



# NTNU

Det skapende universitet

**TDT4110 Informasjonsteknologi grunnkurs:**  
Kapittel 1 – Introduksjon til Programmering og Python

Førstelektor  
Børge Haugset



<https://www.youtube.com/watch?v=nKlu9yen5nc>

# Læringsmål og pensum

- Mål
  - Lære om programmering og hva er et program
  - Lære å designe et program
  - Lære om hvordan bruke Python
  - Lære om programmeringsomgivelse, skript, prompt, variabler
- Pensum
  - Starting out with Python, chapter 1 og 2.1

# Hva er programmering?

- Å programmere er å fortelle en datamaskin hva den skal gjøre.
  - Vi bruker et programmeringsspråk for å gjøre dette
  - Python er et programmeringsspråk
- Før en datamaskin kan kjøre et program, må programmet oversettes til maskinkode.
  - Maskinkode er et språk som prosessoren forstår
  - Programmet som oversetter programmeringsspråket til maskinkode kalles kompilator, tolker eller oversetter.

# Hva er programmering?

Skriv program

```
x=123  
y=321  
sum=x+y  
print(sum)
```

Python

↙  
Oversetter

```
01001010101010101  
01010101001111010  
10101010101010101  
01010101010100010  
10101010101010110  
10101010101011010
```

kode  
Maskin

Kjører prog



NTNU

Det skapende universitet

Skriv summen  
av 123+321  
på skjermen



# Hva er et program?

- Et program er en oppskrift med instruksjoner som forteller en datamaskin hva den skal gjøre
- Et program kan bestå av instruksjoner som:
  - Oppretter (deklarerer) og gir verdier (tilordne) til variabler
  - Evaluerer og regner på variabler
  - Gjør valg
  - Utfører ulike operasjoner (f.eks. viser grafikk, spiller av lyd, tar imot informasjon fra brukeren)
  - Osv.

# Hvordan et program fungerer

- Programmer utføres linje for linje (altså ei linje av gangen).
- Dette er uhyre viktig å forstå for å klare å henge med etter hvert som vi lanserer avanserte programmer.
- Vi snakker gjerne om en *programpeker* som flytter seg nedover linje for linje i programmet.
  - Det som står øverst *skjer* først!
  - Når vi kommer til funksjoner blir ting bittelitt mer avansert, de kjøres først når en ber dem om det...

# Koding er som matlaging. Omtrent. I alle fall ganske likt. På en måte...

## **Baking av målanalyse (ingredienser):**

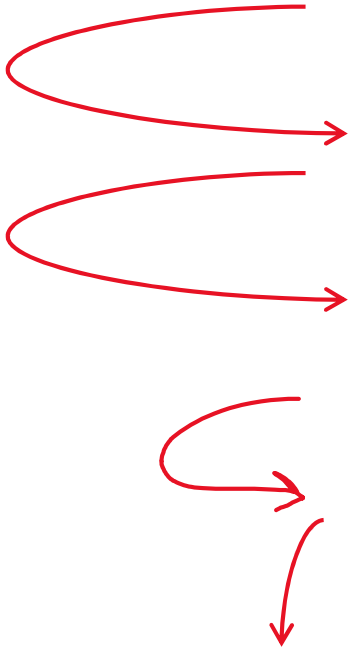
- Ett tall: hjemmemål
- Ett tall: bortemål

## **Baking av målanalyse (oppskrift):**

- 1: **hvis** det er flere hjemmemål enn bortemål, da skriver vi 'hjemmeseier' (ikke trenger vi sjekke mer heller)
- 2: **ellers sjekker** vi om det er flere bortemål enn hjemmemål, hvis det er sant skriver vi 'borteseier' og slutter å sjekke
- 3: **ellers** skriver vi 'uavgjort'



