



**TDT4110 Informasjonsteknologi grunnkurs:**  
Tema: Filer og unntak (exceptions)

Terje Rydland - IDI/NTNU

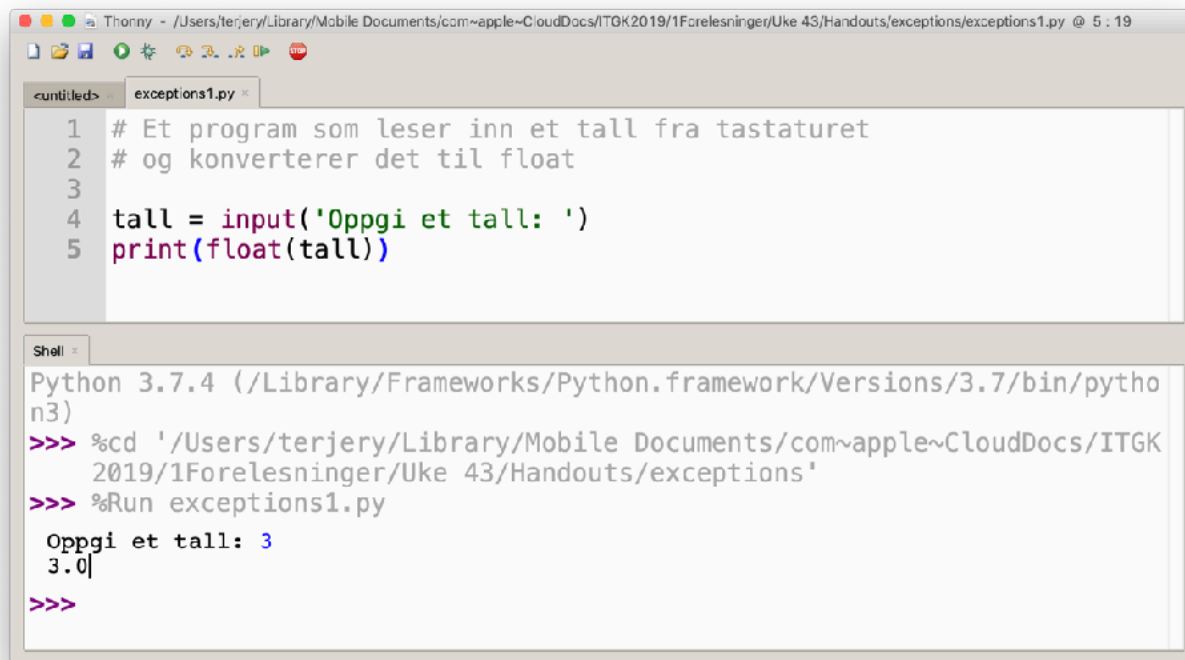
# Unntak/Feilhåndtering ("Exceptions")

Kapittel 6.4

# Feilhåndtering

- Tidligere
  - Brukt løkker for å unngå å få urimelige data (dividere på 0, alder større enn 120 år etc...)
  - Det håndterer ikke situasjoner der vi gir inn data av feil type
    - F.eks. Tekst der det forventes et tall og man skriver tallet med bokstaver

# Exception / Unntak



The screenshot shows the Thonny IDE with a file named `exceptions1.py`. The code in the editor is:

```
1 # Et program som leser inn et tall fra tastaturet
2 # og konverterer det til float
3
4 tall = input('Oppgi et tall: ')
5 print(float(tall))
```

