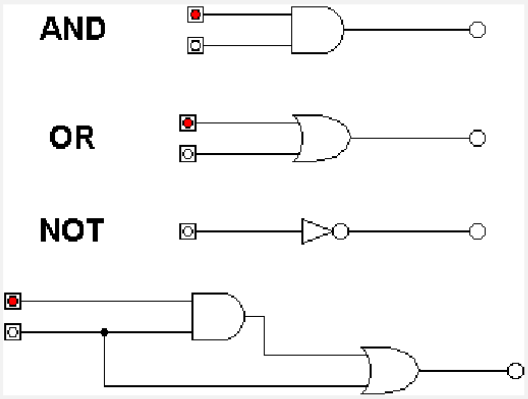
Løsningsforslag eksamen 2025 Operativsystemer

**Datamaskinarkitektur**

## 1)

1 **Fire kretser**

Hvilke av de fire kretsene i figuren under har riktig output?
(*Rød = 1 og hvit = 0, flere riktige svar er mulig. Feil svar gir trekk, men ikke mulig å få negativ score*)




**Velg ett eller flere alternativer**

- [x] AND
- [ ] OR
- [ ] NOT
- [x] AND-OR krets

Riktig. 10 av 10 poeng. Prøv igjen

## 2)

2 **Boolsk algebra**

Hva kan uttrykket AB + AB forenkles til med Boolsk algebra?

**Velg ett alternativ**

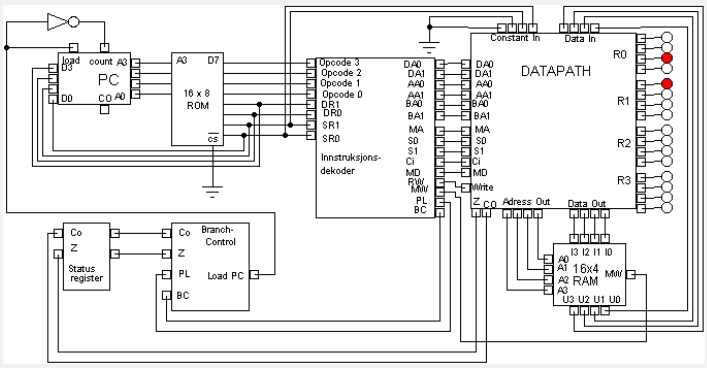
- 0
- B
- A + B
- A
- AB ✓
- Det kan ikke forenkles
- AB + BA
- 1

Riktig. 10 av 10 poeng. Prøv igjen

3)

3 **Datamaskinarkitektur**

Hvilken del av denne CPU'en inneholder logikken som oversetter maskininstruksjonene til signaler som gjør at de riktige operasjonene utføres i ALU mellom de riktige registerene?