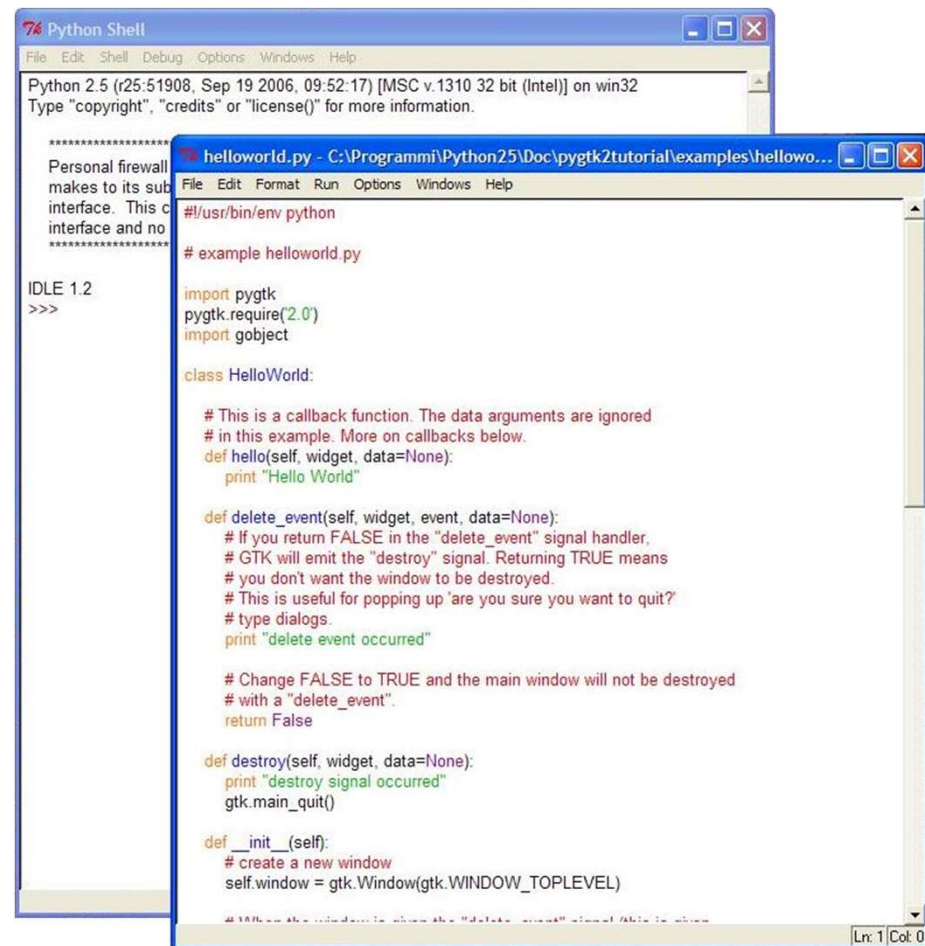
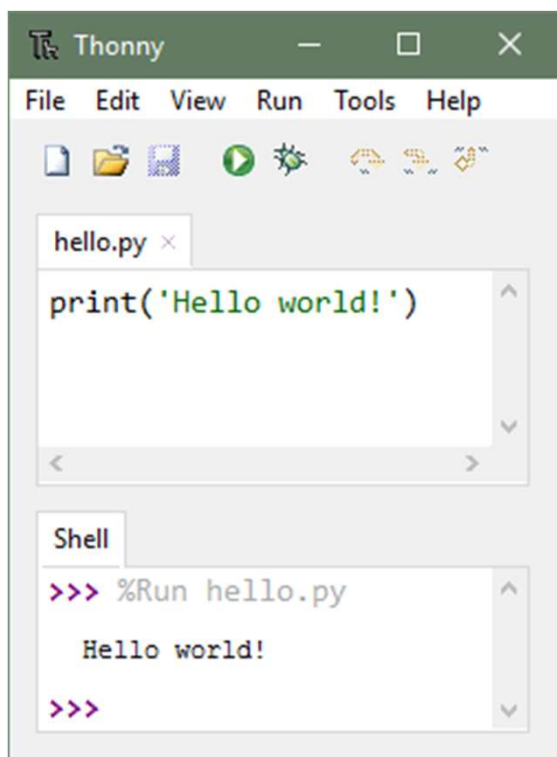
Koding er som matlaging. Omtrent.  
I alle fall ganske likt. På en måte...



```
homeGoals = 2  
awayGoals = 3  
  
if homeGoals > awayGoals:  
    print('Hjemmeseier')  
elif homeGoals < awayGoals:  
    print('Borteseier')  
else:  
    print('Uavgjort')
```

# Thonny versus IDLE

- Thonny enklere (synes jeg)