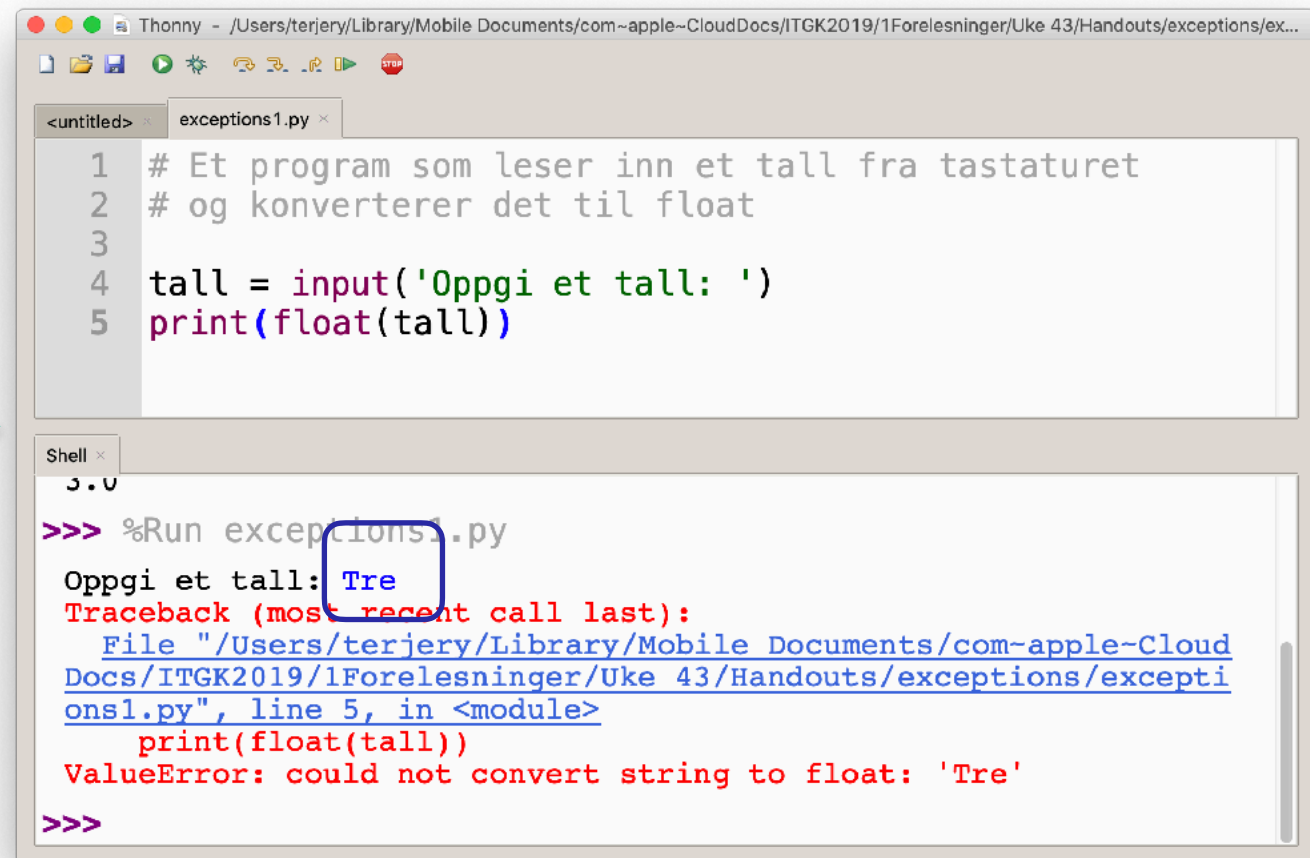
The Shell window shows the execution of the program:

```
Python 3.7.4 (/Library/Frameworks/Python.framework/Versions/3.7/bin/python3)
>>> %cd '/Users/terjery/Library/Mobile Documents/com~apple~CloudDocs/ITGK2019/1Forelesninger/Uke 43/Handouts/exceptions'
>>> %Run exceptions1.py
Oppgi et tall: 3
3.0
>>>
```

A green arrow points from the text "Dette går fint" to the code in the editor.

Dette går fint

Men dette...



The screenshot shows the Thonny IDE with the same file `exceptions1.py`. The code is identical to the previous screenshot:

```
1 # Et program som leser inn et tall fra tastaturet
2 # og konverterer det til float
3
4 tall = input('Oppgi et tall: ')
5 print(float(tall))
```

The Shell window shows the execution of the program with an error:

```
3.0
>>> %Run exceptions1.py
Oppgi et tall: Tre
Traceback (most recent call last):
  File "/Users/terjery/Library/Mobile Documents/com~apple~CloudDocs/ITGK2019/1Forelesninger/Uke 43/Handouts/exceptions/exceptions1.py", line 5, in <module>
    print(float(tall))
ValueError: could not convert string to float: 'Tre'
>>>
```