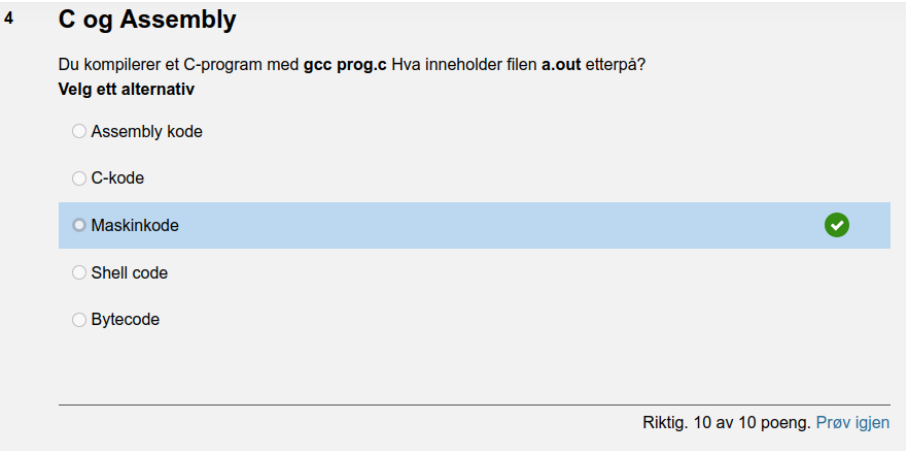



**Velg ett alternativ**

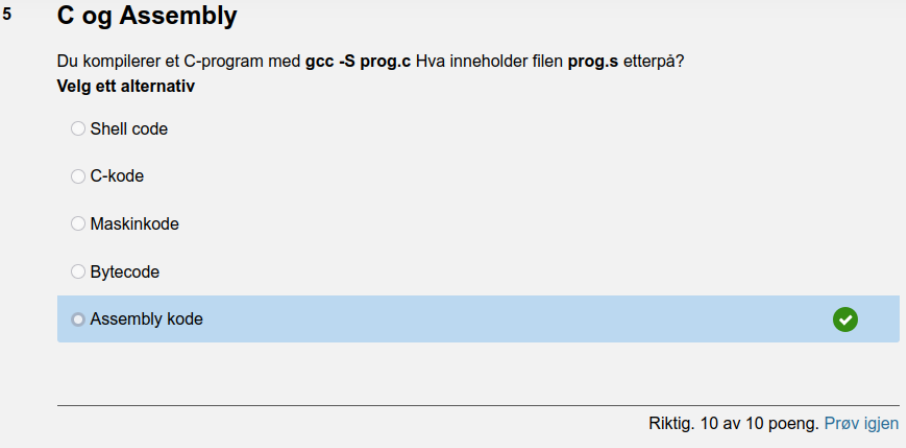
- ROM
- Ingen av delene
- PC
- RAM
- DATAPATH
- Branch-control
- Instruksjonsdekoder ✓

Riktig. 10 av 10 poeng. Prøv igjen

## 4)




## 5)




### Linux i praksis

## 6)

```
kan25@os:~$ cat file1
dkU7g
```

## 7)

```
kan25@os:~$ chmod 644 chmod.txt
```

## 8)

```
kan25@os:~$ man grep
-E, --extended-regexp
       Interpret PATTERNS as extended regular expressions (EREs, see below).
-R, --dereference-recursive
       Read all files under each directory, recursively.  Follow all symbolic links, unlike -r.
```

```
-I     Process  a  binary  file  as  if  it  did  not  contain  matching  data;  this  is  equivalent  to
       --binary-files=without-match option.
```

Så -E er for å kunne bruke denne typen regulært uttrykk, -R er for at det skal letes rekursivt nedover i fil-treet og -I er for å unngå treff fra binære filer.

## 9)

```
kan25@os:~/mapper25$ grep -REI ''\b\w{5}\b''
halt.target.wants/card/gv100/sony/ex.tex:ezDTF
```

## 10)

```
kan25@os:~$ sudo su
sudo: unable to resolve host os: Temporary failure in name resolution
root@os:/home/kan25# service docker start
 * Starting Docker: docker       [ OK ]
 root@os:/home/kan25# docker ps -a
 CONTAINER ID   IMAGE     COMMAND               CREATED       STATUS                       PORTS     NAMES
 108f5a168c3b   osx25     ``tail -f /dev/null''   8 hours ago   Exited (137) 7 hours ago             xenodochial_g
 root@os:/home/kan25#
IMAGE = osx25
```

## 11)

```
root@os:/home/kan25# docker exec -it 108 bash
root@osx25:/# ls -l /home/ubuntu/
total 4
-rw-r--r-- 1 ubuntu ubuntu 6 May 11 14:50 fil.txt
root@osx25:/# cat /home/ubuntu/fil.txt
2cRHt
```

## 12)

```
root@osx25:~# service ssh start
 * Starting OpenBSD Secure Shell server sshd       [ OK ]
 root@osx25:~#
 exit
 root@os:/home/kan25#
 exit
 kan25@os:~$ ssh ubuntu@osx25
 The authenticity of host `osx25 (172.17.0.2)' can't be established.
 ED25519 key fingerprint is SHA256:do+ULMMXirLXAFj9duyky5wCdUuIbW7fc8/c0ZM8lrM.
 This key is not known by any other names
 Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
 Warning: Permanently added `osx25' (ED25519) to the list of known hosts.
 ubuntu@osx25's password:
 Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-59-generic x86_64)

  * Documentation:  https://help.ubuntu.com
   * Management:     https://landscape.canonical.com
    * Support:        https://ubuntu.com/pro

    This system has been minimized by removing packages and content that are
    not required on a system that users do not log into.
```

```
    To restore this content, you can run the `unminimize' command.

    The programs included with the Ubuntu system are free software;
    the exact distribution terms for each program are described in the
    individual files in /usr/share/doc/*/copyright.

    Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
    applicable law.

    ubuntu@osx25:~$ ls -l
    total 4
    -rw-r--r-- 1 ubuntu ubuntu 6 May 11 14:50 fil.txt
```

Vi har nå logget inn med ssh som brukeren ubuntu på containeren som vi tidligere startet og har havnet i ubuntu-brukeren sin hjemmemappe.