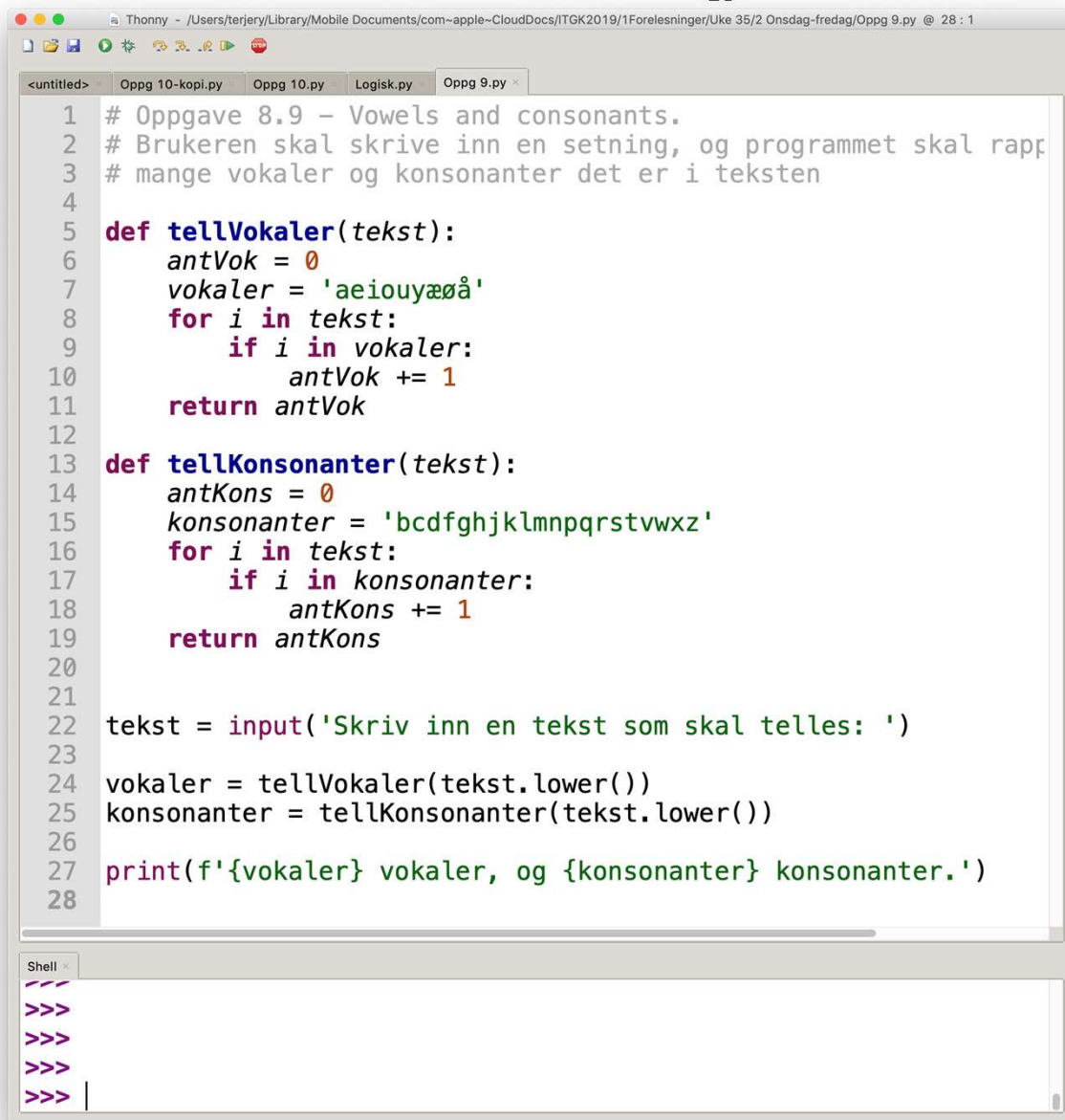


# Programmering i Python

- Python kan programmeres på følgende måter:
  - Interaktiv modus: Skrive en og en instruksjon inne i en tolker i Python-programmet
  - Lage Python programmer: Lage en tekstfil med Python-kommandoer ved hjelp av en teksteditor, og bruke Python-programmet til å oversette programmet til maskinkode og kjøre programmet. (Rett og slett som å lime det inn i interaktiv modus)

# Hva er et program?

- Et program er en oppskrift med instruksjoner som forteller en datamaskin hva den skal gjøre.



```
1 # Oppgave 8.9 – Vowels and consonants.
2 # Brukeren skal skrive inn en setning, og programmet skal rap
3 # mange vokaler og konsonanter det er i teksten
4
5 def tellVokaler(tekst):
6     antVok = 0
7     vokaler = 'aeiouyæøå'
8     for i in tekst:
9         if i in vokaler:
10             antVok += 1
11     return antVok
12
13 def tellKonsonanter(tekst):
14     antKons = 0
15     konsonanter = 'bcdfghjklmnpqrstvwxz'
16     for i in tekst:
17         if i in konsonanter:
18             antKons += 1
19     return antKons
20
21
22 tekst = input('Skriv inn en tekst som skal telles: ')
23
24 vokaler = tellVokaler(tekst.lower())
25 konsonanter = tellKonsonanter(tekst.lower())
26
27 print(f'{vokaler} vokaler, og {konsonanter} konsonanter.')
28
```

Shell

```
>>>
>>>
>>>
>>>
```

Ingrediensene er:

