PYTHON KOMMANDOKART

PROFAG
UNIVERSITETET I OSLO

GRUNNLEGGENDE PYTHON

- Tips til feilsøking

- Les og prøv å tolke feilmeldingen! De to nederste linjene viser ofte plassering av feilen og type feil.
- Det kan være feil i linja over den feilmeldingen foreslår.
- Skriv ut variabler underveis – da ser du hva du har å gjøre med.
- Kommenter ut snutter i programmet for å kjøre utvalgte deler.
- Spør noen andre om å lese gjennom programmet ditt.
- Bruk dokumentasjon på nettet flittig.

- Generelt

# kommentar	Alt etter # vil bli oversett av programmet
Triple anførselstegn på hver side av koden	Kommentere ut kode over flere linjer
'\n'	linjeskift

- Datatyper

Tilordning av verdi til en variabel er gitt ved:

a = 1	heltall
b = 1.0	flyttall
c = "hei" *eller* c = 'hei'	streng (tekst)
d = True *eller* d = False	boolsk

For å omgjøre en variabeltype til en annen:

float(1)	output: 1.0
int(1.0)	output: 1
string(1.0)	output: '1.0'

Husk at det er forskjell på teksten '1' og tallet *1*.

- Output og input

print('hei!')	output: 'hei'
a = input()	lagrer input som tekst i *a*
a = float(a)	Konverterer *a* til flyttall

PROGRAMFLYT

- Beslutninger

```
if <betingelse>:
    <gjør dette>
elif <betingelse>:
    <gjør noe annet>
else:
    <gjør noe annet>
```

Bytt ut det som står inni <betingelse> med en logisk betingelse (som f.eks. *a < 1*), og <gjør dette> med en beslutning (f.eks. *print(a)*).

- While-løkker:

```
while <betingelse>:
    <gjør dette>
```

Eksempel:

```
i = 0
while i < 5:
    print(i)
    i = i + 1
```

- For-løkker:

```
for <tellevariabel> in range(fra, til, steglengde):
    <gjør dette>
```

Eksempel:

```
a = 2
for i in range(10):
    print(a**i)
```

Hvis vi bruker henholdsvis to argumenter eller ett argument, fungerer *range* som *range(fra, til, steglengde = 1)* og *range(fra = 0, til, steglengde = 1)*

- Funksjoner:

```
def funksjonsnavn(parameter1, parameter2,...):
    return verdi
```

Eksempel:

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

LISTER OG ARRAYER

- Lister

Lister lages ved bruk av klammeparentes: [element1, element2].

Legge til	
Legge til element	listenavn.append(element)
Eksempel:	tall.append(3)
Slette	
Sletter første element i lista med gitt navn	listenavn.remove(element)
Slette element nr. *a*	del listenavn[a]
Slette element nr. *a*	listenavn.pop(a)
Telle	
Lengden på lista	len(listenavn)
Antall av et visst element	listenavn.count(elementnavn)
Maksimum tallverdi i lista	max(listenavn)
Minimum tallverdi i lista	min(listenavn)
Sortere	
Reversere lista	listenavn.reverse()
Sorte lista fra minst til størst	listenavn.sort()
Finne og bruke elementer	
Hente element nummer *a* (fra 0)	listenavn[a]
Hente element *a* til *b*	listenavn[a:b]
Returnerer *True* hvis element er i liste	element in listenavn

- Arrayer

For å lage arrayer, må et nytt bibliotek importeres:
from pylab import *

Opprette array	
Fra liste til array	array(listenavn)
Array med *n* 0-er	zeros(n)
Array med *n* jevnt fordelte tall fra *a* til *b*	linspace(a,b,n)
Finne og bruke elementer	
Hente element nummer *a* (fra 0)	arraynavn[a]
Hente element *a* til *b*	listenavn[a:b]
Matematiske operasjoner	
Skalarprodukt av array A og B	dot(A,B)
Kryssprodukt av A og B	cross(A,B)

- Tupler

Tupler lages med vanlig rund parentes.
Tupler er statiske datasamlinger som ikke kan endres.
Eksempel: tuppel = (1, 2, 'a', 'b')

UiO : Universitetet i Oslo

PYTHON KOMMANDOKART

PROFAG
UNIVERSITETET I OSLO

OPERATORER

Eksemplene bruker a=10 og b=20:

- Aritmetiske operatorer

Op.	Beskrivelse	Eksempel
+	Addisjon	a + b gir output: 30
-	Subtraksjon	a - b gir output: -10
*	Multiplikasjon	a * b gir output: 200
/	Divisjon	b / a gir output: 2
%	Modulus	a % b gir output: 0
**	Eksponent	a**b gir output: 10^{20}
//	Heltallsdivisjon	9 // 2 gir output: 4

