Eksamensinformasjon

Fakultet: Teknologi, kunst og design
Utdanning: Teknologiske Fag
Emnenavn: Operativsystemer
Emnekode: DATA2500

Bokmål

Dato: 30 juli 2020
Tid: kl. 09.00-12.00 (3 timer)

Hjelpemidler:
Alle hjelpemidler er tillatt.

Andre opplysninger:
Les nøye gjennom oppgavene før du begynner og pass på å besvare alle spørsmålene.
Sett gjerne egne forutsetninger dersom du synes oppgaveteksten er uklar. Beskriv forutsetningene og løs oppgaven utifra dem.

Maks poeng på de fleste av deloppgavene er 10 og de bidrar 3.33 % hver til sluttpoengsummen som maksimalt er 300. Men legg merke til at det er noen unntak. Følgende oppgaver gir inntil 20 poeng:

Oppgave 8, 9 og 10 om prosesser

Oppgave 19 om bash-scripting

Oppgave 23 om PowerShell-scripting

Oppgave 21 om bash-scripting gir inntil 30 poeng og teller totalt sett 10%.

Support:
Har du spørsmål til oppgaven må du ta kontakt med Hårek Haugerud via Zoom

Har du spørsmål om eller problemer med Inspera ta kontakt med eksamen@oslomet.no eller på telefon 67 23 50 10. Vi oppfordrer at kun akutte henvendelser kommer inn på telefon da denne har begrenset bemanning. Epost henvendelser vil bli behandlet fortløpende, husk å skrive inn studentnummer, kandidatnummer og emnekode i mailen for raskere behandling.

Opplever du problemer med innleveringen må du ta kontakt med eksamenskontoret **før innleveringsfristen går ut.**

1 Datamaskinarkitektur

Hvilken type logisk port er dette sannhetstabell for? A og B er input og UT er resultatet.

A	B	UT
0	0	0
0	1	1
1	0	1
1	1	1

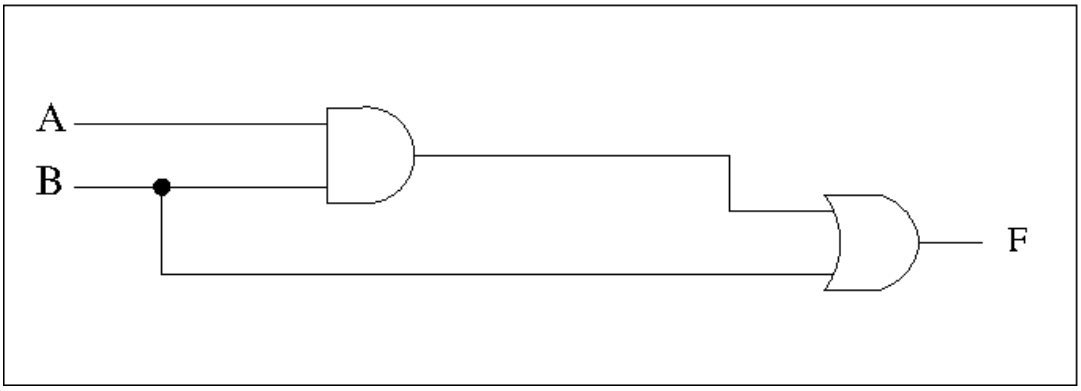
Velg ett alternativ

- PMOS
- XOR
- NOR
- OR
- NOT
- AND
- NMOS

Maks poeng: 10

2 Datamaskinarkitektur

Hva blir det logiske uttrykket F(A,B) for denne kretsen?



