

TEKSTFILER OG CSV

INF100

HØST 2025

Torstein Strømme

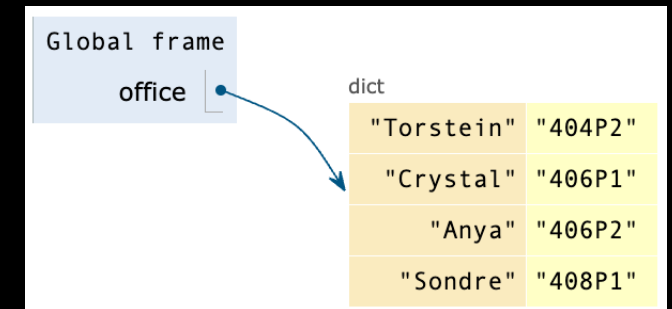
OPPSLAGSVERK

OPPSLAGSVERK

```
# Et oppslagsverk er som en telefonkatalog:  
# man kan slå opp på en nøkkel, og få vite dens verdi
```

```
office = {  
    'Torstein': '404P2',  
    'Crystal': '406P1',  
    'Anya': '406P2',  
    'Sondre': '408P1',  
}
```

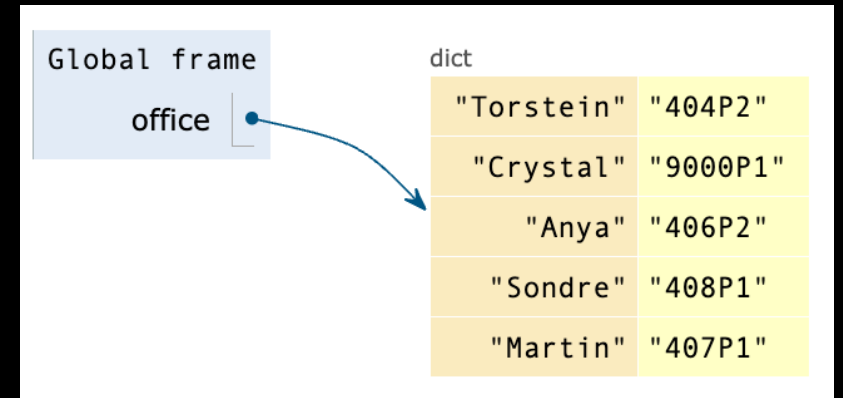
```
print('Torstein sitt kontor er', office['Torstein'])
```



OPPSLAGSVERK

```
office = {  
    'Torstein': '404P2',  
    'Crystal': '406P1',  
    'Anya': '406P2',  
    'Sondre': '408P1',  
}
```

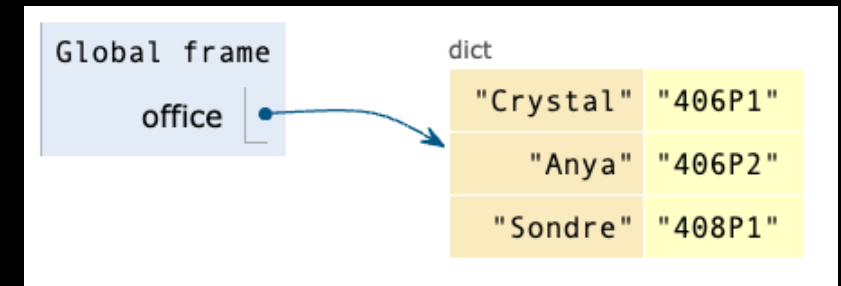
```
office['Martin'] = '407P1' # Legg til ny nøkkel/verdi  
office['Crystal'] = '9000P1' # Crystal bytter kontor
```



OPPSLAGSVERK

```
office = {  
    'Torstein': '404P2',  
    'Crystal': '406P1',  
    'Anya': '406P2',  
    'Sondre': '408P1',  
}
```

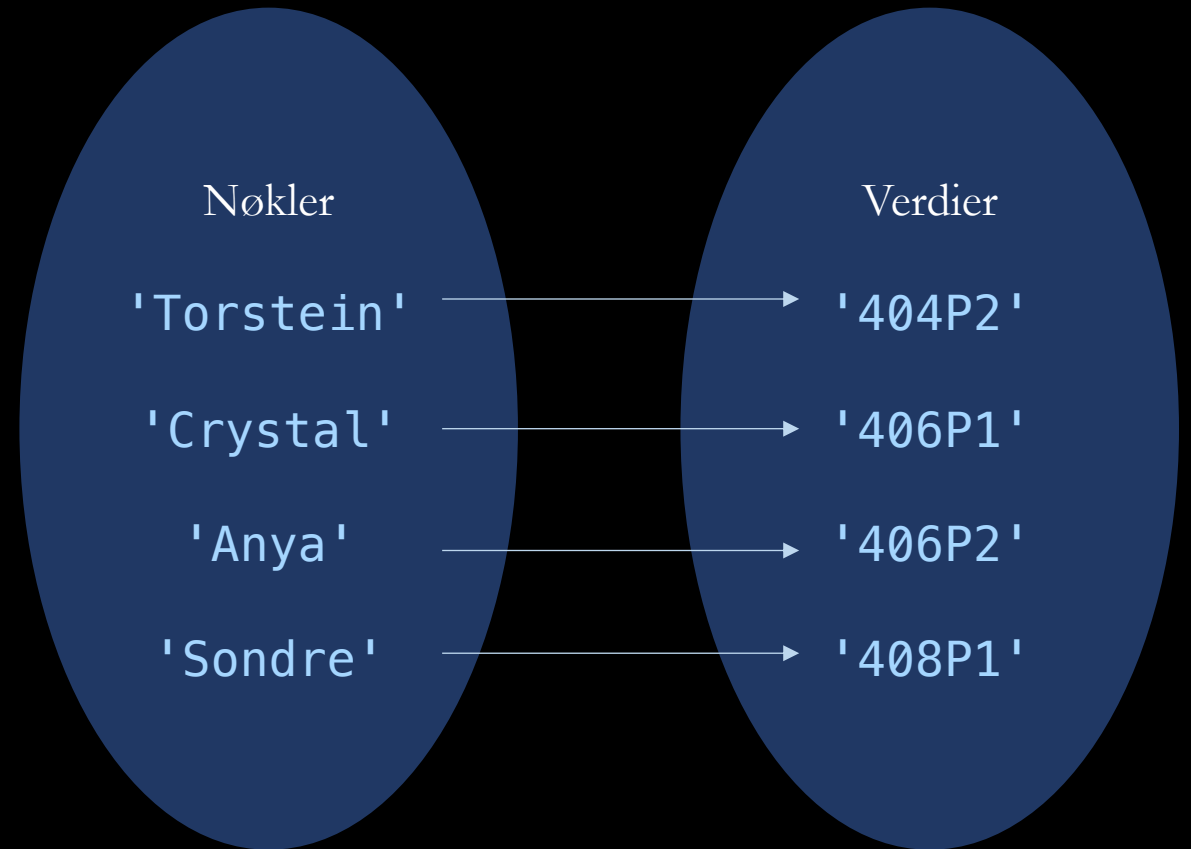
```
office.pop('Torstein') # Fjern nøkkel
```