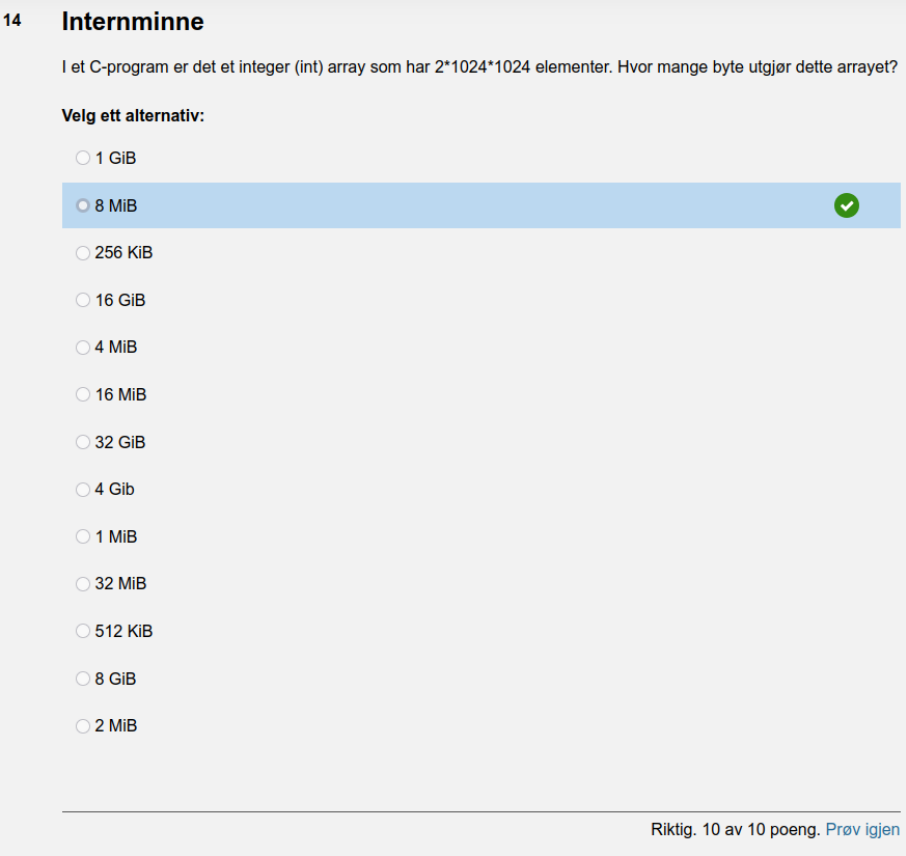
## 13)

```
kan25@os:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kan25/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kan25/.ssh/id_rsa
Your public key has been saved in /home/kan25/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:yGzRJ+wRWC85MMBbNWo3sEJUvwF1BUPWWL+CAK+XyCc kan25@os
The key's randomart image is:
+---[RSA 3072]----+
|   o+oB=*.**o    |
|   .. o& B{\ldots} .   |
|    .o= / o   .  |
|    .* B % .   . |
|      E S . . .  |
|     . +     .   |
|                 |
|                 |
|                 |
+----[SHA256]-----+
kan25@os:~$ ssh-copy-id ubuntu@osx25
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ``/home/kan25/.ssh/id_rsa.pub''
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new ke
ubuntu@osx25's password:

Number of key(s) added: 1

Now try logging into the machine, with:   ``ssh `ubuntu@osx25'''
and check to make sure that only the key(s) you wanted were added.

kan25@os:~$ ssh ubuntu@osx25
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-59-generic x86_64)

 * Documentation:  https://help.ubuntu.com
  * Management:     https://landscape.canonical.com
   * Support:        https://ubuntu.com/pro

   This system has been minimized by removing packages and content that are
```