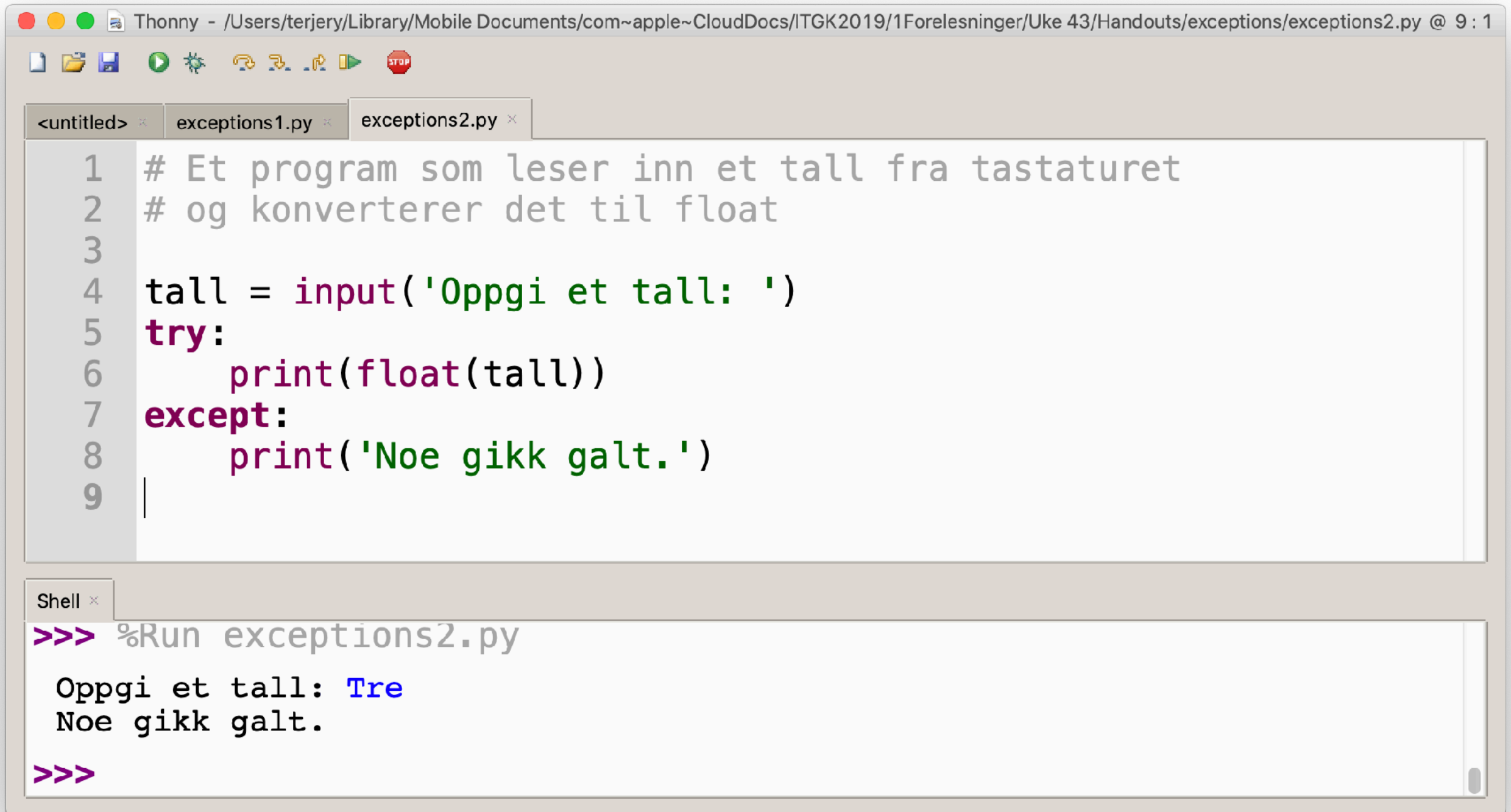
A green arrow points from the text "Men dette..." to the input "Tre" in the Shell window. A blue box highlights the input "Tre".

## Exception / Unntak

- En **exception** er en feil som oppstår under kjøring som får programmet til å stoppe opp.
- Typiske feil som gir **exception** er:
  - Prøver å gjøre om tekststrenger til tall med strenger uten tall
  - Divisjon på 0
  - Prøver å åpne filer som ikke eksisterer
- En måte å unngå dette er å sjekke bruker-input.
- I Python kan du også bruke **try/except** uttrykk for å unngå at programmet stopper opp under slike feil.

# Exception / Unntak

Fix



The screenshot shows the Thonny Python IDE interface. The title bar indicates the file path: `/Users/terjery/Library/Mobile Documents/com~apple~CloudDocs/ITGK2019/1Forelesninger/Uke 43/Handouts/exceptions/exceptions2.py @ 9 : 1`. The editor has three tabs: `<untitled>`, `exceptions1.py`, and `exceptions2.py`. The `exceptions2.py` tab is active, displaying the following Python code:

```
1 # Et program som leser inn et tall fra tastaturet
2 # og konverterer det til float
3
4 tall = input('Oppgi et tall: ')
5 try:
6     print(float(tall))
7 except:
8     print('Noe gikk galt.')
9 |
```

Below the editor is a 'Shell' window. It shows the command `>>> %Run exceptions2.py` being executed. The output of the program is:

```
Oppgi et tall: Tre
Noe gikk galt.
>>>
```

# Exception / Unntak

Eller enda bedre

```

1 # Et program som leser inn et tall fra tastaturet
2 # og konverterer det til float
3
4 while True:
5     try:
6         tall = input('Oppgi et tall: ')
7         print(float(tall))
8         break
9     except:
10        print('Noe gikk galt. Skrev du et tall? Du må skrive', end = ' ')
11        print('3, ikke Tre.')
12

```

```

>>> %Run exceptions3.py
Oppgi et tall: Fire
Noe gikk galt. Skrev du et tall? Du må skrive 3, ikke Tre.
Oppgi et tall: 5
5.0
>>>

```

```

1 # Et program som leser inn et tall fra tastaturet
2 # og konverterer det til float
3 feil = True
4 while feil:
5     try:
6         tall = input('Oppgi et tall: ')
7         print(float(tall))
8         feil = False
9     except:
10        print('Noe gikk galt. Skrev du et tall? Du må skrive', end = ' ')
11        print('3, ikke Tre.')
12