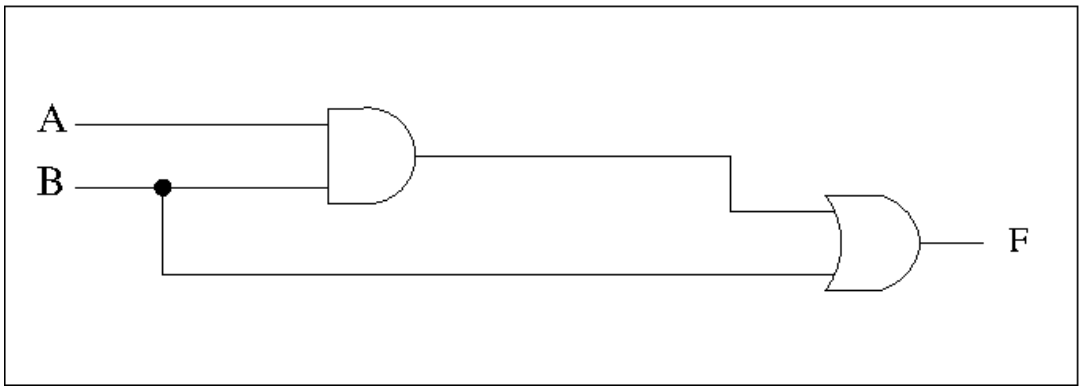
Velg ett alternativ

- $A \cdot B + B$
- $A \cdot B \cdot B$
- $(A + B) \cdot B$
- $(A + B) \cdot A$
- $\overline{A + B}$
- $A \cdot B + A$
- $\overline{B} \cdot A + \overline{B}$

Maks poeng: 10

3 Datamaskinarkitektur

Uttrykket for den kretsen kan forenkles, utifra sannhetstabellen eller ved å bruke boolsk algebra. Hva kan uttrykket for F(A,B) forenkles til?



Velg ett alternativ

- B
- 1
- $A \cdot B$
- A
- 0
- $\overline{B}$
- $\overline{A}$
- $A + B$

Maks poeng: 10

4 Threads

Du lager en java-thread klasse hvor du deklarerer to variabler:

int x;
static int y;

Deretter starter du opp mange slike threads. det vil da finnes:

Velg ett alternativ

- En felles variabel y og en felles variabel x
- En felles variabel x og en variabel y i hver thread
- En variabel x i hver thread og en variabel y i hver thread
- En felles variabel y og en variabel x i hver thread

Maks poeng: 10

5 Prosesser

I hvilken av følgende aktiviteter er OS-kjernen IKKE direkte involvert?
Velg ett alternativ

- Multitasking
- Hyperthreading
- Multithreading
- Multiprocessing
- Scheduling
- Context switching

Maks poeng: 10

6 Prosesser

I denne og de følgende oppgavene skal vi studere kjøringen av et 100% CPU-avhengig C-program som kun består av en lang for-løkke som regner og regner med heltall. Dette programmet har blitt kompilert og kjørt på tre forskjellige Linux-servere, som i teksten kalles linux1, linux2 og linux3.

Før kjøringen settes timeformat slik:
TIMEFORMAT='Real:%R User:%U System:%S %P%%'
for at output fra time-funksjonen skal bli mer kompakt.

Følgende viser hvordan programmet kjøres på linux1:

linux1$ **gcc -O sum.c**
linux1$ **time ./a.out**
Real:1.809 User:1.804 System:0.004 99.95%

Forklar kort hva kommandoene (vist med uthevet skrift) gjør og hva de fire tallene i output viser.

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 10

7 Prosesser

Følgende viser på linux1-serveren to forskjellige måter å kjøre programmet på:

linux1$ **for i in {1..4}**
> **do**
> **time ./a.out**
> **done**
Real:1.814 User:1.812 System:0.000 99.90%
Real:1.797 User:1.792 System:0.000 99.73%
Real:1.826 User:1.816 System:0.012 100.09%
Real:1.836 User:1.832 System:0.000 99.77%
linux1$

linux1$ **for i in {1..4}**
> **do**
> **time ./a.out&**
> **done**
[1] 1329
[2] 1330
[3] 1332
[4] 1334
linux1$
Real:1.803 User:1.796 System:0.000 99.63%
Real:1.816 User:1.812 System:0.000 99.75%
Real:1.854 User:1.828 System:0.020 99.69%
Real:1.866 User:1.848 System:0.012 99.67%

Forklar kort hva som skjer i de to kjøringene og forklar spesielt hva som er forskjellen på de to måtene å kjøre programmet på. Vil det være noen forskjell på hvor fort de to kjøringene fullføres? Forklar kort.