OPPSLAGSVERK

```
office = {  
    'Torstein': '404P2',  
    'Crystal': '406P1',  
    'Anya': '406P2',  
    'Sondre': '408P1',  
}
```

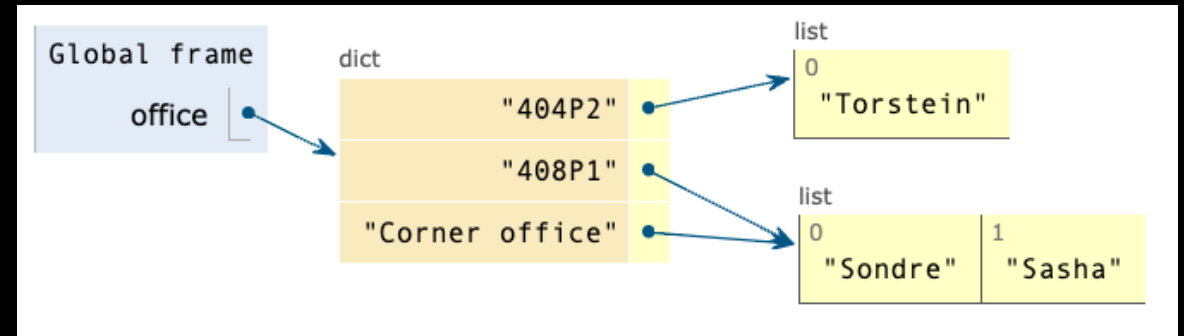
office



OPPSLAGSVERK

```
office = {  
    '404P2': ['Torstein'],  
    '408P1': ['Sondre', 'Sasha'],  
}
```

```
office['Corner office'] = office['408P1'] # alias
```



OPPSLAGSVERK vs FUNKSJON

Oppslagsverk:

- Kan utvides med nye nøkler/verdier dynamisk
- Kan endre verdi til eksisterende nøkler
- Raskere

```
office = {  
    'Torstein': '404P2',  
    'Crystal': '406P1',  
    'Anya': '406P2',  
    'Sondre': '408P1',  
}
```

```
print('Torsteins kontor:',  
      office['Torstein'])
```

```
def office(key):  
    if key == 'Torstein': return '404P2'  
    elif key == 'Anya': return '406P2'  
    elif key == 'Crystal': return '406P1'  
    elif key == 'Sondre': return '408P1'
```

```
print('Torsteins kontor:', office('Torstein'))
```


OPPSLAGSVERK vs LISTE MED TUPLER

Oppslagsverk:

- Maksimum én verdi per nøkkel
- Rekkefølge betyr ingenting
- Raskere

```
office = {  
    'Torstein': '404P2',  
    'Crystal': '406P1',  
    'Anya': '406P2',  
    'Sondre': '408P1',  
}
```

```
print('Torsteins kontor:',  
      office['Torstein'])
```

```
office = [  
    ('Torstein', '404P2'),  
    ('Anya', '406P2'),  
    ('Crystal', '408P1'),  
    ('Sondre', '408P1'),  
]
```

```
def get_key(key, li):  
    for k, value in li:  
        if k == key:  
            return value
```

```
print('Torsteins kontor:',  
      get_key('Torstein', office))
```

OPPSLAGSVERK vs VARIABLER

Oppslagsverk:

- Ingen “forbudte ord”
- Kan bruke andre typer som nøkler (int/float/bool etc.)

```
office = {  
    'Torstein': '404P2',  
    'Crystal': '406P1',  
    'Anya': '406P2',  
    'Sondre': '408P1',  
}
```

```
print('Torsteins kontor:',  
      office['Torstein'])
```

```
Torstein = '404P2'  
Anya = '406P2'  
Crystal = '408P1'  
Sondre = '408P1'
```

```
print('Torsteins kontor:', Torstein)
```