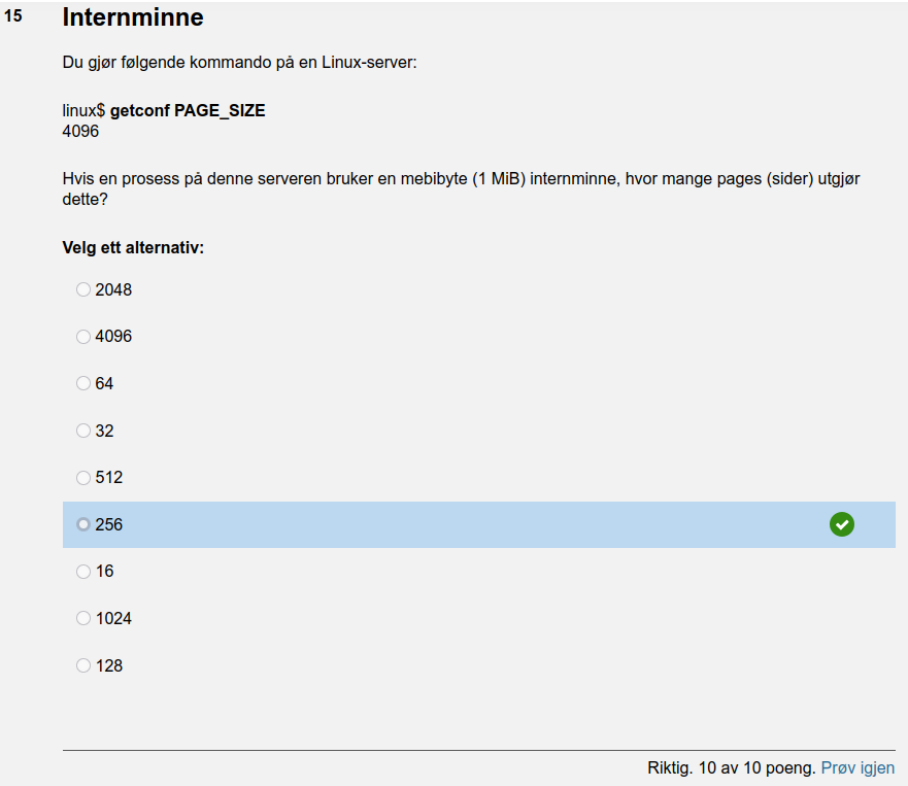
```
not required on a system that users do not log into.

To restore this content, you can run the `unminimize' command.
Last login: Sun May 11 22:29:10 2025 from 172.17.0.1
ubuntu@osx25:~$
```

### Internminne

## 14)




## 15)




**16)** Denne typen cache er et mellomlager mellom CPU-registere og RAM. Cache er laget av hurtig SRAM og er viktig fordi moderne CPUer prosesserer data så hurtig at en instruksjon vil bli ferdig lenge før man rekker å hente den neste fra RAM. L1 er minst og sitter nærmest registerne og CPUen, L2 er middels stor og L3 er den største delen som sitter nærmest RAM. Hvis man ikke har cache vil prosesseringen gå vesentlig sakter fordi CPU hele tiden må vente på nye instruskjoner.

**17)** Ja, flere CPUer har normalt tilgang til samme område av RAM. Kode som kjører på en kjerne kan flyttes til en annen kjerne og koden hentes da fra samme sted i RAM. (I store severe med veldig mange CPU-kjerner, kan noen av kjernene være tettere tilknyttet såkalte NUMA-noder og det går raskere å hente data derfa, men disse områdene av RAM kan også nås av alle kjerner)

### Bash scripting

**18)**

```
#! /bin/bash

if [ "$SHELL" != "" ]
then
     echo SHELL er definert til $SHELL
else
     echo "SHELL ikke definert"
fi
```

**19)** Hensikten med programmet er å finne ut hvilket passord som vil gi en gitt hash når passordet kjøres gjennom en hashing-algoritme. Programmet tester om passordet er ett av alle mulige tobokstavers ord ved å sende de små ordene som passord-input til mkpasswd med det riktige saltet som leses ut av hash'en. Hvis samme hash genereres er passordet riktig og da avsluttes programmet og passordet skrives ut.