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 10

8 Prosesser

linux1

Det følgende viser først kompilering og kjøring av det samme 100% CPU-avhengige programmet og så tre forskjellige kjøringer av programmet på den første av serverene, linux1. Forklar med utgangspunkt i tiden fra den første kjøringen de tidene som er resultatet av de tre kjøringene. Vis ved utregning hvordan størrelsen på Real-tiden omtrentlig kan beregnes utifra resultatet 1.8 sekunder i den første kjøringen. Forklar også prosent-tallene for hver kjøring. Prøv utifra resultatene av kjøringen å finne ut hvor mange cores (med en egen ALU) denne serveren har og om hver core er hyperthreading eller ikke. Forklar kort hvordan du kommer frem til din konklusjon.

linux1$ **gcc -O sum.c**
linux1$ **time ./a.out**
Real:1.809 User:1.804 System:0.004 99.95%
linux1$ **for i in {1..4}**
> **do**
> **time ./a.out &**
> **done**
Real:1.819 User:1.812 System:0.000 99.59%
Real:1.833 User:1.812 System:0.004 99.08%
Real:1.833 User:1.812 System:0.000 98.85%
Real:1.855 User:1.836 System:0.004 99.21%

linux1$ **for i in {1..6}; do time ./a.out & done**
Real:2.510 User:1.800 System:0.000 71.70%
Real:2.617 User:1.800 System:0.000 68.77%
Real:2.661 User:1.804 System:0.008 68.09%
Real:2.686 User:1.828 System:0.004 68.21%
Real:2.806 User:1.800 System:0.008 64.42%
Real:2.832 User:1.840 System:0.000 64.98%

linux1$ **for i in {1..8}; do time ./a.out & done**
Real:3.553 User:1.816 System:0.000 51.10%
Real:3.594 User:1.844 System:0.000 51.30%
Real:3.625 User:1.828 System:0.004 50.53%
Real:3.623 User:1.812 System:0.016 50.45%
Real:3.642 User:1.820 System:0.008 50.19%
Real:3.650 User:1.812 System:0.004 49.75%
Real:3.671 User:1.820 System:0.008 49.80%
Real:3.746 User:1.836 System:0.004 49.11%

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 20

9 Prosesser

linux2

Det følgende viser først kompilering og kjøring av det samme 100% CPU-avhengige programmet og så tre forskjellige kjøringer av programmet på den andre av serverene, linux2. Forklar med utgangspunkt i tiden fra den første kjøringen de tidene som er resultatet av de tre kjøringene. Vis ved utregning hvordan størrelsen på Real-tiden omtrentlig kan beregnes utifra resultatet 0.6 sekunder i den første kjøringen. *Prøv i dette tilfellet spesielt å forklare forskjellene i Real-tid mellom prosessene i den andre av de tre kjøringene (***for i in {1..6}-**kjøringen). Forklar også prosent-tallene for hver kjøring. Prøv utifra resultatene av kjøringen å finne ut hvor mange cores (med en egen ALU) denne serveren har og om hver core er hyperthreading eller ikke. Forklar kort hvordan du kommer frem til din konklusjon.

linux2$ **gcc -O sum.c**
linux2$ **time ./a.out**
Real:0.614 User:0.612 System:0.000 99.63%

linux2$ **for i in {1..4}; do time ./a.out& done**
Real:0.634 User:0.628 System:0.000 99.06%
Real:0.633 User:0.628 System:0.000 99.15%
Real:0.634 User:0.628 System:0.000 98.97%
Real:0.637 User:0.624 System:0.000 97.98%

linux2$ **for i in {1..6}; do time ./a.out& done**
Real:0.624 User:0.620 System:0.000 99.41%
Real:0.623 User:0.620 System:0.000 99.48%
Real:0.908 User:0.904 System:0.000 99.58%
Real:0.911 User:0.900 System:0.000 98.81%
Real:0.914 User:0.912 System:0.000 99.83%
Real:0.914 User:0.908 System:0.000 99.33%

linux2$ **for i in {1..8}; do time ./a.out& done**
Real:1.182 User:1.180 System:0.000 99.82%
Real:1.186 User:1.180 System:0.000 99.48%
Real:1.188 User:1.184 System:0.000 99.70%
Real:1.186 User:1.184 System:0.000 99.80%
Real:1.187 User:1.184 System:0.000 99.73%
Real:1.187 User:1.184 System:0.000 99.75%
Real:1.188 User:1.184 System:0.000 99.69%
Real:1.192 User:1.176 System:0.000 98.69%