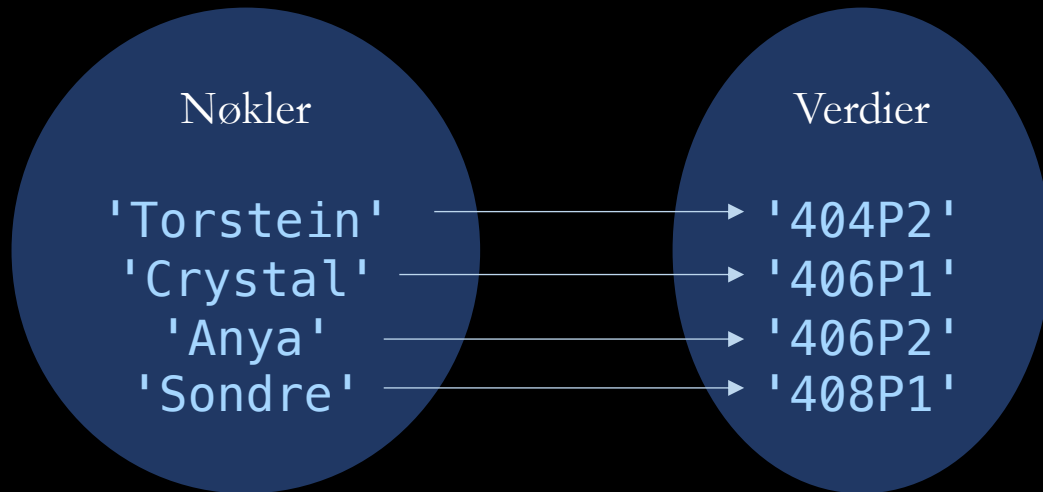
Variabler ER faktisk nøkler i et oppslagsverk!

OPPSLAGSVERK

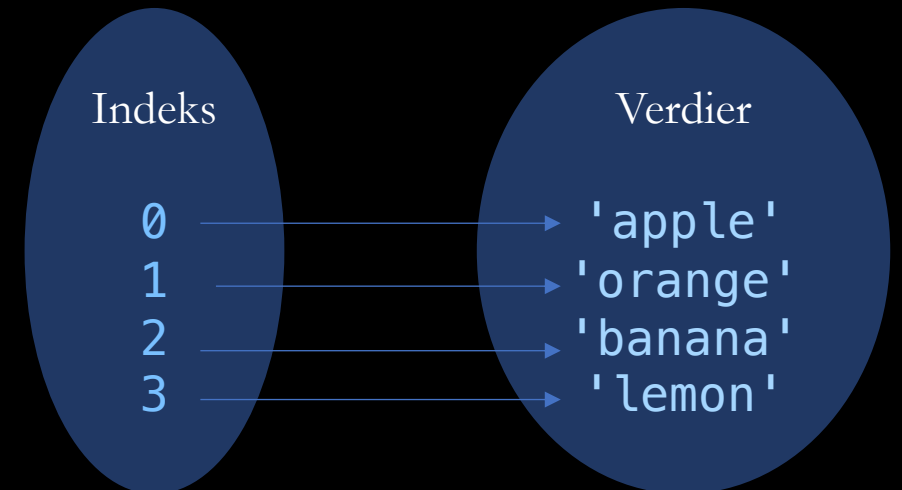
- En samling med “variabler”
- Kan bruke ulike typer som nøkler
 - int, float, str, bool, etc.
 - (men ikke noe som muterbart)
- Oppslagsverk er muterbare

OPPSLAGSVERK vs LISTER

```
office = {  
    'Torstein': '404P2',  
    'Crystal': '406P1',  
    'Anya': '406P2',  
    'Sondre': '408P1',  
}
```



```
fruits = [  
    'apple',  
    'orange',  
    'banana',  
    'lemon',  
]
```



FINN VANLIGSTE BOKSTAV

- Gitt en tekstfil: skriv ut hvilke bokstaver den inneholder og hvor mange av hver bokstav

```
nsf2022.txt x
05_lister > nsf2022.txt
1 a
2 ab
3 abaca
4 abacaen
5 abacaene
6 abacaer
7 abaki
8 abakien
9 abakiene
10 abakier
11 abakus
12 abakusen
13 abakusene
14 abakuser
15 abalienasjon
16 abalienasjonen
17 abalienasjonene
```



```
'a' -> 619290
'\n' -> 879902
'b' -> 180822
'c' -> 12238
'e' -> 1587179
'n' -> 934403
'r' -> 836100
'k' -> 471437
...
```

CSV

Comma separated values

- En CSV-fil: ren tekst som representerer tabell-data

```
personnummer,medisin,startdato  
01010111111,kake,2022-08-22  
22020222222,fluor,2022-01-11  
01010111111,tran,2022-06-01
```



	A	B	C
1	personnummer	medisin	startdato
2	10101111111	vitaminbjørner	22/08/2022
3	22020222222	fluortabletter	11/01/2022
4	10101111111	tran	01/06/2022

- En CSV-fil: ren tekst som representerer tabell-data

```
personnummer,medisin,startdato  
01010111111,kake,2022-08-22  
22020222222,fluor,2022-01-11  
01010111111,tran,2022-06-01
```

- I Python: tabell-data representert som streng

```
table_string = '''  
personnummer,medisin,startdato  
01010111111,kake,2022-08-22  
22020222222,fluor,2022-01-11  
01010111111,tran,2022-06-01  
'''
```


- En CSV-fil: ren tekst som representerer tabell-data

```
personnummer,medisin,startdato  
01010111111,kake,2022-08-22  
22020222222,fluor,2022-01-11  
01010111111,tran,2022-06-01
```

