Indulás	12	0	3	9
Befejezes	36	3	9	12
Várakozás	12	0	1	4
Kör.ford	36	3	7	7
Válasz	12	0	1	4

Algoritmus neve	
Cpu kihasználtság	98,90%
Körülfordulási idő	13,25
Várakozási idők átlaga	4,25
Válaszidől átlaga	4,25

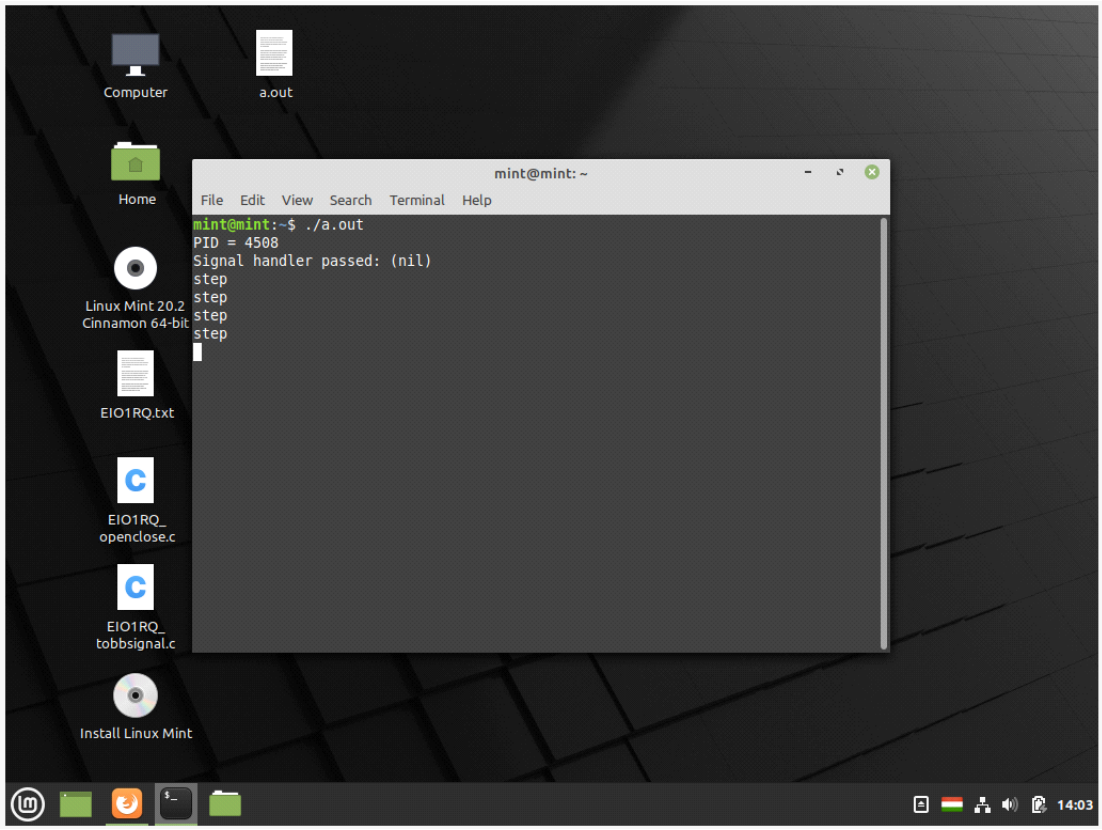
RR	P1	P2	P3	P4
Érkezés	0,4, 15	0	2,11	5
CPU idő	24,20,16	3	6,2	3
Indulás	0,11,20	4	7,18	15
Befejezes	0,7,5	7	11,2	18
Várakozás	0	4	5,7	10
Kör.ford	36	7	18	13

Algoritmus neve	
Cpu kihasználtság	93,70%
Körülfordulási idő	18,5
Várakozási idők átlaga	9,5
Válaszidől átlaga	4,75

1. gyakorlati feladat - Írjon C nyelvű programot, amelyik kill() seg.-vel SIGALRM-et küld egy argumentumként megadott PID-u processznek, egy másik futó program a SIGALRM-hez rendeljen egy fv.-t amely kiírja pl. neptunkodot, továbbá pause() fv.-el blokkolódjon, majd kibillenés után jelezze, hogy kibillent és terminálódjon.
Mentés. neptunkod_gyak9_1.c



2. gyakorlati feladat - Írjon C nyelvű programot, amelyik a SIGTERM-hez hozzárendel egy fv-t., amelyik kiírja az int paraméter értéket, majd végtelen ciklusban fusson, 3 sec-ig állandóan blokkolódva elindítás után egy másik shell-ben kill paranccsal (SIGTERM) próbálja terminálni, majd SIGKILL-el.” Mentés. neptunkod_gyak9_2.c


Computer
a.out
Home
mint@mint: ~
File Edit View Search Terminal Help
mint@mint:~$ ./a.out
PID = 4508
Signal handler passed: (nil)
step
step
step
step
Linux Mint 20.2
Cinnamon 64-bit
EIO1RQ.txt
EIO1RQ_
openclose.c
EIO1RQ_
tobbsignal.c
Install Linux Mint
14:03