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 20

10 Prosesser

linux3

Det følgende viser først kompilering og kjøring av det samme 100% CPU-avhengige programmet og så tre forskjellige kjøringer av programmet på den tredje av serverene, linux3. Forklar med utgangspunkt i tiden fra den første kjøringen de tidene som er resultatet av de tre kjøringene. Forklar også prosent-tallene for hver kjøring. Prøv utifra resultatene av kjøringen å finne ut hvor mange cores (med en egen ALU) denne serveren har og om hver core er hyperthreading eller ikke. Forklar kort hvordan du kommer frem til din konklusjon.

linux3$ **gcc -O sum.c**

linux3$ **time ./a.out**
Real:2.282 User:2.280 System:0.000 99.90%

linux3$ **for i in {1..4}; do time ./a.out& done**
Real:2.283 User:2.280 System:0.000 99.85%
Real:2.291 User:2.280 System:0.000 99.53%
Real:2.293 User:2.280 System:0.000 99.41%
Real:2.294 User:2.280 System:0.000 99.41%

linux3$ **for i in {1..6}; do time ./a.out& done**
Real:2.283 User:2.280 System:0.000 99.85%
Real:2.283 User:2.280 System:0.000 99.85%
Real:2.284 User:2.280 System:0.000 99.82%
Real:2.295 User:2.280 System:0.000 99.36%
Real:2.294 User:2.280 System:0.000 99.39%
Real:2.294 User:2.280 System:0.000 99.38%

linux3$ **for i in {1..8}; do time ./a.out& done**
Real:2.284 User:2.280 System:0.000 99.84%
Real:2.284 User:2.280 System:0.000 99.81%
Real:2.284 User:2.280 System:0.000 99.82%
Real:2.283 User:2.280 System:0.000 99.85%
Real:2.286 User:2.280 System:0.000 99.72%
Real:2.289 User:2.280 System:0.000 99.59%
Real:2.287 User:2.280 System:0.000 99.68%
Real:2.294 User:2.280 System:0.000 99.37%

Skriv ditt svar her...

Format | Σ

Words: 0

Maks poeng: 20

11 Docker

Skriv en dockerfile som lager en container med default versjon av ubuntu og med apache2 installert. Og som gjør slik at filen index.html legges i /var/www/html på containeren. Sørg også for at apache startes når containeren startes.

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 10

12 Docker

Skriv ned en docker-kommando som med utgangspunkt i dockerfilen i forrige oppgave bygger et docker-image med navn ubuntuA. Forklar om det spiller noen rolle fra hvilken mappe kommandoen kjøres.

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 10

13 Docker

Skriv ned en docker-kommando som bruker ubuntuA-imaget du bygde i forrige oppgave til å starte i bakgrunnen en container som svarer på web-tilkoblinger på port 5555 og viser siden index.html som ble inkludert i dockerfilen i den første docker-oppgaven. Blir index.html inkludert når image't ubuntuA bygges eller når containeren startes opp? Forklar kort.

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 10

14 Internminne

En byte skal kopieres til et register i CPU. Hvilken av følgende enheter går det raskest å kopiere fra?