```

```

>>> %Run exceptions3.py
Oppgi et tall: tre
Noe gikk galt. Skrev du et tall? Du må skrive 3, ikke Tre.
Oppgi et tall: 3
3.0
>>>

```

## Exception: try – except uttrykk

- “Usikker” kode skrives inne i et **try:** uttrykk
  - Tester ut om denne koden kjører uten problemer
- I tillegg må vi legge til kode som fanger opp eventuelle feil  
**except ExceptionName:**

```
try:                                # En feil i try-blokka, trigger except
    uttrykk
    uttrykk
    ...
except ExceptionName: # Hopper hit hvis feil i try
    uttrykk
    uttrykk
```



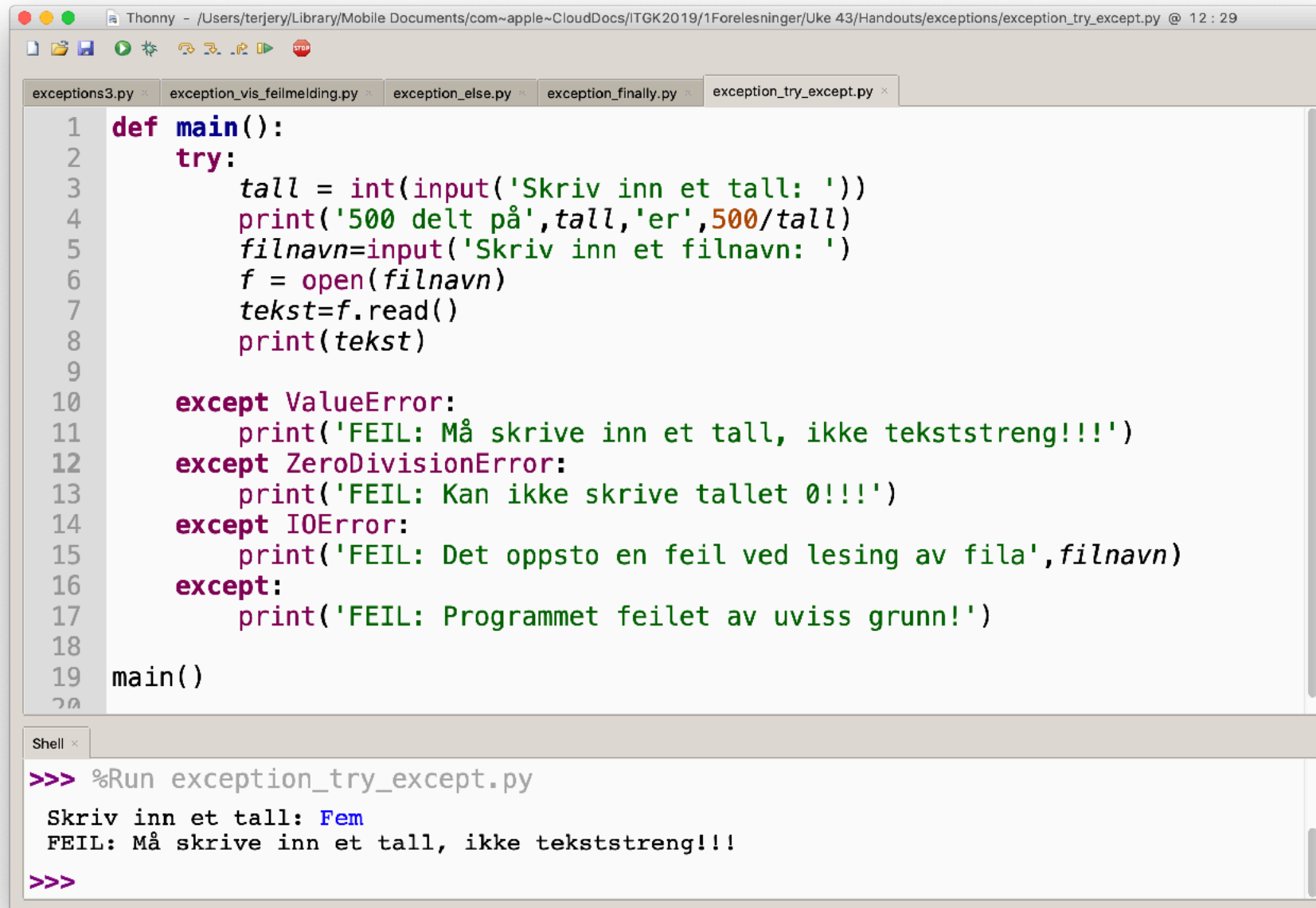
## Exception – ExceptionName

- Ulike typer Exceptions har ulike navn.
- Vi kan fange opp disse ved å lage en exception i kode.
- Typiske ExceptionName er:
  - ValueError: Typisk feil i datatype (streng når det skal være tall)
  - ZeroDivisionError: Prøver å dividere med 0
  - IOError: Feil med filbehandling
  - Exception: Alle mulige feil (generell)
- Ser på et eksempel på bruk av try – except:

