LAB3

- mellomrom vs. tab
- `g(xi) == (1/8)*xi**2 - 2*xi + 10`

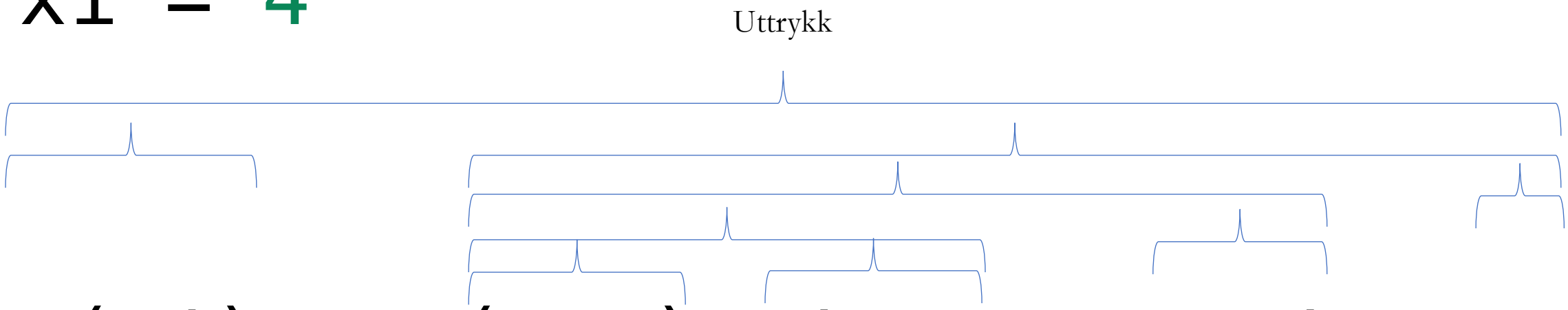
MELLOMROM vs TAB

- Konsekvent innrykk/indentering er viktig for riktig syntaks ved funksjoner/if/løkker og lignende
- Blanding av tab og mellomrom i samme fil: ☠️ ☠️ ☠️
- Noen prosjekter bruker tab, andre bruker mellomrom 😣

MERK!

- VSCode konverterer automatisk tab til mellomrom som standard 🤩
- Feilmeldingene forteller ofte hva som er galt

```
xi = 4
```




```
g(xi) == (1/8)*xi**2 - 2*xi + 10
```

Funksjonskall

Sammenligning
(operasjon med laveste presedens)

```
xi = 4
```

```
4.0  == (1/8)*xi**2 - 2*xi + 10
```

```
xi = 4
```

```
4.0  == 0.125*xi**2 - 2*xi + 10
```

```
xi = 4
```

```
4.0  == 0.125* 4**2 - 2*xi + 10
```

```
xi = 4

 4.0  == 0.125* 16   - 2*xi + 10
```

```
xi = 4

 4.0  ==    2.0       - 2*xi + 10
```

```
xi = 4

 4.0  ==    2.0      - 2*4  + 10
```

```
xi = 4


 4.0  ==    2.0       -  8    + 10
```

```
xi = 4


 4.0  ==          −6.0          + 10
```

```
xi = 4
```

```
 4.0  ==  4.0
```

```
xi = 4
```

```
True
```

= vs ==

- Tilordningsoperatør =
  - Venstreside må være et variabelnavn
  - Høyresiden er et uttrykk
- Likhetsoperatør ==
  - Del av et uttrykk: evaluerer til True eller False
  - Sammenligner høyre og venstre side
  - Ingen sideeffekter

I matematikk brukes = som "er lik"

"Vi definerer en funksjon"

$$g(x) = \frac{1}{8}x^2 - 2x + 10$$

Python:

```
def g(x):
    return (1/8)*x**2 - 2*x + 10
```

"(Det er sant at) venstresiden er lik høyreside"

$$2x + 4 = x + 8$$

Python(?):

```
assert(2 * x + 4 == x + 8)
```

"La x være lik 2"

$$x = 2$$

Python(?):

```
X = 2

x = 2
```

```
def x():
    return 2
```

UNIVERSITETET I BERGEN

STRENGER

INF100

HØST 2022

Torstein Strømme

I DAG

- Operasjoner på strenger
- Indeksering og beskjæring

- Hva er en streng *egentlig*?