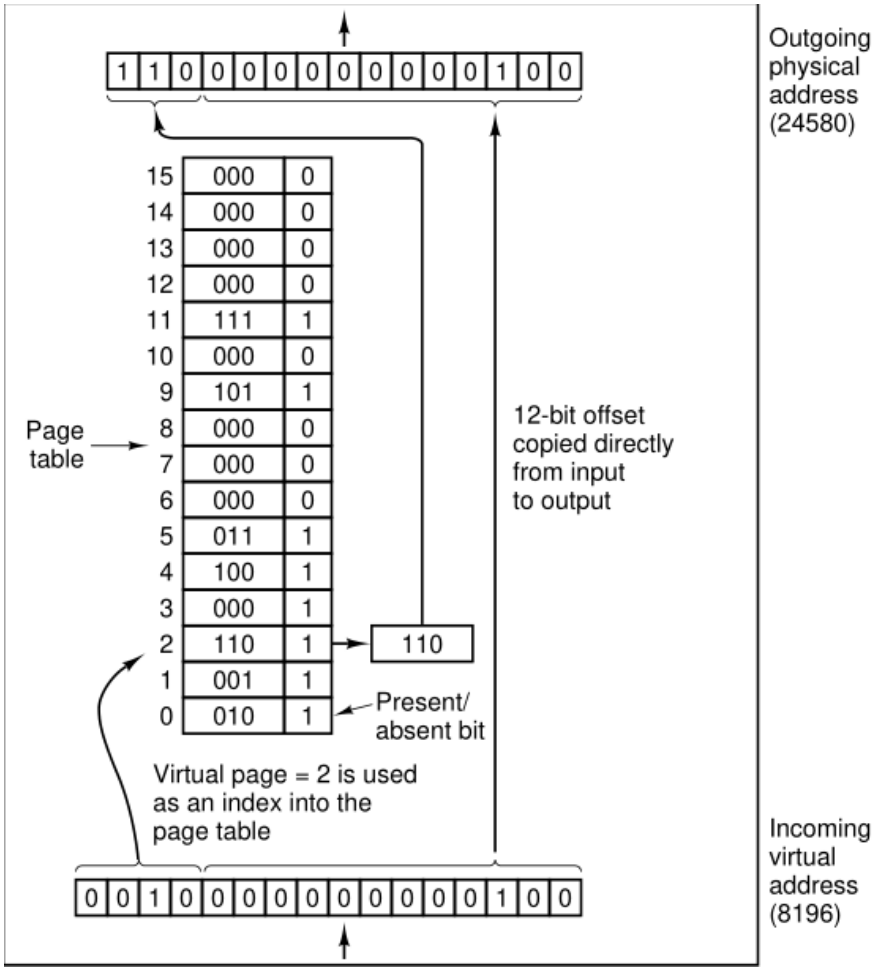
Velg ett alternativ

- L2 cache
- L1 cache
- RAM
- SSD disk
- L3 cache
- HDD disk

Maks poeng: 10

15 MMU

Anta at en innkommende virtuell adresse til MMU er 8. Hva blir da den utgående fysiske adressen?



Velg ett alternativ:

- Gir page fault
- 24588
- 4104
- 8200
- 8204
- 4096
- 24584
- 8
- 24580
- 8196

Maks poeng: 10

16 Linux kommandolinje

Hvilken mappe referer . (ett punktum) til i et bash-shell?

Velg ett alternativ

- Mappen over den du står i
- En skjult fil
- Roten av filsystemet
- Hjemme-mappen din
- Mappen under den du står i
- Mappen du står i
- En skjult mappe

Maks poeng: 10

17 Linux kommandolinje

Hvilken Linux-kommando lager en tom fil med navn fil.txt?

Velg ett alternativ

- make fil.txt
- newitem fil.txt
- chown fil.txt
- chmod fil.txt
- create fil.txt
- touch fil.txt
- cat fil.txt
- mk fil.txt

Maks poeng: 10

18 Linux kommandolinje

Hva slags Linux-filer pleier å ha filendelse **.lin** ?
Velg ett alternativ

- Filer som inneholder informasjon om Linux-konfigurasjonen
- Konfigurasjonsfiler for Linux-kjernen
- Konfigurasjonsfiler som ligger i /etc
- Filer som inneholder informasjon om Linux-distribusjonen
- Dette er ikke en vanlig filendelse i Linux
- Filer som inneholder bash-script

Maks poeng: 10

19 Bash-scripting

I et Linux bash-shell gir følgende kommando (vist i boldface)

linux$ **date -d 01:16:40**
ma. 04. mai 01:16:40 +0200 2020

tidspunktet 16 minutter og 40 sekunder over kl 01 på dagens dato. Videre gir

linux$ **date -d 01:16:40 +%s**
1588547800

antall sekunder som har passert siden midnatt 1. januar 1970.
Tilsvarende gir for et tidspunkt som er ett minutt og 10 sekunder senere:

linux$ **date -d 01:17:50 +%s**
1588547870

som dermed er et tall som er 70 sekunder større enn det forrige.