# Kan bruke det for å fange opp feil i input

Mange mulige feil her:

- 1: Skriver inn en tekst -> `int()` feiler
- 2: Skriver 0 -> divisjon med 0
- 3: Oppgir et filnavn som ikke finnes



```
Thonny - /Users/terjery/Library/Mobile Documents/com~apple~CloudDocs/ITGK2019/1Forelesninger/Uke 43/Handouts/exceptions/exception_try_except.py @ 12 : 29

exceptions3.py x exception_vis_feilmelding.py x exception_else.py x exception_finally.py x exception_try_except.py x

1 def main():
2     try:
3         tall = int(input('Skriv inn et tall: '))
4         print('500 delt på',tall,'er',500/tall)
5         filnavn=input('Skriv inn et filnavn: ')
6         f = open(filnavn)
7         tekst=f.read()
8         print(tekst)
9
10    except ValueError:
11        print('FEIL: Må skrive inn et tall, ikke tekststreng!!!')
12    except ZeroDivisionError:
13        print('FEIL: Kan ikke skrive tallet 0!!!!')
14    except IOError:
15        print('FEIL: Det oppsto en feil ved lesing av fila',filnavn)
16    except:
17        print('FEIL: Programmet feilet av uviss grunn!')
18
19    main()
20

Shell x
>>> %Run exception_try_except.py
Skriv inn et tall: Fem
FEIL: Må skrive inn et tall, ikke tekststreng!!!
>>>
```