STRENG

```
s = "Hei!\n”

s[0]     "H"
s[1]     "e"
s[2]     "i"
s[3]     "!"
s[4]     "\n"
```


indeks

- En ordnet samling med *tegn* (engelsk: character)
- Posisjonene er nummerert
- Én posisjon → ett tegn

TIDLIGERE

```
s1 = "Rød\n"
s2 = "Løk"

s1 + s2    # Konkatenasjon:  Rød\nLøk
s2 * 3     # Repetisjon:     LøkLøkLøk

len(s1)    # Lengde:         4
```

TIDLIGERE

```
x = 15
s1 = f'Ruud har {x} poeng'

y = 40
s2 = "Alcaraz har " + str(y) + ' poeng'
```

NOEN HAR PRØVD

```
s1 = "Rød\n"
s2 = "Løk"

max(s1, s2)  # Rød\n
s1 < s2      # False
```

Hvit løgn:
Sammenligner "alfabetisk"

Virkeligheten:
Sammenligner basert på symbolenes *unicode*-verdi

ANDRE OPERASJONER

```
s1 = "Rød\n"
s2 = "Løk"

"x" in s1    # False
"øk" in s2   # True
"" in s2     # True
```

```
s1 or s2     # Rød\n
s1 and s2    # Løk

if s1:
    blah()
```

IKKE GJØR DETTE HJEMME!

HERMETEGN og ESCAPE-SEKVENSER

https://inf100.ii.uib.no/notat/strenger

METODER PÅ STRENGER

```
s = "Rød\n"
s.lower()                  # rød\n
s.upper()                  # RØD\n
s.strip()                  # Rød
s.replace("ø", "oe")       # Roed\n
                           # ... mange flere
```


- En *metode* er en funksjon som kalles "på" et objekt (en verdi).
- Kalles med punktum etter objektet, så metodenavn og argumenter.
- Typen til objektet avgjør hvilke metoder som eksisterer

INDEKSERING

```
s = "Hei!\n"

s[0]    "H"
s[1]    "e"
s[2]    "i"
s[3]    "!"
s[4]    "\n"
```

Første tegn har indeks: 0

Siste tegn har indeks: len(s) - 1

INDEKSERING

```
s = "Hei!\n"
```

```
s[-5]       "H"
s[-4]       "e"
s[-3]       "i"
s[-2]       "!"
s[-1]       "\n"
```

Første tegn har indeks: - len(s)

Siste tegn har indeks: - 1

INDEKSERING

```
s = "Hei!\n"
```

```
s[5]
```
IndexError: string index out of range

```
s[-6]
```
IndexError: string index out of range

BESKJÆRING

```
s = "abcdefgh"

s[<start>:<end>]

s[0:3]   # 'abc'
s[1:3]   # 'bc'
s[2:3]   # 'c'
s[3:3]   # ''
```

BESKJÆRING

```
s = "abcdefgh"

s[<start>:<end>:<step>]

s[0:6:2]            # 'ace'
s[1:6:2]            # 'bdf'

s[0:len(s):3]       # 'adg’
s[7:3:-1]           # 'hgfe'
```

BESKJÆRING

```
s = "abcdefgh"

s[:6:2]             # 'ace'
s[1::2]             # 'bdfh'
s[::2]              # 'aceg'

s[:3]               # 'abc'
s[3:]               # 'defgh'
s[::-1]             # 'hgfedcba'
```

INDEKSERING OG BESKJÆRING

www.menti.com
2635 1745

TEKSTFIL

- Mange filformater er basert på "ren tekst"
- .txt
- .py/.js/.java/.cpp/.bat/.sh
- .csv/.json/.xml/.yaml/
- .html/.css/.tex/.md
- .svg
- …

Andre filformater er *ikke* basert på ren tekst:

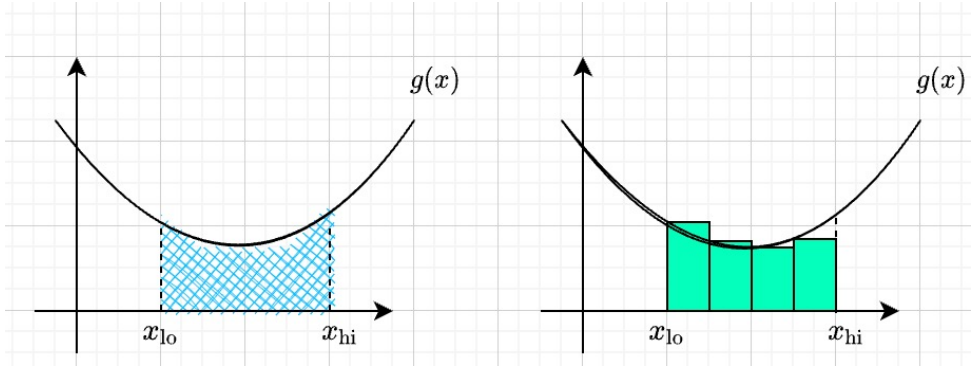
- .docx/.rtf/.pdf
- .xlsx
- .zip
- .mp3/.acc/.mp4
- .png/.jpeg/.gif

