Uke 2, time 1

Onsdag 27. august

Introduksjon til Python

Konseptforståelse

Pythons innebygde datatyper

Evaluering av uttrykk

Prosedyrer

Kodeflyt

Datatyper

Det er mange datatyper i Python, men de vanligste vi skal bruke er

- str – string, eller tekst
- int – integer, eller heltall
- float – float, eller flyttall
- bool – boolean, eller boolsk/dikotom (sant/usant)

str

```
fornavn = "Sara"

bok_tittel = "Ringenes Herre"

velkomst = 'Velkommen til IN1000'

mange_linjer = """
Dette er tekst som går over
flere linjer!
"""
```

int

```
tallet = 42

alder = 28

antall_appelsiner = 5

negative_tall = -69
```

float

```
ingenting = 0.0

pi = 3.1415

appelsin_vekt = 0.86

uendelig = float('inf')

nanometer = 1e-9
```

bool

```
sant = True

usant = False

ikke_sant = not sant

>>> print(sant and usant)
False

>>> print(sant or usant)
True
```

Typekonvertering

Når vi bruker `input()` så får vi en string

Man kan konvertere mellom du ulike typene ved å bruke `str()` (konvertere til string) `int()` (til heltall), `float()` (til flyttall) og `bool()` (til boolean)

Hvis Python ikke klarer å konvertere så får vi en ValueError feil

Fra	Til	Resultat
float	int	Alt etter komma fjernes
int/float	bool	0 = False, alt annet = True
str	bool	tom string = False, ellers True
bool	int / float	True = 1, False = 0

Operatorer

Operatorer er bl.a. + (pluss) - (minus) * (gange) / (dele) osv.

De har ulik funksjon avhengig av datatype

Rekkefølgen avhenger av hvilke operator vi bruker, men følger stort sett vanlige matematiske regler, og vi kan også bruke parenteser (,), for at et uttrykk

https://docs.python.org/3/reference/expressions.html#operator-precedence

Operatorer

For stringer så vil + slå sammen to til en string

For heltall og flyttall så vil + addere de til en ny verdi

Som regel så kan vi ikke blande sammen ulike datatyper, men må eksplisitt konvertere

Vi får feilen TypeError om vi blander sammen datatyper

Operatorer

```
print("4" + "2")                    # 42

print(4 + 2)                        # 6

print(int("4") + int("2"))          # 6

print(int("4" + "2"))               # 42
```

Uttrykk

Et uttrykk er et sett med kommandoer, verdier og variabler med operatorer

Når vi vil tilordne en verdi til en variabel, da må vi vite hvordan uttrykk evalueres

Alt til høyre for = evalueres *først*, og det er resultatet som variabelen får

Til venstre kan vi bare ha variabel

Tilordning

```
appelsin_vekt = 0.4

appelsin_vekt = appelsin_vekt + 0.3

appelsin_vekt += 0.3
```

Trix

Oppgave 02.02, 02.03, 02.04, 02.05, 02.06

.

Boolske uttrykk

Dette er uttrykk som gir oss en boolsk verdi, altså enten True eller False

De vanligste operatorene er

```
==   !=   <   >   <=   >=   and   or   in   is   not
```

Vi kan bruke disse med bl.a. if og elif, og vi kan tilordne til variabler

De fungere også forskjellig avhengig av datatype

Boolske uttrykk

```
>>> print("a" != "b")
True

>>> print("hest" == "best")
False

>>> print("hest" > "fest")
True
```

Boolske uttrykk

```
>>> print(-1 < 0)
True

>>> print(not True)
False

>>> print(True and True)
True
```

Obs ved flyttall

```
# Hva skjer her?

if 1.1+2.2 == 3.3:
    print("Riktig!")
else:
    print("Feil")

print(1.1+2.2)
print(3.3)
```

Trix

Oppgave 02.07, 02.08, 02.10, 02.11

.