Variabler

Sløyfer

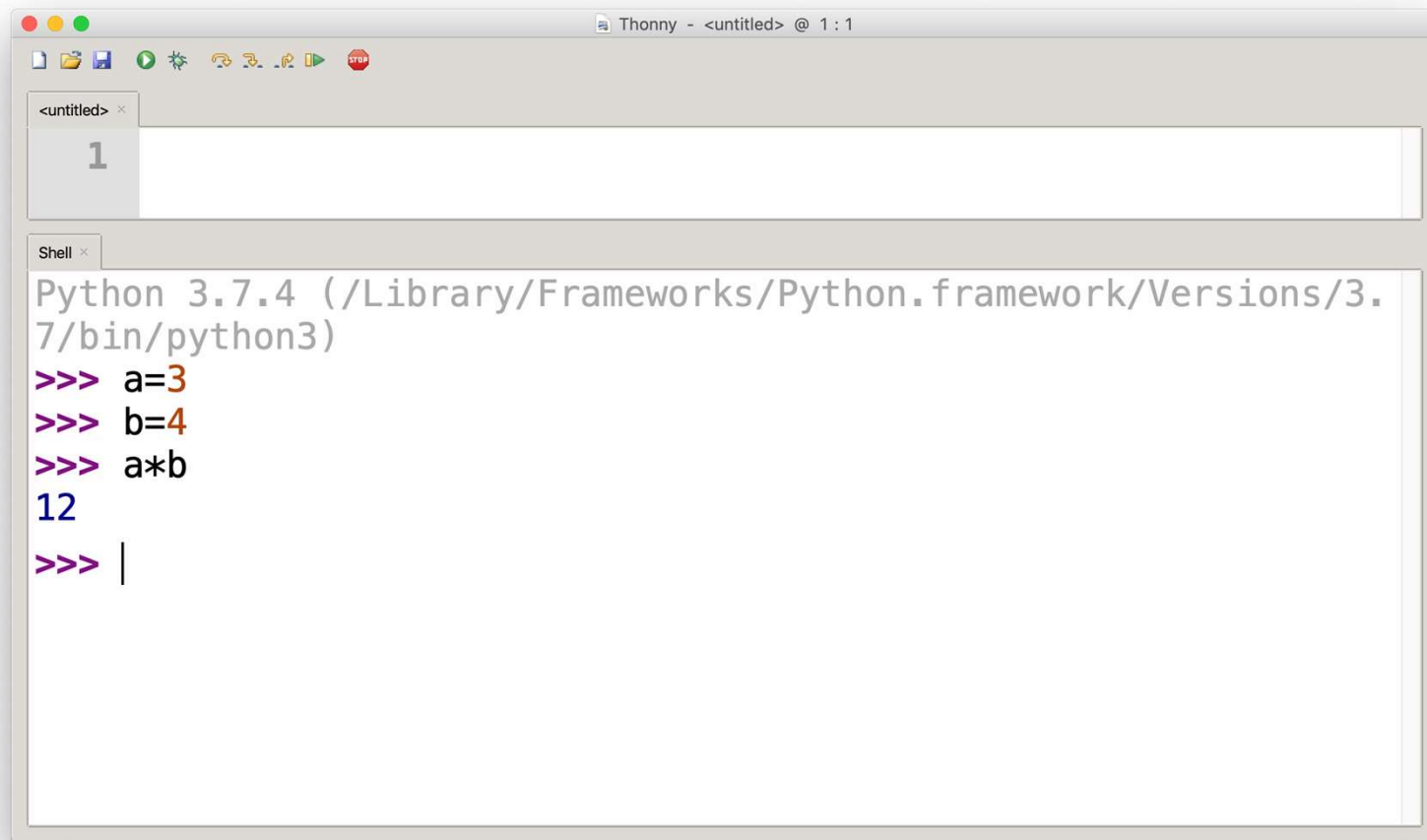
Valg

Funksjoner

Programmering blir å sette  
disse ingrediensene i rett rekkefølge

# Interaktiv modus

- Skrive en og en instruksjon inne i en tolker i Python-programmet - f.eks. i Thonny



The screenshot shows the Thonny Python IDE interface. At the top, there's a title bar that says "Thonny - <untitled> @ 1 : 1". Below it is a toolbar with various icons. The main window is divided into two panes. The top pane, labeled "<untitled> x", contains a single line with the number "1". The bottom pane, labeled "Shell x", displays the Python 3.7.4 interpreter's output. It shows the path "/Library/Frameworks/Python.framework/Versions/3.7/bin/python3" and the interactive prompt ">>>". The prompt has been used to enter three lines of code: "a=3", "b=4", and "a\*b". The result of the last line, "12", is displayed below the code. The prompt ">>>|" is currently at the bottom of the shell pane.

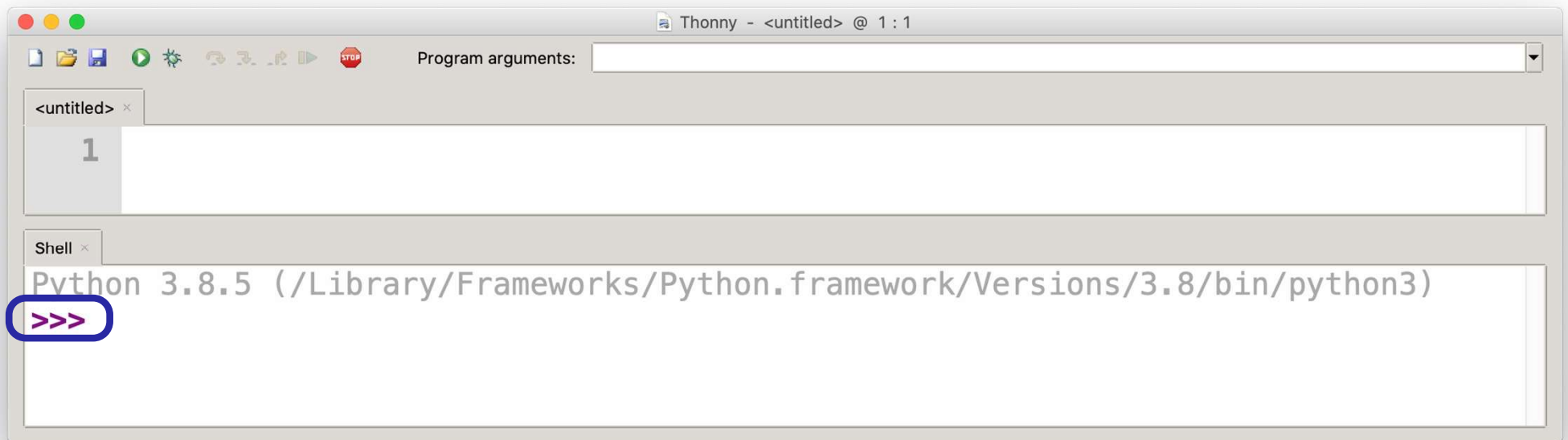
```
Thonny - <untitled> @ 1 : 1

<untitled> x
1

Shell x
Python 3.7.4 (/Library/Frameworks/Python.framework/Versions/3.7/bin/python3)
>>> a=3
>>> b=4
>>> a*b
12
>>> |
```