Skriv et bash-script med navn **time1.sh** som tar et starttidspunkt og et slutttidspunkt på formen TT:MM:SS som argumenter og skriver ut forskjellen i antall sekunder mellom tidspunktene. Du kan anta at slutt er større enn start og trenger ikke å sjekke antall argumenter og om argumentene virkelig er på formen TT:MM:SS. Når scriptet kjører skal det gi resultater som vist nedenfor:

linux$ **./time1.sh 01:16:40 01:16:50**
10
linux$ **./time1.sh 01:16:40 01:18:30**
110
linux$ **./time1.sh 01:18:50 02:18:50**
3600
linux$ **./time1.sh 01:18:50 02:19:00**
3610

Skriv ditt svar her...

```
1
```

Maks poeng: 20

20 Bash-scripting

Når du bruker kommandoen **date** på følgende måte blir dette resultatet:

linux$ **date -d@10 -u +%H:%M:%S**
00:00:10
linux$ **date -d@110 -u +%H:%M:%S**
00:01:50
linux$ **date -d@3610 -u +%H:%M:%S**
01:00:10

Bruk dette til å lage en versjon 2 av scriptet fra forrige oppgave som du kaller **time2.sh** og som gir følgende output når det kjøres:

linux$ **./time2.sh 01:16:40 01:16:50**
00:00:10
linux$ **./time2.sh 01:16:40 01:18:30**
00:01:50
linux$ **./time2.sh 01:18:50 02:18:50**
01:00:00
linux$ **./time2.sh 01:18:50 02:19:00**
01:00:10

Det skal altså gi forskjellen i tid mellom de to tidspunktene på formen TT:MM:SS

Du kan anta at det andre tidspunktet er etter det første og du trenger ikke å sjekke antall argumenter og om argumentene virkelig er på formen TT:MM:SS

Skriv ditt svar her...

```
1
```

Maks poeng: 10

21 Bash-scripting

I denn oppgaven skal du skrive et bash-script **lagDelFilmer.sh** som tar utgangspunkt i en stor mp4 videofil og lager mange mindre delfilmer av den.

Deler av koden du skrev i de forrige to oppgavene kan brukes i denne oppgaven. Problemstillingen er at du har en lang mp4-video som skal deles inn i flere mindre mp4-videoer. Du kan tenke deg at du allerede har laget et script som du har kalt **fromTo.sh** som bruker videoediteringsprogrammet **ffmpeg** fra kommandolinjen til å trekke ut og lagre en vilkårlig del av en videoen. Programmet **ffmpeg** forventer å få spesifisert hvilken del av videoen som skal trekkes ut ved hjelp av to tidsangivelser på formen TT:MM:SS som i de to forrige oppgavene. Derfor har du laget **fromTo.sh** som skjuler noen av detaljene og som virker på følgende måte:

linux$ **fromTo.sh inn.mp4 ut.mp4 00:02:34 00:01:13**

Dette scriptet forventer at **inn.mp4** er en lang video som det skal tas et utdrag fra, **ut.mp4** er videoen som lages, **00:02:34** er tidspunktet i den originale videoen **inn.mp4** der **ut.mp4** skal starte og **00:01:13** er lengden på den resulterende videoen **ut.mp4.**

Når du editerer videoen, ser du på den og skriver ned start- og stopp-tidspunkt for hver mindre delfilm som skal lages. Du kunne kjørt **fromTo.sh** for hver eneste del-video, men siden du ofte lager 12-15 videoer ønsker du å automatisere denne jobben. Derfor skriver du et bash-script **lagDelFilmer.sh** som du eksempelvis kan kjøre på denne måten:

linux$ **lagDelFilmer.sh inn.mp4 0:00:04 0:08:05 0:08:25 0:10:26 0:11:03 0:21:55**