- I Python: tabell-data representert som liste av strenger

```
table_string_list = [  
    'personnummer,medisin,startdato',  
    '01010111111,kake,2022-08-22',  
    '22020222222,fluor,2022-01-11',  
    '01010111111,tran,2022-06-01',  
]
```

- En CSV-fil: ren tekst som representerer tabell-data

```
personnummer,medisin,startdato  
01010111111,kake,2022-08-22  
22020222222,fluor,2022-01-11  
01010111111,tran,2022-06-01
```

- I Python: tabell-data representert som 2D-liste

```
table_2d_list = [  
    ['personnummer', 'medisin', 'startdato'],  
    ['01010111111', 'kake', '2022-08-22'],  
    ['22020222222', 'fluor', '2022-01-11'],  
    ['01010111111', 'tran', '2022-06-01'],  
]
```

- En CSV-fil: ren tekst som representerer tabell-data

```
personnummer,medisin,startdato  
01010111111,kake,2022-08-22  
22020222222,fluor,2022-01-11  
01010111111,tran,2022-06-01
```

- I Python: tabell-data representert som liste av oppslagsverk

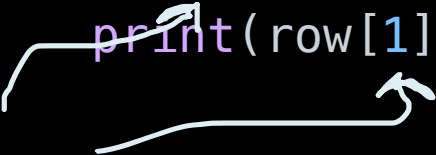
```
table_dict_list = [  
    {'personnummer': '01010111111', 'medisin': 'kake', 'startdato': '2022-08-22'},  
    {'personnummer': '22020222222', 'medisin': 'fluor', 'startdato': '2022-01-11'},  
    {'personnummer': '01010111111', 'medisin': 'tran', 'startdato': '2022-06-01'}  
]
```

- Liste av lister: hvilke medisiner er registrert på person 01010111111?

```
table_2d_list = [  
    ['personnummer', 'medisin', 'startdato'],  
    ['01010111111', 'kake', '2022-08-22'],  
    ['22020222222', 'fluor', '2022-01-11'],  
    ['01010111111', 'tran', '2022-06-01'],  
]
```

```
for row in table_2d_list:  
    if row[0] == '01010111111':  
        print(row[1])
```

magiske tall (fy!)



- Liste av oppslagsverk: hvilke medisiner er registrert på person 01010111111?

```
table_dict_list = [  
    {'personnummer': '01010111111', 'medisin': 'kake', 'startdato': '2022-08-22'},  
    {'personnummer': '22020222222', 'medisin': 'fluor', 'startdato': '2022-01-11'},  
    {'personnummer': '01010111111', 'medisin': 'tran', 'startdato': '2022-06-01'}  
]
```

```
for row in table_dict_list:  
    if row['personnummer'] == '01010111111':  
        print(row['medisin'])
```



FRA CSV TIL LISTE AV OPPSLAGSVERK

- Fra CSV til streng

```
from pathlib import Path
table_string = Path('org.csv').read_text()
```

```
table_string = '''
personnummer,medisin,startdato
01010111111,kake,2022-08-22
22020222222,fluor,2022-01-11
01010111111,tran,2022-06-01
'''
```

- Fra CSV til streng
- **Fra streng til liste av linjer**
 - Øverste linje er header
 - Øvrige linjer er data

```
from pathlib import Path
table_string = Path('org.csv').read_text()
```

```
lines = table_string.splitlines()
header_line = lines[0]
data_lines = lines[1:]
```

```
lines = [
    'personnummer,medisin,startdato',
    '01010111111,kake,2022-08-22',
    '22020222222,fluor,2022-01-11',
    '01010111111,tran,2022-06-01',
]
```

```
data_lines = [
    '01010111111,kake,2022-08-22',
    '22020222222,fluor,2022-01-11',
    '01010111111,tran,2022-06-01',
]
```

```
header_line = 'personnummer,medisin,startdato'
```


- Fra CSV til streng
- **Fra streng til liste av linjer**
 - Øverste linje er header
 - Øvrige linjer er data

```
from pathlib import Path
table_string = Path('org.csv').read_text()

lines = table_string.splitlines()
header_line = lines[0]
data_lines = lines[1:]
headers = header_line.split(',')
```

```
lines = [
    'personnummer,medisin,startdato',
    '01010111111,kake,2022-08-22',
    '22020222222,fluor,2022-01-11',
    '01010111111,tran,2022-06-01',
]
```

```
data_lines = [
    '01010111111,kake,2022-08-22',
    '22020222222,fluor,2022-01-11',
    '01010111111,tran,2022-06-01',
]
```

```
header_line = 'personnummer,medisin,startdato'
```

```
headers = ['personnummer', 'medisin', 'startdato']
```

- Fra CSV til streng
- Fra streng til liste av linjer
 - Øverste linje er header
 - Øvrige linjer er data
- Fyll tabell med rader

```
table_dict = [  
    {},  
    {},  
    {},  
]
```

```
from pathlib import Path  
table_string = Path('org.csv').read_text()  
  
lines = table_string.splitlines()  
header_line = lines[0]  
data_lines = lines[1:]  
headers = header_line.split(',')  
  
table_dict = []  
for line in data_lines:  
    row_dict = {}  
    # begynn fyll opp row_dict her ...  
  
    # ... ferdig fylt opp row_dict her  
    table_dict.append(row_dict)
```