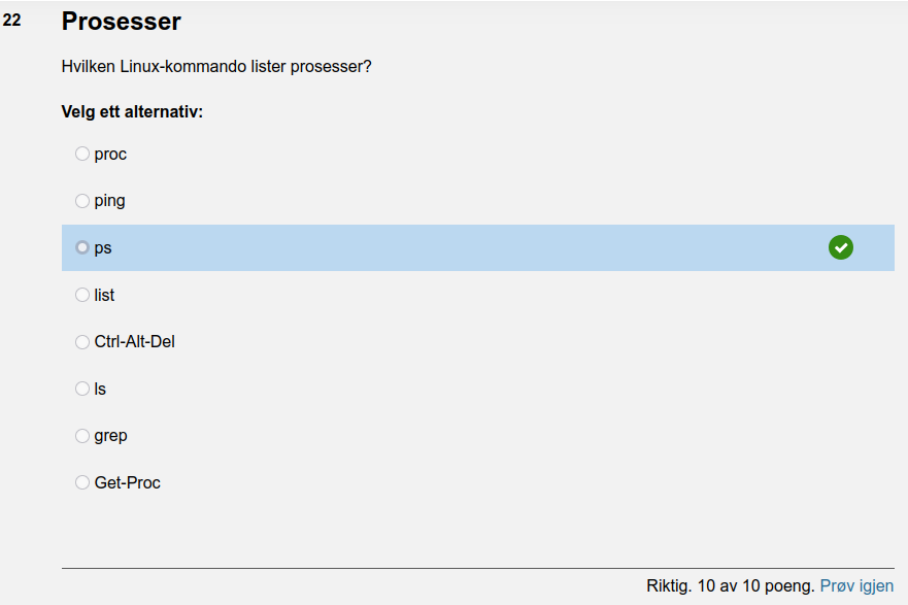
**20)** Får å istedet sende igjennom alle ordene i ordboken, kan man endre linje nr 6 i scriptet til

```
for pord in $(catordbok.txt)
```

**21)** Når man så kjører programmet, finner man passordet:

```
kan25@os:~/crack$ chmod 700 crack.sh
kan25@os:~/crack$ ./crack.sh
Passordet er "roser"
```

**22)**



22 **Prosesser**

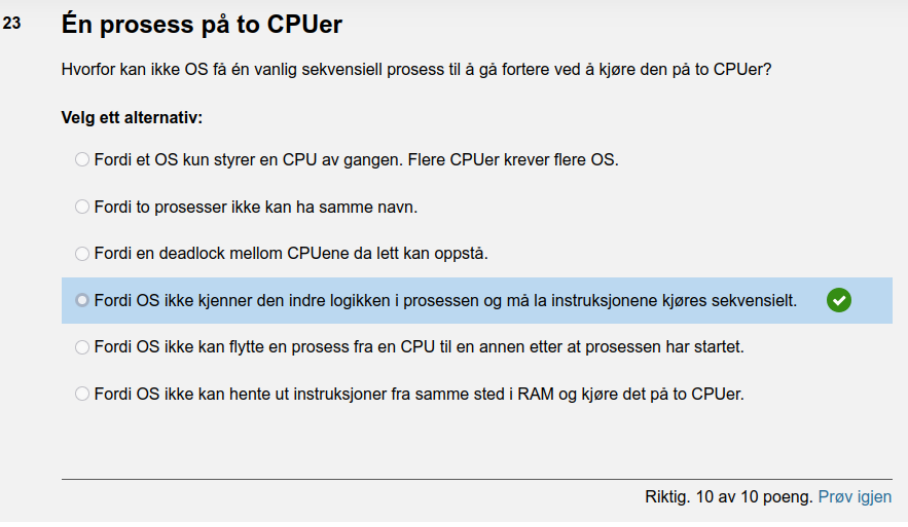
Hvilken Linux-kommando lister prosesser?

**Velg ett alternativ:**

- proc
- ping
- ps ✓
- list
- Ctrl-Alt-Del
- ls
- grep
- Get-Proc

Riktig. 10 av 10 poeng. Prøv igjen

**23)**



23 **Én prosess på to CPUer**

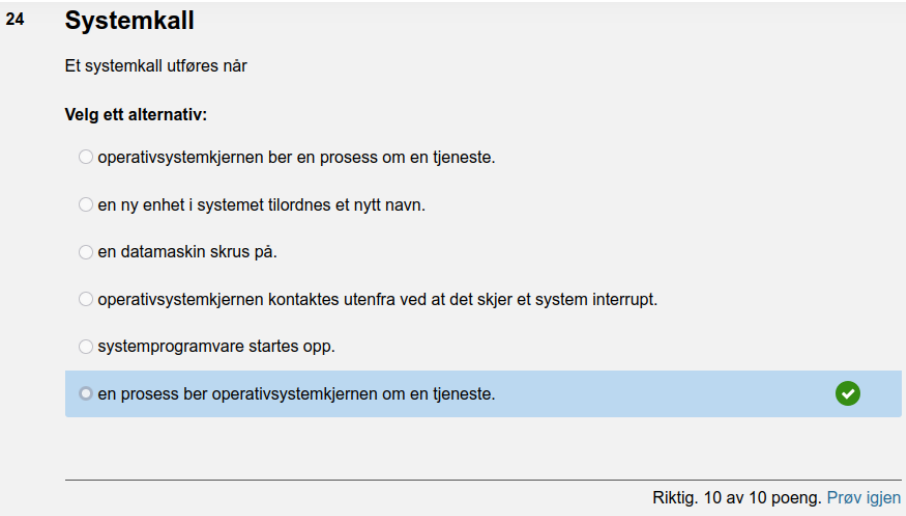
Hvorfor kan ikke OS få én vanlig sekvensiell prosess til å gå fortere ved å kjøre den på to CPUer?