I dette eksempelet skal da scriptet med utgangspunkt i filen **inn.mp4** lage tre del-filmer (siden det er 3 par av tidsangivelser) ved at det kjører scriptet **fromTo.sh** tre ganger som uavhengige prosessoer med følgende kommandoer:

linux$ **./fromTo.sh inn.mp4 inn1.mp4 0:00:04 00:08:01**
linux$ **./fromTo.sh inn.mp4 inn2.mp4 0:08:25 00:02:01**
linux$ **./fromTo.sh inn.mp4 inn3.mp4 0:11:03 00:10:52**

og dermed produsere tre videoer **inn1.mp4**, **inn2.mp4** og **inn3.mp4**. Legg merke til at det siste argumentet som gis til **fromTo.sh** må være lengden på videoen og ikke tidspunktet i originalfilmen der klippet slutter. Som et eksempel så starter det første filmklippet angitt ved 0:00:04 0:08:05 etter 4 sekunder i den originale videoen **inn.mp4** og avslutter etter 8 minutter og 5 sekunder. Det ønskede klippet er dermed 8 minutter og ett sekund langt og det må angis med det siste argumentet til **fromTo.sh**, dermed må det være 00:08:01. For å få til dette kan du bruke bash-koden du skrev i de to forrige oppgavene.

Skriv scriptet **lagDelFilmer.sh** og skriv det slik at det kan lage alt fra en til vilkårlig mange del-filmer, angitt ved par av tidsangivelser med starttidspunkt og lengde. Scriptet skal også kunne ta andre video-navn enn inn.mp4 som argument.

Et par hint som kan være nyttige:

Kommandoen **shift** leser inn neste argument i variabelen $1

Følgende kommando kan brukes til å fjerne endelsen på et filnavn:

$ **basename -s .mp4 inn.mp4**
inn

Skriv ditt svar her...

```
1
```

Maks poeng: 30

22 PowerShell

Vi har i bash-script oppgaven sett at i et Linux bash-shell gir følgende kommando (vist i boldface)

linux$ **date -d 01:16:40**
ma. 04. mai 01:16:40 +0200 2020

En tilsvarende PowerShell-kommando gir:

PS C:\Users\os> **Get-Date 01:16:40**
mandag 4. mai 2020 01.16.40

Det er noen forskjeller i hvordan informasjonen skrives ut, men forklar hvilken fundamental forskjell det er på resultatet man får fra en PowerShell-kommando sammenlignet med det man får fra en Linux-kommando. (En forskjell som også finnes for andre kommandoer enn Get-Date.)

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 10

23 PowerShell

I denne oppgaven skal du skrive et PowerShell-script **time2.ps1** som virker på samme måte som bash-scriptet **time2.sh** fra bash-script oppgaven. Det vil si at scriptet skal virke som følger:

```
PS C:\Users\os> .\time2.ps1 00:02:45 00:04:50
00:02:05
PS C:\Users\os> .\time2.ps1 01:02:45 14:04:50
13:02:05
PS C:\Users\os> .\time2.ps1 01:02:45 14:04:40
13:01:55
```

Scriptet skal altså returnere forskjellen mellom andre og første tidspunkt på den angitte formen. Du kan anta at det andre tidspunktet er etter det første og du trenger ikke å sjekke antall argumenter og om argumentene virkelig er på formen TT:MM:SS

Du gjør først litt research i PowerShell og kommer frem til følgende resultater som kan være nyttige for at scriptet skal fungere korrekt (kommandoer uthevet):

```
PS C:\Users\os> $start = Get-Date 00:03:45
PS C:\Users\os> $stop  = Get-Date 00:04:55
PS C:\Users\os> $stop - $start

Days              : 0
Hours             : 0
Minutes           : 1
Seconds           : 10
Milliseconds      : 0
Ticks             : 700000000
TotalDays         : 0,000810185185185185
TotalHours        : 0,0194444444444444
TotalMinutes      : 1,16666666666667
TotalSeconds      : 70
TotalMilliseconds : 70000

PS C:\Users\os> '{0:d2}' -f 0
00
PS C:\Users\os> '{0:d2}' -f 2
02
PS C:\Users\os> '{0:d2}' -f 02
02
PS C:\Users\os> '{0:d2}' -f 22
22

PS C:\Users\os> $s = "name"
PS C:\Users\os> "$s:tekst"
PS C:\Users\os> "${s}:tekst"
name:tekst
```

Skriv ditt svar her...

```
1
```

Maks poeng: 20