- Fra CSV til streng
- Fra streng til liste av linjer
 - Øverste linje er header
 - Øvrige linjer er data
- Fyll tabell med rader
- **Fra linje til oppslagslagsverk**

```
line = '01010111111,kake,2022-08-22'
```

```
cells = ['01010111111', 'kake', '2022-08-22']
```

```
table_dict = []
for line in data_lines:
    row_dict = {}
    # begynn fyll opp row_dict her ...

    cells = line.split(',')
    for i in range(number_of_cols):
        header = headers[i]
        value = cells[i]
        row_dict[header] = value

    # ... ferdig fylt opp row_dict her
    table_dict.append(row_dict)
```

- Fra CSV til streng
- Fra streng til liste av linjer
 - Øverste linje er header
 - Øvrige linjer er data
- Fyll tabell med rader
- **Fra linje til oppslagslagsverk**

```
number_of_cols = 3
```

```
i = 0
```

```
header = 'personnummer'
```

```
value = '01010111111'
```

```
number_of_cols = len(headers)
```

```
table_dict = []
```

```
for line in data_lines:
```

```
    row_dict = {}
```

```
    # begynn fyll opp row_dict her ...
```

```
    cells = line.split(',')
```

```
    for i in range(number_of_cols):
```

```
        header = headers[i]
```

```
        value = cells[i]
```

```
        row_dict[header] = value
```

```
    # ... ferdig fylt opp row_dict her
```

```
    table_dict.append(row_dict)
```

```
row_dict['personnummer'] = '01010111111'
```

- Fra CSV til streng
- Fra streng til liste av linjer
 - Øverste linje er header
 - Øvrige linjer er data
- Fyll tabell med rader
- **Fra linje til oppslagslagsverk**

```
number_of_cols = 3
```

```
i = 1
```

```
header = 'medisin'
```

```
value = 'kake'
```

```
number_of_cols = len(headers)
```

```
table_dict = []
```

```
for line in data_lines:
```

```
    row_dict = {}
```

```
    # begynn fyll opp row_dict her ...
```

```
    cells = line.split(',')
```

```
    for i in range(number_of_cols):
```

```
        header = headers[i]
```

```
        value = cells[i]
```

```
        row_dict[header] = value
```

```
    # ... ferdig fylt opp row_dict her
```

```
    table_dict.append(row_dict)
```

```
row_dict['medisin'] = 'kake'
```

- Fra CSV til streng
- Fra streng til liste av linjer
 - Øverste linje er header
 - Øvrige linjer er data
- Fyll tabell med rader
- **Fra linje til oppslagslagsverk**

```
number_of_cols = 3
```

```
i = 2
```

```
header = 'startdato'
```

```
value = '2022-08-22'
```

```
number_of_cols = len(headers)
```

```
table_dict = []
```

```
for line in data_lines:
```

```
    row_dict = {}
```

```
    # begynn fyll opp row_dict her ...
```

```
    cells = line.split(',')
```

```
    for i in range(number_of_cols):
```

```
        header = headers[i]
```

```
        value = cells[i]
```

```
        row_dict[header] = value
```

```
    # ... ferdig fylt opp row_dict her
```

```
    table_dict.append(row_dict)
```

```
row_dict['startdato '] = '2022-08-22 '
```

- Fra CSV til streng
- Fra streng til liste av linjer
 - Øverste linje er header
 - Øvrige linjer er data
- Fyll tabell med rader
- **Fra linje til oppslagslagsverk**

```
number_of_cols = len(headers)

table_dict = []
for line in data_lines:
    row_dict = {}
    # begynn fyll opp row_dict her ...

    cells = line.split(',')
    for i in range(number_of_cols):
        header = headers[i]
        value = cells[i]
        row_dict[header] = value

    # ... ferdig fylt opp row_dict her
    table_dict.append(row_dict)
```

```
row_dict = {'personnummer': '01010111111', 'medisin ': 'kake', 'startdato': '2022-08-22'}
```

- Fra CSV til streng
- Fra streng til liste av linjer
 - Øverste linje er header
 - Øvrige linjer er data
- Fyll tabell med rader
- **Fra linje til oppslagslagsverk**

```
number_of_cols = len(headers)

table_dict = []
for line in data_lines:
    row_dict = {}
    # begynn fyll opp row_dict her ...

    cells = line.split(',')
    for i in range(number_of_cols):
        header = headers[i]
        value = cells[i]
        row_dict[header] = value

    # ... ferdig fylt opp row_dict her
    table_dict.append(row_dict)
```

```
table_dict_list = [
    {'personnummer ': '01010111111', 'medisin': 'kake', 'startdato': '2022-08-22'},
    {'personnummer ': '22020222222', 'medisin': 'fluor', 'startdato': '2022-01-11'},
    {'personnummer ': '01010111111', 'medisin': 'tran', 'startdato': '2022-06-01'},
]
```



```
from pathlib import Path
```

```
table_as_csv = Path('filename.csv').read_text(encoding='utf-8')
```

```
lines = table_as_csv.splitlines()
```

```
headers = lines[0].split(',')
```

```
data_lines = lines[1:]
```

```
number_of_cols = len(headers)
```

```
table_dict = []
```

```
for line in data_lines:
```

```
    row_dict = {}
```

```
    cells = line.split(',')
```

```
    for i in range(len(headers)):
```

```
        header = headers[i]
```

```
        value = cells[i]
```

```
        row_dict[header] = value
```

```
    table_dict.append(row_dict)
```

NOEN MÅ HA GJORT DETTE FØR

Alternatív A

```
from pathlib import Path  
from csv import DictReader
```

```
with Path('filename.csv').open('rt', encoding='utf-8', newline='') as file:  
    reader = DictReader(file, delimiter=',', quotechar='\"')  
  
    headers = reader.fieldnames  
    data = list(reader)
```