- Matematiske funksjoner

Matematiske funksjoner trenger biblioteket *pylab*:
from pylab import *

sqrt(a)	Kvadratrota av *a*
log10(a)	Briggske logaritmen av *a*
log(a)	Naturlige logartimen av *a*
exp(a)	e^a
sin(a)	Sinus av *a* gitt i radianer
cos(a)	Cosinus av *a* gitt i radianer
tan(a)	Tangens av *a* gitt i radianer
arcsin(a)	Invers sinus av *a*, gir radianer
arccos(a)	Invers cosinus av *a*, gir radianer
arctan(a)	Invers tangens av *a*, gir radianer
radians(a)	Omgjør *a* til radianer
degrees(a)	Omgjør *a* til grader
abs(a)	Absoluttverdien til *a*

- Logiske operatorer

10 == 10	Logisk lik
10 != 11	Ikke lik
12 > 10	Større enn
10 >= 10	Større enn eller lik
10 < 11	Mindre enn
a < b and a > c	begge kriterier sanne
a < b or a > c	én av kriteriene sanne
a not b	gir True hvis *a* ikke er *b*

- Tilfeldige tall

Tilfeldige tall trenger biblioteket *pylab*:
from pylab import *

randint(a, b)	Tilfeldig heltall fra og med *a* til *b*.
uniform(a, b)	Tilfeldig flyttall mellom *a* og *b*

PLOTTING, GRAFIKK OG DATABEHANDLING

- Plotting

Plotting trenger biblioteket *pylab*:
from pylab import *

Funksjon	Beskrivelse
plot(x,y)	Plotter y mot x.
show()	Datamaskinen viser plottet
title('tittel')	Tittel på plottet
xlabel('x-aksetittel')	Beskrivelse av x-aksen
ylabel('y-aksetittel')	Beskrivelse av y-aksen
legend(['graf1', 'graf2'])	Liste med merkelapper på grafene
xlim(fra, til)	Definisjonsverdi (x-aksen)
ylim(fra, til)	Verdimengde (y-aksen)
grid()	Skru på rutenett
axhline(y=0, color='black')	x-akse
axvline(x=0, color='black')	y-akse
subplot(rader, kolonner, nr.)	Separate plott i samme bilde

- Grafikk: Turtle

Plotting trenger biblioteket *turtle*: from turtle import *

Leonardo = Turtle()	Lager et turtle-objekt
Leonardo.shape('arrow')	Gir form til objektet
Leonardo.color('blue')	Gir farge til objektet
Leonardo.speed(1)	Tegnefart fra 1 til 10 (0 er raskest)
Leonardo.pos()	Gir posisjonen til objektet
Leonardo.forward(50)	Flytter objektet 50 piksler fram
Leonardo.backward(50)	Flytter objektet 50 piksler tilbake
Leonardo.left(45)	Vender 45 grader mot venstre
Leonardo.right(45)	Vender 45 grader mot høyre
Leonardo.penup()	Streken tegnes ikke
Leonardo.goto((20,30))	Flyttes til punktet (20,30)
exitonclick()	Avslutter når en klikker på ruta

- Lese fra fil

fil = open("filnavn.type", "r")	Åpner og leser (r = read) fila
innhold = fil.read()	Lagrer innholdet som streng
readline()	Leser/hopper over ei linje
fil.close()	Lukker fila etter bruk
loadtxt('filnavn.type', float)	Lager array av en fil vha. pylab

STRENGER

Strenger og output av strenger kan manipuleres med en rekke operasjoner. Eksemplene nedenfor benytter denne variabelen:
tekst = 'Hei, verden!'

Finne og bruke elementer

tekst[2]	output: 'i'
tekst[4]	output: ' '
tekst[1:3]	output: 'ei'

Husk at mellomrom også er et tegn,
og husk at indeksering starter på 0!

Dele opp strenger

tekst.split(' ')	output: ['Hei,','verden!']
tekst.split(',')	output: ['Hei',' verden'!']

Tegnet i parentes er *separatoren* vi ønsker å splitte med.

Operatorer på strenger

Vi bruker følgende strenger til eksemplene nedenfor:
a = ['Hallo'] og b = ['verden!']

Oper.	Beskrivelse	Eksempel
+	Trekker sammen strengene	a + b output: 'Halloverden'
*	Repeterer strengen	a*2 output: 'HalloHallo'
in	Inneholdes i streng eller liste	'r' in b output: True
%	Formatering av strenger	Eksempler følger nedenfor

Strengformatering

Symbol	Betydning
%f	formatering av flyttall
Generell syntaks:	'tekst %.<ant. desimaler>f' % (tall)
Eks.: '%.2f' % (3.14159)	output: 3.14
%d	formatering av heltall
Generell syntaks:	'tekst %d' % (tall)
Eksempel: '%d' % (42)	output: 42
%s	formatering av streng
Generell syntaks:	'tekst %<plass mot høyre>s' % (streng)
Eks.: '%10s' % ('Knut')	output: ' Knut'
Eks.: '%-10s' % ('Knut')	output: 'Knut '

Et alternativ til strengformatering av tall
er funksjonen *round(tall, antall desimaler)*.

UiO : Universitetet i Oslo