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)

python-circle.png

âPNG<CR>
^Z
<NUL><NUL><NUL><CR>IHDR<NUL><NUL>^Ah<NUL><NUL>^Ah<BS>^F<NUL><NUL><NUL>zÂa'<NUL><NU
<NP>A^Põ^Bë»ÅÌ(V<RS><FS>
·[êá@ŒÄ¯A~ÈÓ^W<ESC>AÚ^Vdx2ÙêƒFî^GA‡É^R1æGêa(<BS>c) –+Ú%ÔÌ˜<NP>5O{Ô/µw^Eø1Øj^QAdM$
◆}Î[_…nønœ¿^O»^Y¯Ÿ/|Òa£–∞Î–Â√¢0+ä¬^Z≠ñm„‰Àa^TZö$ñnSÛ<œNœŒ<RS>^æxiª<∑i``€mfÒzeií/†?
Å^EAp±òœmp2¥dì
›ö<US>^D^OìMlÂ
ÍV≠Vm>õÈ˜öç¶eª
›n˜Àqºyòeô UkUÀ
∑Ètj^•biſµ˜>xfr<◆.AE ¢aYLAzµ\Yø◊{ µZOV •1A A¬ U±H«ΔU≤X<NP>NU w…zcæÔ˙f^Px^W
tmΩãO{^K–Ô,ß¸]zœ_¸Ï°^OÔúúâ^QœÁã/üüü?ºº º^R3mw:^V<US>òm<FS>√p˜vrr"∞É˝Êª^B^lÎı1FÄ…Ô¨^V
^AR@˜ÙÙ'ÆØØ≠Ÿl <^A‰N´mÀÂ"^VÛÖ'Í5^A?0fi<US><NP>l_^T6^Yçu,õı F‡/l5^EÆæÁ[´›≤ÈxlQµ*9d
µ*%œ¸¿◊Ò$õç$^U^V^VéªZØÈ±0qæ&„…^EÔµT/?Ÿ%;°ıèøy^K/fl•kÓ]{ö[Ä~◊>Ò?«°˝Í^Wø^XÄ≠◊<ESC>_é

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

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 enkoding
- Enkoding: 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 æøåéüℒ†ƒ💩

Binary	Oct	Dec	Hex	Glyph 1963	Glyph 1965	Glyph 1967
010 0000	040	32	20	space		
010 0001	041	33	21	!		
010 0010	042	34	22	"		
010 0011	043					
010 0100	044					
010 0101	045					
010 0110	046					
010 0111	047					
010 1000	050					
010 1001	051					
010 1010	052					
010 1011	053					
010 1100	054					
100 0001	101	65	41	A		
100 0010	102	66	42	B		
100 0011	103	67	43	C		
100 0100	104	68	44	D		
100 0101	1					
100 0110	1					
100 0111	1					
100 1000	1					
100 1001	1					
100 1010	1					
100 1011	1					
100 1100	1					
110 0001	141	97	61			a
110 0010	142	98	62			b
110 0011	143	99	63			c
110 0100	144	100	64			d
110 0101	145	101	65			e
110 0110	146	102	66			f
110 0111	147	103	67			g
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
- Standard i Python 3

- 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					
010 0100	044					
010 0101	045					
010 0110	046					
010 0111	047					
010 1000	050					
010 1001	051					
010 1010	052					
010 1011	053					
010 1100	054					

100 0001	101	65	41	A
100 0010	102	66	42	B
100 0011	103	67	43	C
100 0100	104	68	44	D
100 0101	1			
100 0110	1			
100 0111	1			
100 1000	1			
100 1001	1			
100 1010	1			
100 1011	1			
100 1100	1			

110 0001	141	97	61	a
110 0010	142	98	62	b
110 0011	143	99	63	c
110 0100	144	100	64	d
110 0101	145	101	65	e
110 0110	146	102	66	f
110 0111	147	103	67	g
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

PALINDROM

- Et palindrom er en streng som er lik fremlengs og baklengs
- Skriv en funksjon som sjekker om en streng er et palindrom