Alternativ B

```
from pathlib import Path
from csv import DictReader
from io import StringIO

file_string = Path('filename.csv').read_text(encoding='utf-8')
reader = DictReader(StringIO(file_string), delimiter=',', quotechar='\"')

headers = reader.fieldnames
data = list(reader)
```

Fungerer også hvis du har en CSV-streng som ikke kommer fra en fil
(men er litt langsommere med store filer)

Skríve til streng

```
from pathlib import Path
from csv import DictWriter
from io import StringIO
```

```
headers = ...
data = ...
```

```
output = StringIO()
writer = DictWriter(output, fieldnames=headers, delimiter=',', quotechar='"')
writer.writeheader()
writer.writerows(data)
```

```
output_string = output.getvalue()
output.close()
```

Skríve til fil

```
from pathlib import Path
from csv import DictWriter
```

```
headers = ...
data = ...
```

```
with Path('newfile.csv').open('wt', encoding='utf-8', newline='') as file:
    writer = DictWriter(file, fieldnames=headers, delimiter=',', quotechar='"')
    writer.writeheader()
    writer.writerows(data)
```

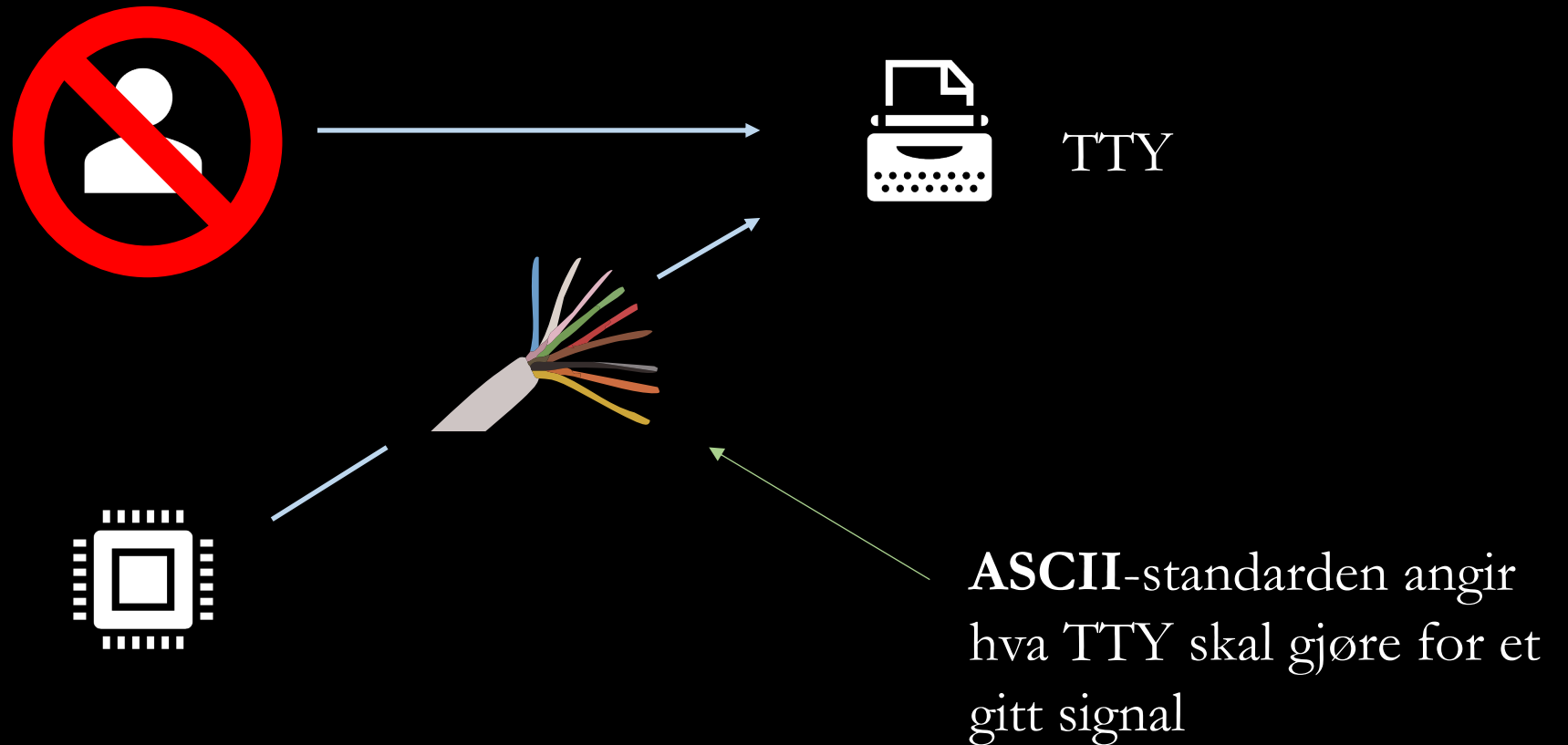
TEKSTFILER

HISTORIETIME: EN COMPUTER



1954, NACA

HISTORIETIME: EN COMPUTER



ASCII



BEL: Ring med bjellen

LF: line feed (flytt papiret én rad ned)

CR: carriage return (flytt skrivehodet tilbake til starten av arket)

SP: Space (flytt skrivehodet én plass frem)

ASCII (1977/1986)																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1x	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2x	SP	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

BS: Backspace (flytt skrivehodet én plass tilbake)

DEL: Slett forrige tegn

TEKSTFIL

- Mange filformater er basert på «ren tekst» (innholdet i filen er bare en streng)
- .txt
- .py/.js/.java/.cpp/.bat/.sh
- .csv/.json/.xml/.yaml/
- .html/.css/.tex/.md
- .svg
- ...