**Velg ett alternativ:**

- Fordi et OS kun styrer en CPU av gangen. Flere CPUer krever flere OS.
- Fordi to prosesser ikke kan ha samme navn.
- Fordi en deadlock mellom CPUene da lett kan oppstå.
- Fordi OS ikke kjenner den indre logikken i prosessen og må la instruksjonene kjøres sekvensielt. ✓
- Fordi OS ikke kan flytte en prosess fra en CPU til en annen etter at prosessen har startet.
- Fordi OS ikke kan hente ut instruksjoner fra samme sted i RAM og kjøre det på to CPUer.

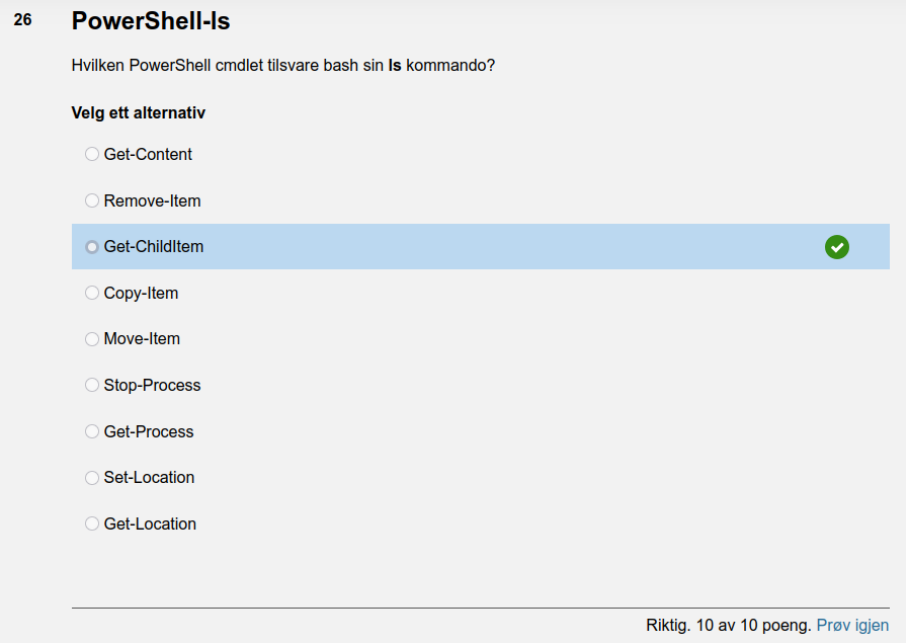
Riktig. 10 av 10 poeng. Prøv igjen

**24)**




**25)** Den første kjøringen viser at fire regne-prosesser kan kjøre samtidig og at hver av dem får omtrent 100% CPU-tid. Den neste kjøringen viser at når 8 prosesser kjøres samtidig så tar hver av dem omtrent dobbelt så lang tid på å fullføre. Dette tyder på at det er 4 kjerner og at to prosesser deler på hver sin kjerne. Men det rapporteres likevel om 100% CPU-bruk for hver av prosessene og det kan bare forklares med at CPUene må være hyperthreading. Dette er fordi da behandler OS en kjerne som om den har to uavhengige regneheter som hver av de to regnejobbene får full tilgang til. Men i virkeligheten deler de på den samme regne-enheten og det tar likevel dobbelt så lang tid. (og nytten av hyperthreading er liten i dette tilfellet, det ville ha sett omtrent likt ut på en server med 4 ikke-hyperhtreading kjerner men OS ville da ha rapportert om 50% bruk for 8 samtidige prosesser)

**26)**



**27)** Problemet er at i pwsh-shellet er ls programmet /usr/bin/ls og ikke en alias til Get-ChildItem. Dermed returnerer ls et streng-objekt og det har ikke et LastWriteTime property. Hvis man bytter ut ls med Get-ChildItem, fungerer scriptet etter hensikten:

```
PS /home/kan25/pwsh> ./passCheck.ps1
/etc/passwd ble endret for 0 dager og 9 timer siden
```