## Interaktiv modus

- Skriv en linje med en kommando og få direkte respons
- I interaktiv modus huskes alle variabler som brukes
- Får feilmelding hvis du har skrevet noe feil
- Startes ved å åpne IDLE eller starte “python”
  - Eller nederste ‘vindu’ i Thonny.
  - Programmeringen foregår i et såkalt “shell” (skall)
- Egner seg ikke til å skrive store programmer!



# Python som kalkulator

- Promptet ( `>>>` ) viser hvilket program du kjører og hvor du kan skrive kommandoer
- `+` `-` `*` `/` de 4 regneartene
- `5.82` desimal**punktum**
- `5//2` gir 2 heltallsdivisjon
- `5%2` gir 1 modulo
- $5^5$  angis med `5**5` eksponent
- Parentes brukes for å sikre korrekt utregning, eks  $(2+5)*7$  vs.  $2+5*7$ 
  - $(2+5)*7$  gir  $7*7$  altså 49
  - $2+5*7$  gir  $2+35$  altså 37

# Kort intro til Python

- De fleste kommandoer gjøres på ei linje.
- Variabler tar vare på informasjon: `x=5`
- Kan lagre tekst: `tekst='Dette er en test'`
- Kan teste logiske uttrykk: `23>3`
- Har innebygde funksjoner: `abs(-3), round(12.3)`



# Feilmeldinger

- Noen ganger (ofte?) får man feilmeldinger
  - Siste linje er den viktige



```
Thonny - <untitled> @ 1 : 1

Program arguments:

<untitled> x
1

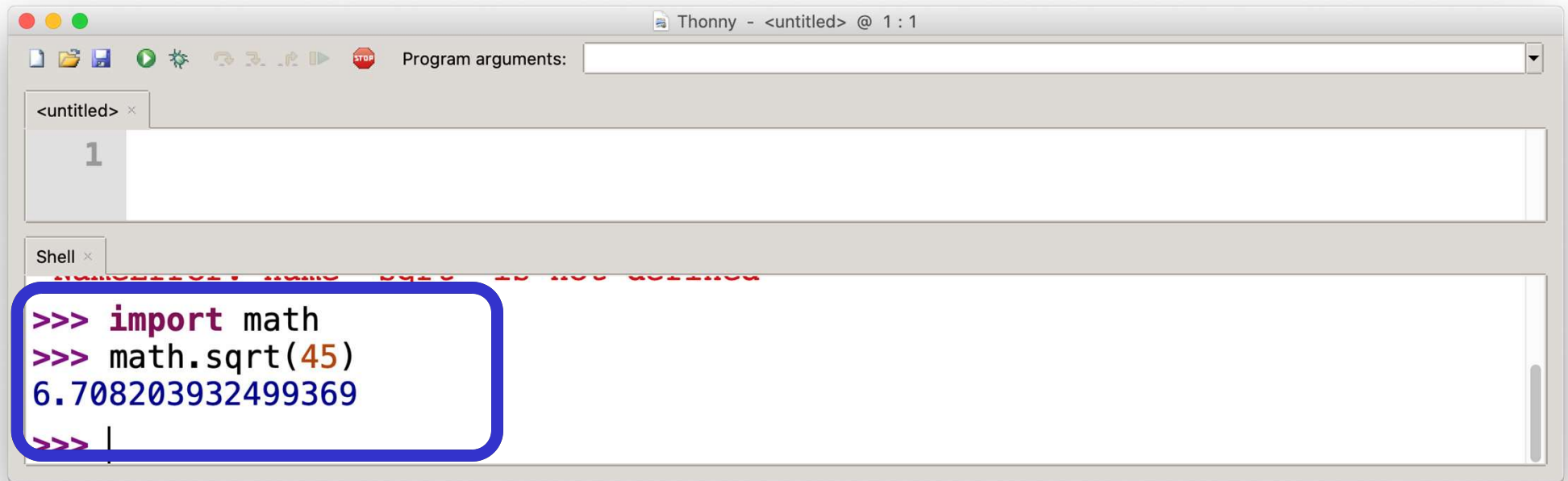
Shell x
Python 3.8.5 (/Library/Frameworks/Python.framework/Versions/3.8/bin/python3)
>>> sqrt(45)
Traceback (most recent call last):
  File "<pyshell>", line 1, in <module>
NameError: name 'sqrt' is not defined
>>>
```