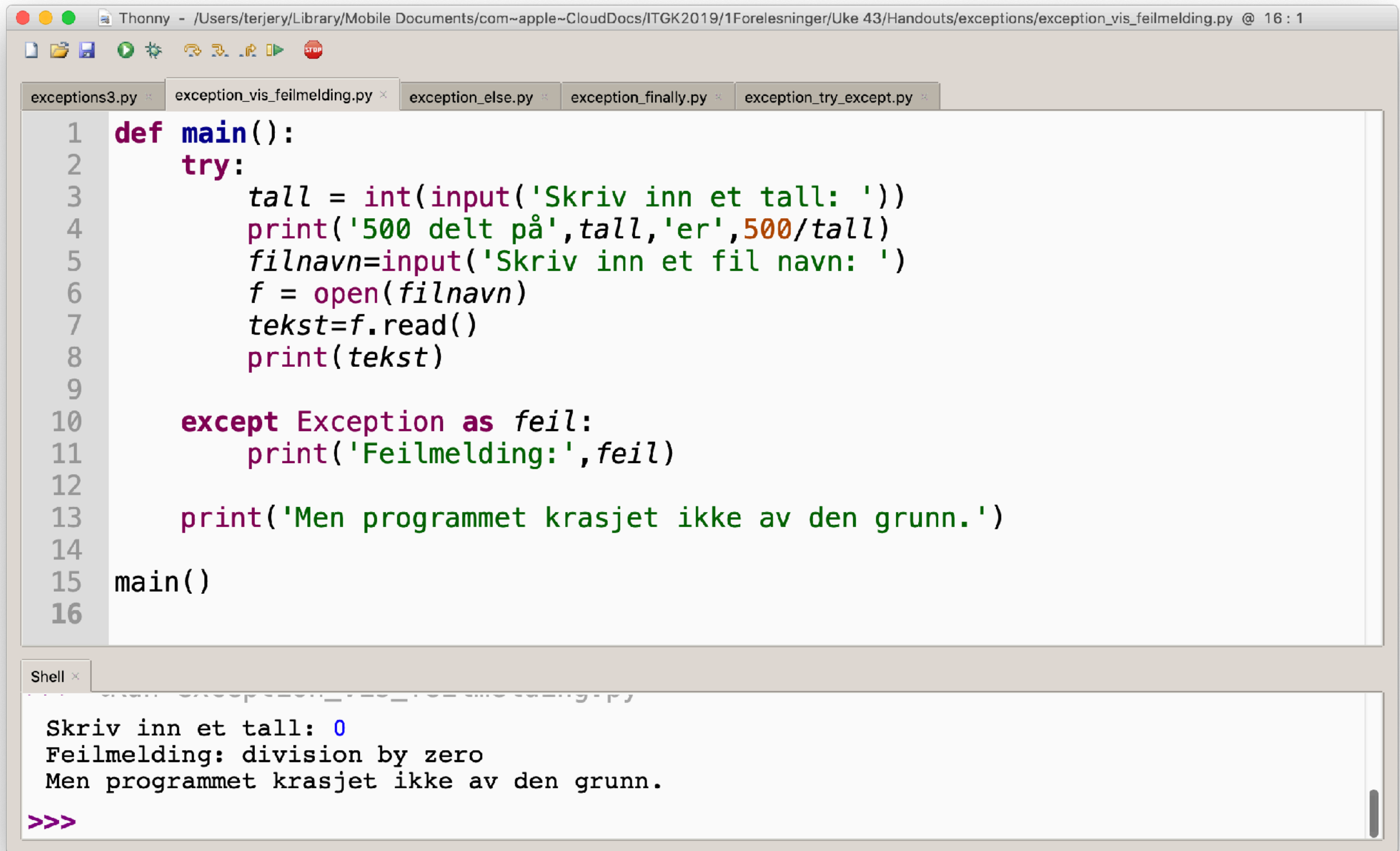
## Exception – vis innebygd feilmelding

- Ved mye kode kan det være vanskelig å vite hvilken feil som er oppstått
- Det er mulig å fange opp feilmeldingen som Python gir ved en Exception ved bruk av følgende kode:

```
try:  
    uttrykk...  
except Exception as variabel:  
    print(variabel)
```

- Uttrykket ***as variabel***, fanger opp feilen og lagrer feilmeldingen i en variabel som opprettes.
- Vi ser på et eksempel:

# Exception\_vis\_feilmelding.py



The screenshot shows a Thonny Python IDE window with the file `exception_vis_feilmelding.py` open. The code defines a `main()` function that prompts the user for a number and a filename. It attempts to calculate `500 / tall` and read the contents of the file. A `try` block encloses the division and file operations, with an `except Exception as feil:` block that prints the error message. After the `try` block, it prints a message indicating the program did not crash. The `main()` function is called at the bottom of the script.

```
1 def main():
2     try:
3         tall = int(input('Skriv inn et tall: '))
4         print('500 delt på', tall, 'er', 500/tall)
5         filnavn=input('Skriv inn et fil navn: ')
6         f = open(filnavn)
7         tekst=f.read()
8         print(tekst)
9
10    except Exception as feil:
11        print('Feilmelding:', feil)
12
13    print('Men programmet krasjet ikke av den grunn.')
14
15 main()
16
```

The Shell window at the bottom shows the execution output:

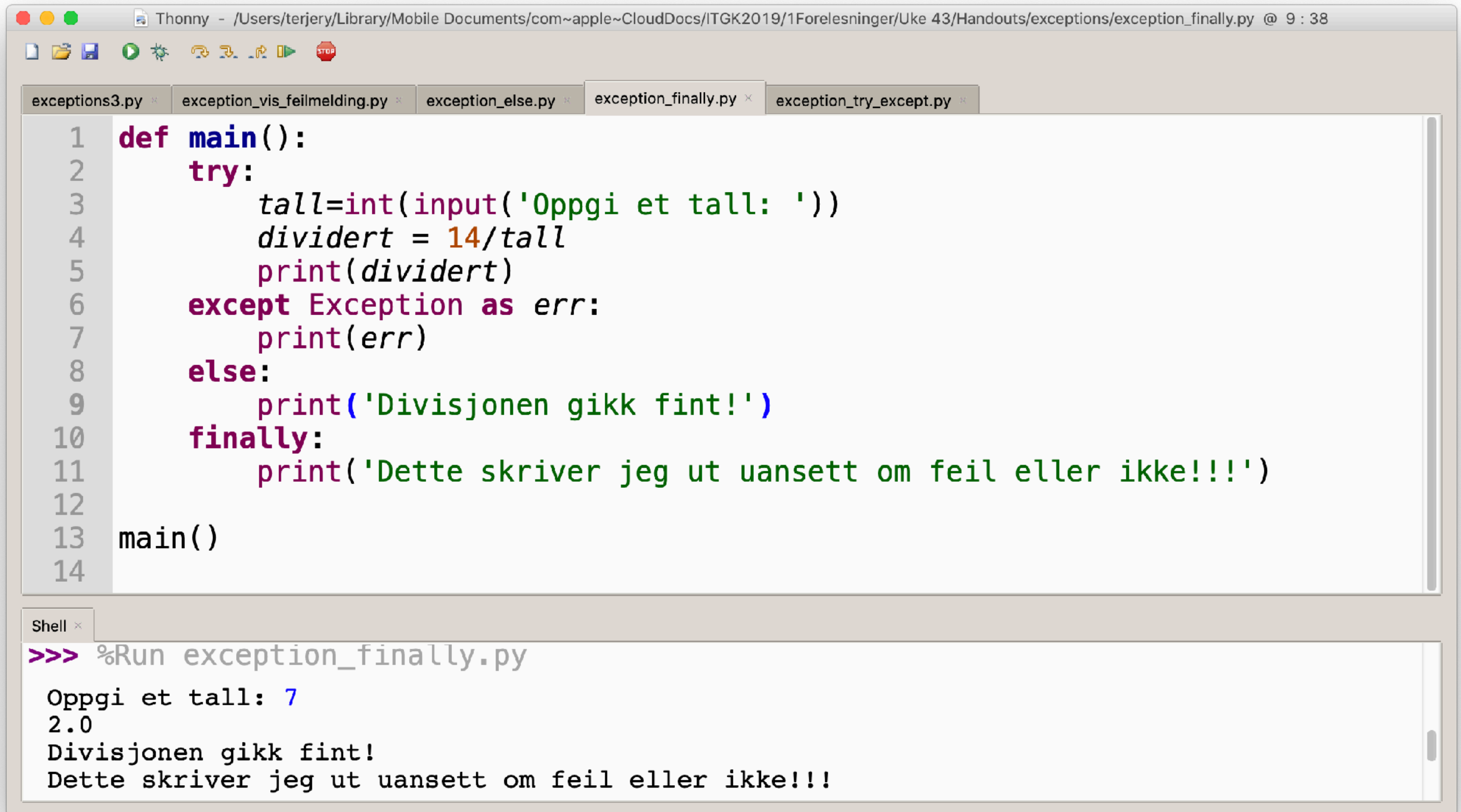
```
Skriv inn et tall: 0
Feilmelding: division by zero
Men programmet krasjet ikke av den grunn.
>>>
```

