Konseptforståelse Uke 2

Gruppe 10 - Prosa

Spørsmål?

- Epost: simontha@uio.no
- Discourse er oppe!
  - https://astro-discourse.uio.no/c/in1000-25h/671

Plan for i dag

- Datatyper
- Typekonvertering
- Boolske uttrykk og operatorer
- Ulike typer feil
- Prosedyrer
- Kodeflyt

Læringsmål for uka

Læringsmål denne uken

- Forstå hvordan én enkelt linje utføres: Datatyper, evaluering av uttrykk og funksjoner
- Ha god forståelse av variabler
- Forstå og kunne bruke enkle prosedyrer uten parametre
- Forstå hvordan et helt program utføres - kodeflyt fra linje til linje, inkludert for beslutninger og prosedyrer

F-strings

- Ikke et krav i IN1000 (men veldig nyttig)
- Er nyttig for utskrift med tekst + variabler (og eventuelle operasjoner med variabler)

```
print("Vanlig string")

print(f"f-string")
```

- Kan legge til variabler i utskrift lettere

```
print("Du er", variabel, "år gammel")

print(f"Du er {variabel} år gammel")
```

```python
flyt = 8.789324678362782938746

print(f"Tallet ditt er {flyt:.2f}")
```

```
Tallet ditt er 8.79
```

Datatyper

- Stor forskjell mellom datatyper i Python
- Vanligste datatyper:

```
streng = "Dette er en string" # String
heltall = 67 # Int
flyttall = 6.72 # Float
sannhetsverdi = True # Bool
```

- De er ikke venner!

```
amalgamasjon = streng + heltall + sannhetsverdi
```

```
  File "c:\Users\timot\OneDrive - Universitetet i Oslo\U
    amalgamasjon = streng + heltall + sannhetsverdi
                   ~~~~~~~~^~~~~~~~~~
TypeError: can only concatenate str (not "int") to str
```

Løsning: Typekonvertering

- Python har noen innebygde funksjoner, blant annet for å konvertere typer til variabler
- Konverterer variablen til en annen
- Nyttig for å 'blande' variabler
- Nyttig for input

```
str(heltall)
int(flyttall)
float(heltall)
bool(streng)
```

Diskusjonsoppgave
5-10 min

```
str(heltall)
int(flyttall)
float(heltall)
bool(streng)
```

- Eksperimenter litt med de forskjellige typekonverteringene
- Er det noen som fungerer med hverandre?
  - Er det noen operatorer som fungerer med forskjellige typer?
- Hvilke fungerer / fungerer ikke? Er det noe som virker annerledes enn forventet?

Hva gjør egentlig input?

- Input() tar inn brukerinput (dette kan dere)
- Alt brukerinput konverteres automatisk til strings
  - Dette må vi være obs på !
- Derfor lurt å konvertere med en gang.

```
# Input først, konvertere etterpå
heltall = input("Skriv inn et heltall: ")
heltall = int(heltall)

# Alt i samme slengen
heltall = int(input("Skriv inn et
heltall: "))
```

- Ikke alt kan konverteres til alt, kan gi feil i koden din. 2 løsninger:
  - Fancy: Sjekk om input er numerisk (ikke pensum)
  - Vanlig: Anta at bruker gir gyldig input (dersom annet ikke er spesifisert i oppgaven)

Boolske uttrykk og operatorer

- Bool er en datatype i python
- En boolsk variabel har to statuser, **True** eller **False | 0/1**
- Viktig med stor forbokstav i True og False
- Alle sammenligninger fører til en boolsk verdi
- Hva skjer egentlig under en if-sjekk?
- Hva printes ut her? Dere kan selv anta verdien av h1/h2

```
print(h1 == h2)
print(h1 >= h2)
print(h1 != h2)
print(h1 <= h2)
```

And, or, not

- Kodeord for å (blant annet) utvide funksjonen av if-sjekker
- Slipper å nøste masse if-sjekker sammen
- **And**-uttrykk er bare sanne når **begge** sider av uttrykket er sanne
- **Or**-uttrykk er sanne når **begge eller en** av sidene er sanne
- **Not** inverterer (**det motsatte**) av verdien
- + mange flere operatorer
  - https://www.w3schools.com/python/python_operators.asp

```
a = 15
b = 20

if a or b > 14:
    print()
```

```
if a > 14 or b > 14:
    print()
```

Hva printes ut?

```
if True and True:
    print(1)

if True or False:
    print(2)

if True:
    print(3)

if False:
    print(4)

if True and (False or True):
    print(5)

if not True and not False:
    print(6)

if not not True:
    print(7)

if False or (False and False):
    print(8)
```

Hva printes ut?

```
if True and True:
    print(1)

if True or False:
    print(2)

if True:
    print(3)

if False:
    print(4)

if True and (False or True):
    print(5)

if not True and not False:
    print(6)

if not not True:
    print(7)

if False or (False and False):
    print(8)
```

Ulike typer feil

- Syntaksfeil
  - Får ikke kjørt koden, problemer med syntaks
- Logiske feil
  - Feil med logikken i koden, ofte din egen feil
- Kjøretidsfeil
  - Feil som oppstår under kjøring av programmet, f.eks konkatenering av to ulike datatyper, ugyldig typekonvertering

Prosedyrer

- Egentlig ikke en greie i python -> funksjoner
  - Dette kommer det mer om senere
- En måte å gjenta deler av koden på etter behov
- Tillater deg å kjøre spesifikk kode basert på hendelser under kjøretid

```
def recipeMenu():
    print('Welcome to all the recipes')
    print('==========================')
    print('1. Search for a recipe')
    print('2. Add a recipe')
    print('3. Quit')

recipeMenu()

choice = input('What option do you want? ')
```

Livekoding