Python forstår ikke kommandoen sqrt uten videre

`sqrt()` er ikke en av de innebygde kommandoene i python. For å bruke den må man importere et bibliotek - math - med kommandoen `import math`

# import

- Da vil det se slik ut:



The screenshot shows the Thonny Python IDE interface. At the top, there's a title bar that says "Thonny - <untitled> @ 1 : 1". Below it is a toolbar with various icons. The main area is divided into two panes. The top pane is labeled "<untitled> x" and contains a single line of code "1". The bottom pane is labeled "Shell x" and contains the following text:

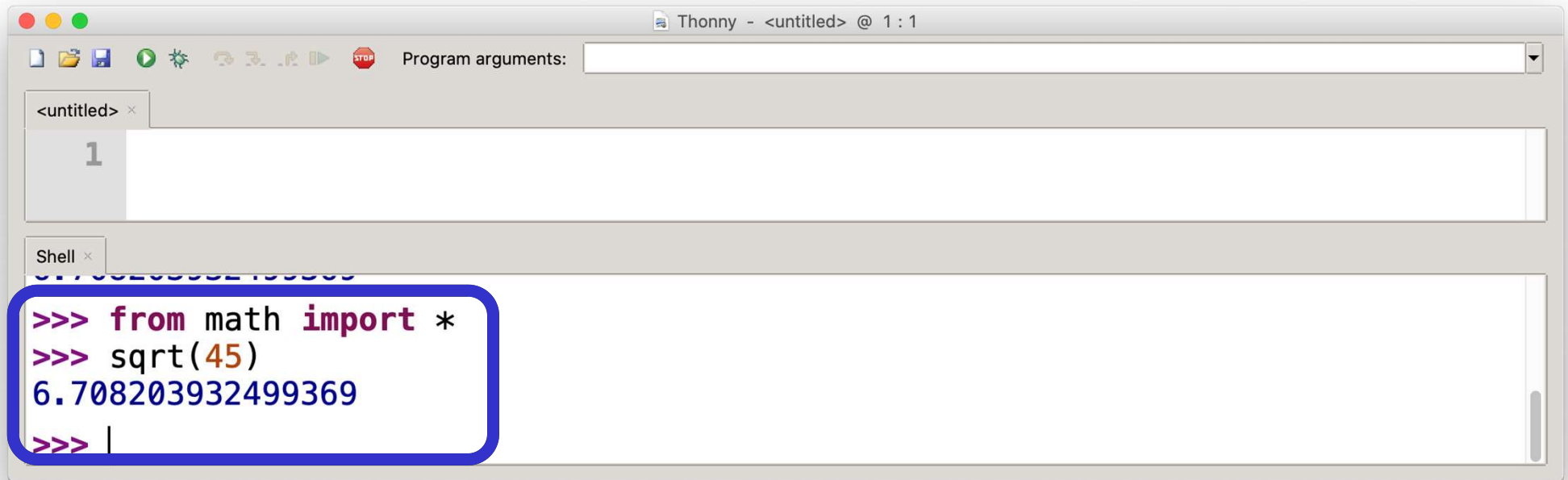
```
>>> import math
>>> math.sqrt(45)
6.708203932499369
>>> |
```

The first two lines of the shell output are highlighted with a blue rounded rectangle.

- Først importerer vi biblioteket math
- så bruker vi funksjonen i biblioteket ved å skrive
  - `math.sqrt(45)`

# import

- Alternativt




The screenshot shows the Thonny Python IDE interface. The top bar indicates the file is 'Thonny - <untitled> @ 1 : 1'. Below the toolbar, the code editor shows a single line of code: `1`. The bottom panel, labeled 'Shell', contains the following interactive session:

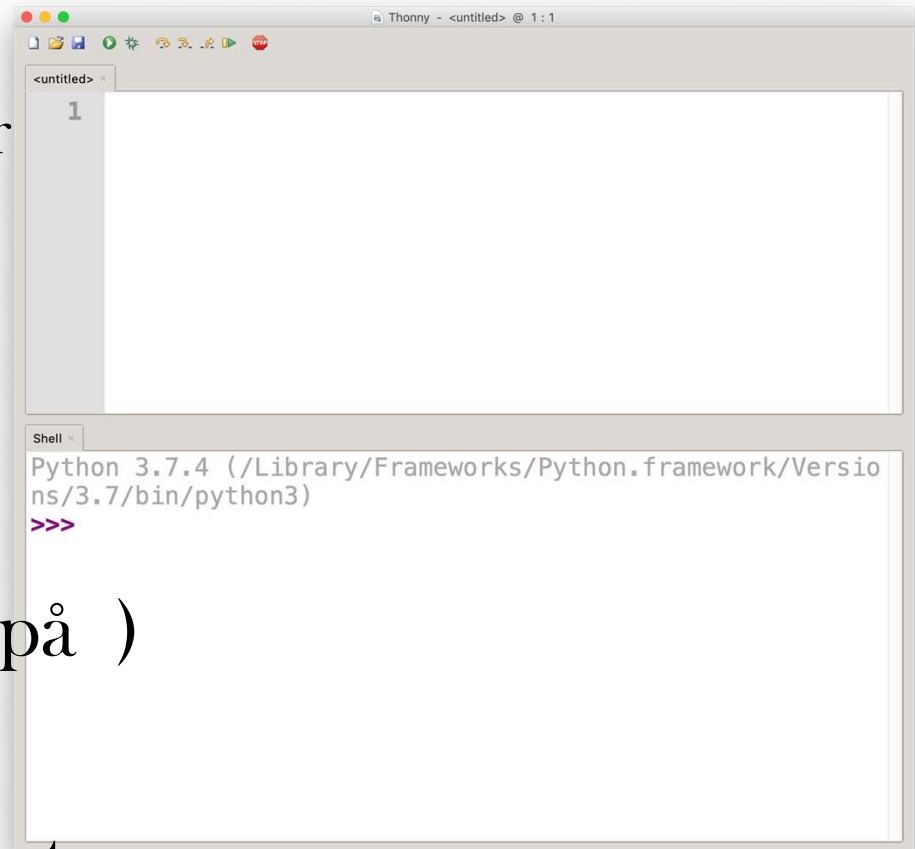
```
>>> from math import *  
>>> sqrt(45)  
6.708203932499369  
>>> |
```

The first three lines of the shell output are enclosed in a blue rounded rectangle.

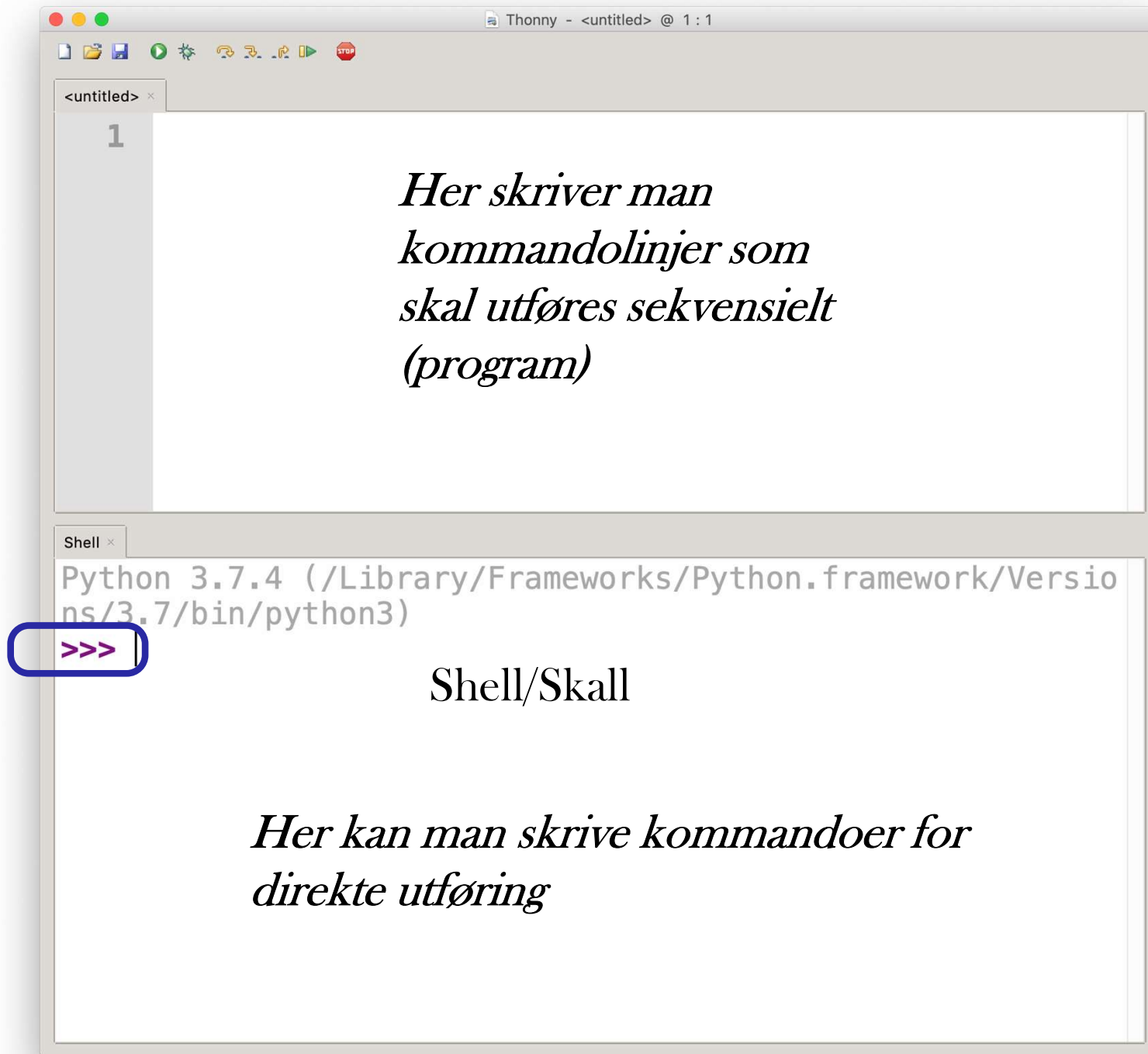
- Ved å skrive
  - `from math import *`
  - slipper vi å skrive `math.sqrt()`