Andre filformater er *ikke* basert på ren tekst:

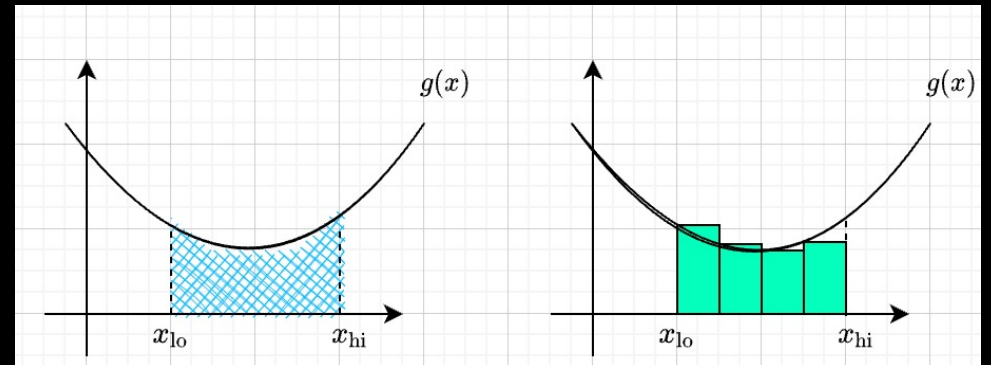
- .docx/.rtf/.pdf
- .xlsx
- .zip
- .mp3/.acc/.mp4
- .png/.jpeg/.gif

TEKSTFIL

- Mange filformater er basert på «ren tekst»
- Filer basert på ren tekst kan håndteres i enhver teksteditor (VSCode, Notepad, TextEdit, vim etc)

area_under_graph.svg

```
...<path d="M 10.5 190.5 L 255.95 190.5" fill="none" stroke="rgb(0, 0, 0)" stroke-width="1.5" stroke-miterlimit="10" pointer-events="stroke"/><path d="M 263.82 190.5 L 253.32 195.75 L 255.95 190.5 L 253.32 185.25 Z" fill="rgb(0, 0, 0)" stroke="rgb(0, 0, 0)" stroke-width="1.5" stroke-miterlimit="10" pointer-events="all"/><path d="M 40.5 220.5 L 40.5 20.05" fill="none" stroke="rgb(0, 0, 0)" stroke-width="1.5" stroke-miterlimit="10" pointer-events="stroke"/>...
```



TEKSTFIL

- Mange filformater er basert på «ren tekst»
- Filer basert på ren tekst kan håndteres i enhver teksteditor (VSCode, Notepad, TextEdit, vim etc)

```
âPNG<CR>
^Z
<NUL><NUL><NUL><CR>IHDR<NUL><NUL>^Ah<NUL><NUL>^Ah<BS>^F<NUL><NUL><NUL>zÂa'<NUL><NU
<NP>A^Pö^Bë»ÅÏ(V<RS><FS>
·[éá@ÆÄ~A~ÈÓ^W<ESC>AÚ^Vdx2ÙêfFî^GA†É^R1æGêa(<BS>c) -+Ú%ÔÏ~<NP>50{Ô/µw^Eø1Øj^QAdM$
◆}Î[_...nøñæi^O»^Y~ÿ/|0af-∞Î-Âvç0+ä-^Z≠ñm,,%Àa^TZö$ññSÛ<æNæÆ<RS>^æxiª<Σi`€mf0zeií/†?
Å^EAp±ðæmp2¥dì
>ö<US>^D^0iMlÅ
ÍV≠Vm>öÈ~öç¶e
>n~Àq°yðeôUkUÅ
ΣÈtj^•bifµ~>xfi<
tmQã0{^K-Ô,ß,]zæ_,Ï°^00úúâ^QæÄã/üüü?ººº^R3mw:~V<US>òm<FS>vp~vrr"∞É"Êª^B^lÎ1FÄ...Ô"~V
^AR@~ÜÜ'ÆØØ≠ÿl<^A&N'mÂÂ"~VÜÖ'Í5^A?0fi<US><NP>l_~T6^Yçu,ö1F†/l5^EÆæÁ['>≤ÈxlQµ*9d
µ*%æ,¿0$öç$^U^V^VéªZØÈ±0qæ&,...^E0µT/?ÿ%;°1èøy^K/fl•kÓ]{ö[Ä~0>0?«°"Í^Wøª^XÄ≠0<ESC>_é
```

.png er ikke basert på tekst,
og kan ikke egentlig åpnes i teksteditor

python-circle.png



TEKSTFIL

- Mange filformater er basert på «ren tekst»
- Filer basert på ren tekst kan håndteres i enhver teksteditor (VSCode, Notepad, TextEdit, vim etc)
- Det er dessverre forskjell på rene tekst-filer også ☹
- Forskjellen går på tvers av fil-endelse, og handler om valg av *koding*
- Koding: regler for hvordan skal 0'ere og 1'ere i datamaskinen tolkes som skriftsymboler

ASCII

- 127 ulike tegn
- Gammel standard
- Det eneste som alltid fungerer
- Mangler æøåéü£†f💩