## Exception – else og finally

- Et `try – except` uttrykk kan også bestå av `else` og `finally`:
- `else` blir utført hvis ingen exceptions ble trigget.
- `finally` blir utført til slutt uansett om exceptions ble trigget eller ikke

```
try:
    uttrykk...
except ExceptionName:
    uttrykk...
else:
    uttrykk...
finally:
    uttrykk...
```

# exception\_finally.py



The screenshot shows a Thonny Python IDE window. The title bar indicates the file path: /Users/terjery/Library/Mobile Documents/com~apple~CloudDocs/ITGK2019/1Forelesninger/Uke 43/Handouts/exceptions/exception\_finally.py @ 9 : 38. The editor displays the following Python code:

```
1 def main():
2     try:
3         tall=int(input('Oppgi et tall: '))
4         dividert = 14/tall
5         print(dividert)
6     except Exception as err:
7         print(err)
8     else:
9         print('Divisjonen gikk fint!')
10    finally:
11        print('Dette skriver jeg ut uansett om feil eller ikke!!!')
12
13 main()
14
```

Below the editor is a Shell window with the following output:

```
>>> %Run exception_finally.py
Oppgi et tall: 7
2.0
Divisjonen gikk fint!
Dette skriver jeg ut uansett om feil eller ikke!!!
```

# exception\_finally.py



The screenshot shows the Thonny Python IDE interface. The top window displays the code for `exception_finally.py`. The code defines a `main()` function that uses a `try` block to attempt a division, an `except` block to catch any exception, and a `finally` block to execute a print statement regardless of whether an exception occurred. The `main()` function is then called. Below the code editor, the Shell window shows the execution output, which includes the prompt 'Oppgi et tall:', the error message 'division by zero', and the final print statement 'Dette skriver jeg ut uansett om feil eller ikke!!!'.