# Hvordan lage Python programmer

-  Skriver et program med mange kommandoer i en teksteditor. Lagrer programmet med etternavn .py
- Må kjøre programmet ved hjelp av python-tolker for å få noe til å skje.
- Thonny er en enkel IDE for Python
  - Velg File / New (eller klikk på )
- For å kjøre programmet i Thonny:
  - Velg Run / Run current script (F5)



# Forskjellige rammer



# Python-programmer

- Lage en tekstfil med Python-kommandoer ved hjelp av en teksteditor.

The image shows a screenshot of the Thonny Python IDE. The top window displays a Python script for finding the most frequent character in a string. The script is as follows:

```
1 # Oppgave 8.10 – Most frequent character.
2 # Brukeren skal skrive inn en setning, og programmet skal rapportere hvilken
3 # bokstav som forekommer oftest.
4
5 tekst = input('Skriv inn en tekst som skal telles: ')
6 # tekst = tekst.lower()
7
8 bokstaver = 'abcdefghijklmnopqrstuvwxyzæøå'
9 antall = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
10
11 for bokstav in tekst.lower():          # oppdaterer antall ut fra hvilken
12     if bokstav in bokstaver:          # bokstav som finnes
13         indeks = bokstaver.index(bokstav)
14         antall[indeks] += 1
15
16 # Nå ligger antallet av de forskjellige bokstavene i listen antall
17 # Må finne den største verdien
18
19 maksVerdi = max(antall)                # Finner største verdi
20 indeks = antall.index(maksVerdi)        # Finner indeks for denne i antall
21 mestVanlig = bokstaver[indeks]         # Finner hvilken bokstav dette tilsvarer
22
23 print(f'{antall}\nDen mest vanlig forekommende bokstaven er: {mestVanlig}')
```

The bottom window shows a shell prompt with four lines of input, each consisting of three greater-than signs (>>>).

# Å knekke koden:

- Mengdetrening!
- Enkle ting skal sitte i fingrene
- Vi gjennomgår pensum på forelesning
  - Se på det på forhånd
  - Gjennomgå det etterpå – bruk det!
  - Benytt mulighetene som er tilstede
  - Du blir ikke flink til å sykle ved å se en Youtubevideo eller tre om det. Men – hvis du aper etter det du ser med egne eksempler...!
  - **Kunnskap** er det vi kan gi dere gjennom forelesninger og liknende.
  - **Kompetanse** er det dere selv får gjennom øvinger, terping, bruk.

# Hvorfor Python og ikke... Assembly?

## Skriv ut «Hello World!» på skjermen:

```
; Define variables in the data section
SECTION .DATA
    hello:    db 'Hello world!',10
    helloLen: equ $-hello

; Code goes in the text section
SECTION .TEXT
    GLOBAL _start

_start:
    mov eax,4        ; 'write' system call = 4
    mov ebx,1        ; file descriptor 1 = STDOUT
    mov ecx,hello     ; string to write
    mov edx,helloLen  ; length of string to write
    int 80h          ; call the kernel

; Terminate program
    mov eax,1        ; 'exit' system call
    mov ebx,0        ; exit with error code 0
    int 80h          ; call the kernel
```



# Oppsummering

- Et program forteller hva datamaskinen skal gjøre
- Vi skal bruke programmeringsspråket Python til å programmere datamaskinen
- Før datamaskina kan gjøre noe med programmet må det oversettes til maskinkode (byte code) ved hjelp av et program
- Oversettelsesprogrammet kalles en *kompilator*, *oversetter* eller *tolker*.
- Python kan programmeres *interaktivt* eller ved å *skrive et program* i en tekst editor.

# Noen kodeeksempler

- Felles for disse er at de gjerne bruker andre sin kode for å hjelpe seg. Moduler er essensielt i faktisk bruk av Python. Derfor vil vi inspirere litt!
  - Kan vi finne ut hvor langt det er fra A til B?
    - Hva hvis vi vil kjøre hele veien?
  - Kan vi hente ned noe fra yr.no og vise det frem?
  - Kan vi finne antall av ulike ord i en tekst?
  - Kalender?
  - Litt tegning av grafer, paier etc.