Prosedyre

Det kan være vanskelig å få struktur med en sammenhengende lang rekke med kommandoer, så da vil vi dele de opp i subrutiner

Kodeblokker som vi ønsker å kunne kjøre flere ganger eller bestemte deler i programmet vårt kaller vi for prosedyrer

Vi gir de navn, og de kjøres når vi kaller på de

Det går også ann å kalle på en prosedyre fra en annen

Prosedyrer

```
def jegogså():
    prosedyre()
    print("Jeg er også en prosedyre")

def prosedyre():
    print("Jeg er en prosedyre")

prosedyre()
prosedyre()
print("-")
jegogså()
```

Kodeflyt

Normalt kjøres programmet linje for linje, fra topp til bunn

Men nå kan vi også bruke prosedyrer, og beslutninger til å styre flyten

På den måten kan vi skrive programmer som gir oss bedre oversikt

Vi må også passe på at variabler er definert når vi skal bruke de i koden

Kodeflyt

```
def hest():
    print("Jeg er en hest")

def sau():
    print("Jeg er en sau")

dyr = input("Skriv et dyr: ")
if dyr == "hest:
    hest()
elif dyr == "sau":
    sau()
else:
    print("Vi har ikke dyret " + dyr)
```

Litt om kommentarer

Kommentarer i Python starter med #

Vi bruker de for å fortelle oss, eller andre som leser koden hva som skjer. Men normalt er det ikke nødvendig med kommentarer som gjentar kommandoene eller variabelnavnene

I prosedyrer, funksjoner og metoder så kan vi bruke "docstrings" som er en forklaring på hva prosedyren gjør. Den setter vi inn rett under navnet, og lages med """ tekst """, altså en flerlinje-string

```
appelsiner = 5 # antall appelsiner
epler = 3      # antall epler
antall_frukt = appelsiner + epler

def prosedyre():
    """En prosedyre som ikke gjør noe nyttig"""
    pass
```

Trix

Oppgave 02.01, 02.09, 02.12, 02.13, 02.14, osv.

.

Oppgaver

Oppgave 1

1. Lag fire variabler, *heltall1*, *heltall2, flyttall1 og flyttall2*. Tilordne variablene tall som passer med variabelnavnene ved bruk av input().
2. Skriv ut summen og produktet av de to heltallene, deretter av de to flyttallene.
3. Undersøk hva som skrives ut i terminalen når dere summerer og multipliserer et heltall med et flyttall. Hva ser dere? Finn ut hvorfor oppførselen er slik.
4. Hva skjer hvis dere summerer et heltall eller et flyttall med en tekststreng?

Oppgave 2

1. Kjør koden nedenfor og gi en detaljert forklaring av innholdet i feilmeldingen. Hva slags type feil er det snakk om? Diskuter hvordan feilen kan rettes.
2. Endre koden, slik at programmet kjører feilfritt. Løs feilen på to forskjellige måter.
3. Utvid programmet slik dere ønsker, men legg til en logisk feil i utvidelsen. Hva er mulige konsekvenser av den logiske feilen i deres utvidede program?

```
flyselskap = "Norwegian"
rutenummer = 8537
print(flyselskap + "s", "rute " + rutenummer + ".")
```

Oppgave 3

Diskuter hvordan boolske uttrykk (og boolske operatorer) henger sammen med evaluering av uttrykk i programmering. Bruk begrepet sannhetsverdier i diskusjonen, og finn eksempler med og uten bruk av tall.

Oppgave 4

Beskriv flyten i programmet nedenfor i detalj (når hvilke programlinjer kjøres og hva som nøyaktig skjer i hver programlinje). Skriv deretter koden inn i Python Tutor og sammenlign Python Tutors beskrivelse av programflyten med deres egen.

```
def opprett_bankkonto():
    print("Velkommen til opprettelse av bankkonto!")
    saldo = float(input("Skriv inn inngående saldo: "))
    if(saldo == 0):
        print("Du kan ikke opprette konto med 0 kroner.")
    elif(saldo < 0):
        print("Du kan ikke opprette konto med et negativt beløp.")
    else:
        print("Konto er opprettet med " + str(saldo) + " kroner.")

opprett_bankkonto()
```