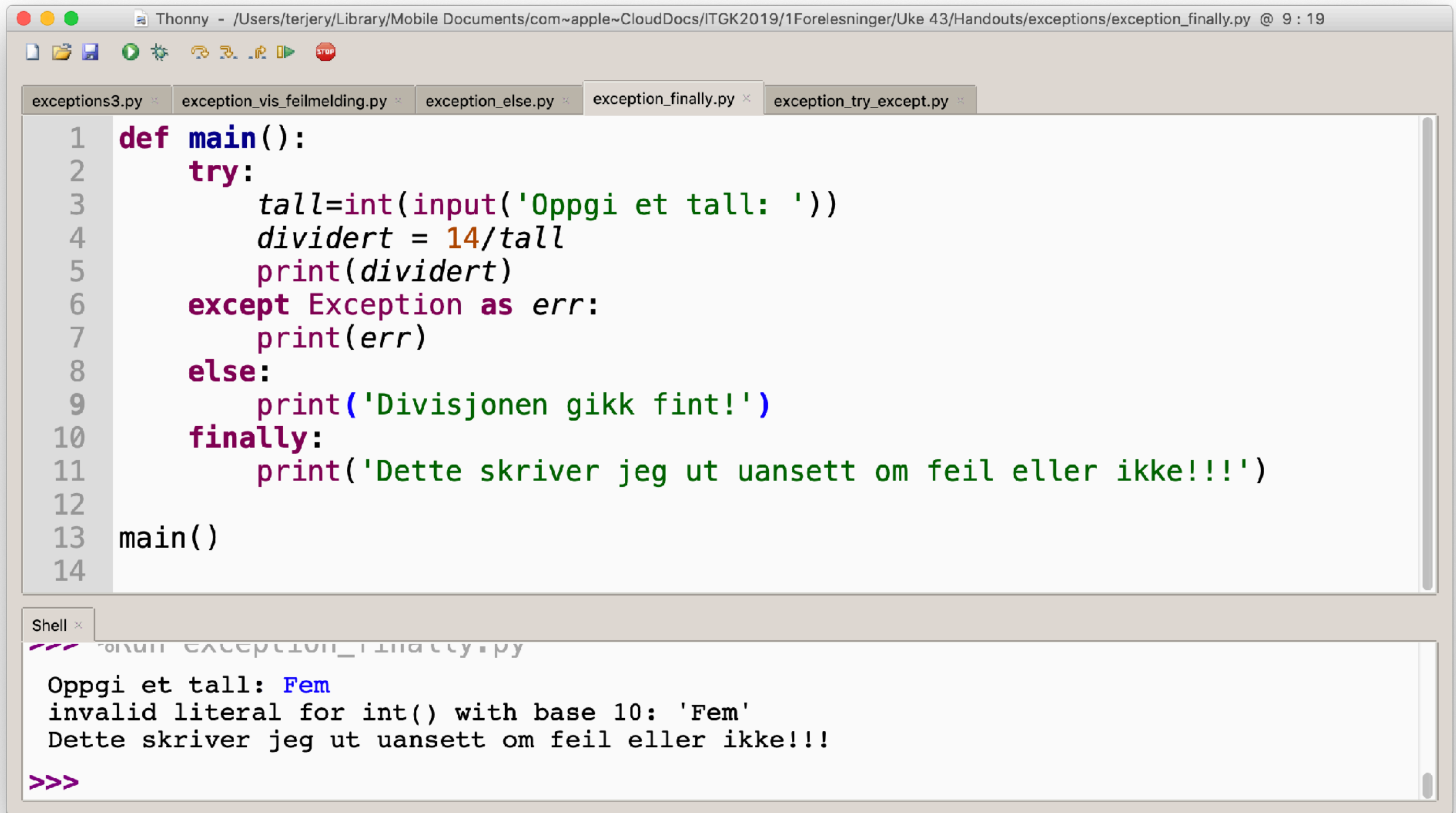
```
1 def main():
2     try:
3         tall=int(input('Oppgi et tall: '))
4         dividert = 14/tall
5         print(dividert)
6     except Exception as err:
7         print(err)
8     else:
9         print('Divisjonen gikk fint!')
10    finally:
11        print('Dette skriver jeg ut uansett om feil eller ikke!!!')
12
13 main()
14
```

Shell

```
Python 3.7.4 Shell
>>> from exception_finally.py
>>> Oppgi et tall: 0
>>> division by zero
>>> Dette skriver jeg ut uansett om feil eller ikke!!!
>>>
```



# exception\_finally.py



```
Thonny - /Users/terjery/Library/Mobile Documents/com~apple~CloudDocs/ITGK2019/1Forelesninger/Uke 43/Handouts/exceptions/exception_finally.py @ 9 : 19

exceptions3.py × exception_vis_feilmelding.py × exception_else.py × exception_finally.py × exception_try_except.py ×

1 def main():
2     try:
3         tall=int(input('Oppgi et tall: '))
4         dividert = 14/tall
5         print(dividert)
6     except Exception as err:
7         print(err)
8     else:
9         print('Divisjonen gikk fint!')
10    finally:
11        print('Dette skriver jeg ut uansett om feil eller ikke!!!')
12
13 main()
14

Shell ×
~/Documents/ITGK2019/1Forelesninger/Uke 43/Handouts/exceptions/exception_finally.py

Oppgi et tall: Fem
invalid literal for int() with base 10: 'Fem'
Dette skriver jeg ut uansett om feil eller ikke!!!

>>>
```





```
Thonny - /Users/terjery/Library/Mobile Documents/com~apple~CloudDocs/ITGK2019/1Forelesninger/Uke 44/Python 9/exceptions/med try-except.py @ 19 : 15

<untitled> x med try-except.py x

1 def leggTilNoe(dikt):
2     try:
3         navn = input('For hvem vil du oppdatere dictionaryen: ')
4         leggTil = input('Oppgi det som skal legges til: ')
5         dikt[navn].append(leggTil)
6     except AttributeError:
7         dikt[navn] = [dikt[navn], leggTil]
8     except KeyError:
9         dikt[navn] = leggTil
10
11 minDikt = {'Per': 92925492}           # Opprette
12 print(minDikt)
13
14 minDikt['Nils'] = 92939495          # Legge til
15 print(minDikt)
16
17 leggTilNoe(minDikt)
18
19 print(minDikt)

Shell x
Python 3.7.4 (/Library/Frameworks/Python.framework/Versions/3.7/bin/python3)
>>> |
```

## Exceptions / Unntak

- Exception: feil som skjer når et program kjører
  - Som regel fører det fører det til at programmet stopper (kræsjer)
- Exception handling: Håndtere ”exceptions” ved å gi brukeren fornuftig tilbakemelding uten at programmet stopper helt opp.
- Benytter:

```
try:          # Prøv om koden lar seg kjøre
except       # Fanger opp hvis koden i try feiler
except Exception as variable: # fanger feilmelding
else:        # Kjøres hvis det ikke blir exception
finally:     # Kjøres uansett til slutt
```