Binary ^	Oct ⇅	Dec ⇅	Hex	Glyph				
				1963 ⇅	1965 ⇅	1967 ⇅		
010 0000	040	32	20	space				
010 0001	041	33	21	!				
010 0010	042	34	22	"				
010 0011	043	100 0001	101	65	41	A		
010 0100	044	100 0010	102	66	42	B		
010 0101	045	100 0011	103	67	43	C		
010 0110	046	100 0100	104	68	44	D		
010 0111	047	100 0101	105	110 0001	141	97	61	a
010 1000	050	100 0110	106	110 0010	142	98	62	b
010 1001	051	100 0111	107	110 0011	143	99	63	c
010 1010	052	100 1000	108	110 0100	144	100	64	d
010 1011	053	100 1001	109	110 0101	145	101	65	e
010 1100	054	100 1010	110	110 0110	146	102	66	f
		100 1011	111	110 0111	147	103	67	g
		100 1100	112	110 1000	150	104	68	h
				110 1001	151	105	69	i
				110 1010	152	106	6A	j
				110 1011	153	107	6B	k
				110 1100	154	108	6C	l

https://en.wikipedia.org/wiki/ASCII#Printable_characters

UTF-8

- Alle unicode –symboler
(144697 ulike tegn akkurat nå)
- Bakoverkompatibel med ASCII

- Unicode-verdien til et tegn:

`ord('A')`

- Tegnet til en gitt unicode-verdi:

`chr(105)`

Binary ▲	Oct ◆	Dec ◆	Hex	Glyph					
				1963 ◆	1965 ◆	1967 ◆			
010 0000	040	32	20	space					
010 0001	041	33	21	!					
010 0010	042	34	22	"					
010 0011	043	100 0001		101	65	41	A		
010 0100	044	100 0010		102	66	42	B		
010 0101	045	100 0011		103	67	43	C		
010 0110	046	100 0100		104	68	44	D		
010 0111	047	100 0101		105	110 0001	141	97	61	a
010 1000	050	100 0110		106	110 0010	142	98	62	b
010 1001	051	100 0111		107	110 0011	143	99	63	c
010 1010	052	100 1000		108	110 0100	144	100	64	d
010 1011	053	100 1001		109	110 0101	145	101	65	e
010 1100	054	100 1010		110	110 0110	146	102	66	f
		100 1011		111	110 0111	147	103	67	g
		100 1100		112	110 1000	150	104	68	h
					110 1001	151	105	69	i
					110 1010	152	106	6A	j
					110 1011	153	107	6B	k
					110 1100	154	108	6C	l

DATA SOM REN TEKST

CSV

Tabelldata ($\leq 2D$)

```
personnummer,medisin,startdato,sluttdato  
01010111111,vitaminbjørner,2022-08-22,2022-11-28  
22020222222,fluortabletter,2022-01-11,2022-03-03  
01010111111,tran,2022-06-01,2022-06-30
```

JSON

Svært fleksibel

```
{  
  "data": [  
    {  
      "id": "8fa94033", "type": "fact", "attributes":  
        {  
          "body": "Dogs have lived with humans for 14,000 years."  
        }  
    },  
    {  
      "id": "c898f9c5", "type": "fact", "attributes":  
        {  
          "body": "More than one in three U.S. families owns a dog."  
        }  
    },  
    {  
      "id": "c6fe1512", "type": "fact", "attributes":  
        {  
          "body": "77 ancient Egyptian dog names are known."  
        }  
    }  
  ]  
}
```

RECAP: STRINGER ↔ LISTER

```
s = 'Marshall,Rubble,Chase,Rocky,Zuma,Sky'  
a = s.split(',')  
  
print(a)
```

```
['Marshall', 'Rubble', 'Chase', 'Rocky', 'Zuma', 'Sky']
```

RECAP: STRONGER ↔ LISTER

```
a = ['Marshall', 'Rubble', 'Chase', 'Rocky', 'Zuma', 'Sky']  
s = ';'.join(a)  
  
print(s)
```

Marshall;Rubble;Chase;Rocky;Zuma;Sky

RECAP: STRENGER ↔ LISTER

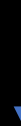
```
s = '''\nMarshall,Rubble,Chase\nRocky,Zuma,Sky\n'''\na = s.splitlines()\n\nprint(a)
```

```
['Marshall,Rubble,Chase', 'Rocky,Zuma,Sky']
```

REFORMATTERING AV CSV

- Du har en CSV-fil med data, men det noen steder mangler informasjon
- Produser en CSV-fil hvor manglende data får en gitt verdi

```
personnummer,medisin,startdato,sluttdato  
01010111111,vitaminbjørner,2022-08-22,2022-11-28  
22020222222,fluortabletter,2022-01-11,  
01010111111,tran,2022-06-01,2022-06-30
```

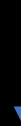


```
personnummer,medisin,startdato,sluttdato  
01010111111,vitaminbjørner,2022-08-22,2022-11-28  
22020222222,fluortabletter,2022-01-11,2024-03-08  
01010111111,tran,2022-06-01,2022-06-30
```

REFORMATTERING AV CSV

- Du har en CSV-fil med data, men det er på feil format

```
personnummer,medisin,startdato,sluttdato
01010111111,vitaminbjørner,2022-08-22,2022-11-28
22020222222,fluortabletter,2022-01-11,2022-03-03
01010111111,tran,2022-06-01,2022-06-30
```



- Produser en CSV-fil hvor data er på ønsket format

```
personnummer,medisin,dato,endring
01010111111,vitaminbjørner,2022-08-22,1
01010111111,vitaminbjørner,2022-11-28,0
22020222222,fluortabletter,2022-01-11,1
22020222222,fluortabletter,2022-03-03,0
01010111111,tran,2022-06-01,1
01010111111,tran,2022-06-30,0
```