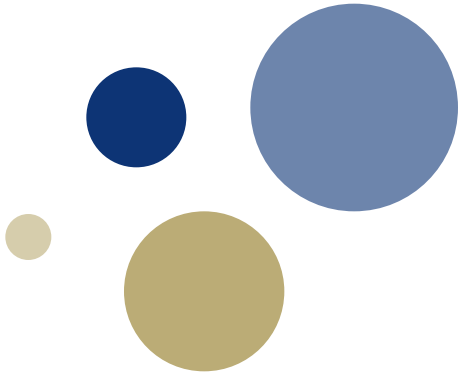
NTNU

Kunnskap for en bedre verden



Python: Funksjoner og moduler

Kapittel 5.7-5.10

TDT4110 IT Grunnkurs

Professor Guttorm Sindre

Læringsmål og pensum

- Mål
  - Kunne lage og kalle funksjoner med returverdi
  - Bruke bibliotek i Python, f.eks **random** og **math**
  - Vite hvordan vi oppnår gjenbruk av funksjoner
- Pensum
  - Starting out with Python: Chapter 5.7-5.10 / 6 Value-Returning Functions and Modules

Hvorfor trenger vi dette?

- Slippe å skrive om igjen kode som allerede er laget!
- Sentralt problem:
  - funksjoner må kommunisere med resten av programmet
- Tre mulige løsninger
  1. Ingen vesentlig kommunikasjon
     - Funksjonen leser selv sine inndata med input ( )
     - Funksjonen skriver selv ut resultat med print( )
  2. Via globale variable
     - Like før kallet må inndata til funksjonen puttes i en eller flere globale variable
     - Når funksjonen har beregnet resultat puttes det i en annen global variabel, programmet kan da bruke denne verdien videre
  3. Via parametre (inndata til funksjon) og returverdi (resultat)

Hvorfor trenger vi dette?

- Slippe å skrive om igjen kode som allerede er laget!
- Sentralt problem:
  - funksjoner må kommunisere med resten av programmet
- Tre mulige løsninger
  1. Ingen vesentlig kommunikasjon
     - Funksjonen leser selv sine inndata med input ( )
     - Funksjonen skriver selv ut resultat med print( )
  2. Via globale variable **DÅRLIG**
     - Like før kallet må inndata til funksjonen puttes i en eller flere globale variable
     - Når funksjonen har beregnet resultat puttes det i en annen global variabel, programmet kan da bruke denne verdien videre
  3. Via parametre (inndata til funksjon) og returverdi (resultat)

Eksempel på dårlighet og godhet

- Vi vil lage en funksjon som beregner absoluttverdi:
  - |x| = x hvis x>=0, |x| = −x hvis x<0
  - Trenger egentlig ikke lage, abs() fins innebygget…
  - …men bare for illustrasjon
- Vi trenger f.eks en slik funksjon i forbindelse med statistisk analyse av lang liste med svar fra en spørreundersøkelse

- Dårlig løsning 1: input og print i funksjonen
- Dårlig løsning 2: globale variable
- Bedre løsning: parametre og return

kode: abso_v0.py …v1 …v2

Intro til returverdifunksjoner: Generering av tilfeldige tall

Kapittel 5.7

Standard Library-funksjoner og import-uttrykket (forts.)

- En funksjon i et bibliotek ≈ en svart boks
  - vi kan bruke funksjonen uten å se inne i boksen
  - vi har allerede brukt flere standardfunksjoner i Python uten å vite hvordan de ser ut innvendig, for eksempel
    - print( )
    - input( )
    - range( )
    - capitalize( )

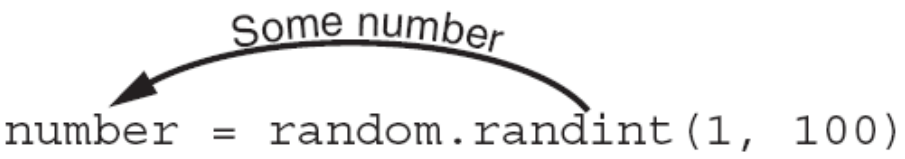
Å generere tilfeldige tall

- Tilfeldig genererte tall, mulige bruksområder
  - Spill og simuleringer
  - Statistiske analyser
- random-modulen:
  - biblioteksfunksjoner for tilfeldige tall
- «Dot notation»: kalle en moduls funksjoner:
  - Format: module_name . function_name()
  - Eksempler:
    - random.random() # tilfeldig flyttall [0.0,1.0>
    - random.uniform(fra,til) #tilfeldig flyttall [fra, til>
    - random.randint(fra,til) # tilfeldig heltall [fra,til]
    - random.randrange(fra,til,steg) # velger ett tall fra serien [fra, fra+steg, fra+2*steg, …, til>

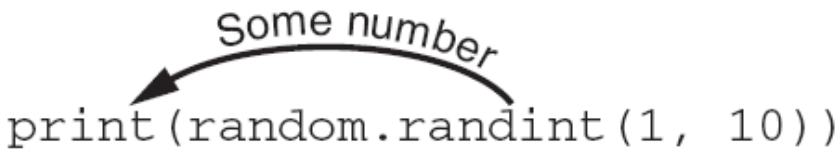
Å generere tilfeldige tall (forts.)

Retur av verdi fra random-funksjon:

Vise et tilfeldig tall i konsollet vha. random-funksjon:



A random number in the range of 1 through 100 will be assigned to the `number` variable.



A random number in the range of 1 through 10 will be displayed.

Seeds for tilfeldige tall

- Random-funksjonene er *pseudo-tilfeldige* tall
  - Liksom-tilfeldighet, maskinen gjør aldri noe helt tilfeldig
  - Mål: så likt en tilfeldig tallrekke som mulig
    - Realistiske simuleringer / statistiske analyser
    - Uforutsigbarhet i lotterier / pengespill
    - Variasjon i dataspill
- *Seed*-verdi: initialiserer formelen som genererer de tilfeldige tallene
  - Forskjellige *seeds* gir forskjellige serier med ”tilfeldige” tall
    - *Default*-verdien for seeds er systemtiden
    - funksjonen `random.seed()` kan sette *seed*-verdi hvis du ikke er fornøyd med å bruke tiden

Skrive egne funksjoner med returverdi

Kapittel 5.8

IPO-kart

- Verktøy for å designe og dokumentere funksjoner
  - Inndata
  - Prosessering
  - Output
- Typisk presentert i kolonner
  - Korte beskrivelser uten å gå i detalj
  - Inkluderer ofte nok informasjon til å erstatte et flytdiagram
- Annet alternativ: kommentarer i koden

  # INPUT: …

  # PROCESSING:

  # OUTPUT:

  plassert like under funksjonshodet

Eksempel på IPO-kart

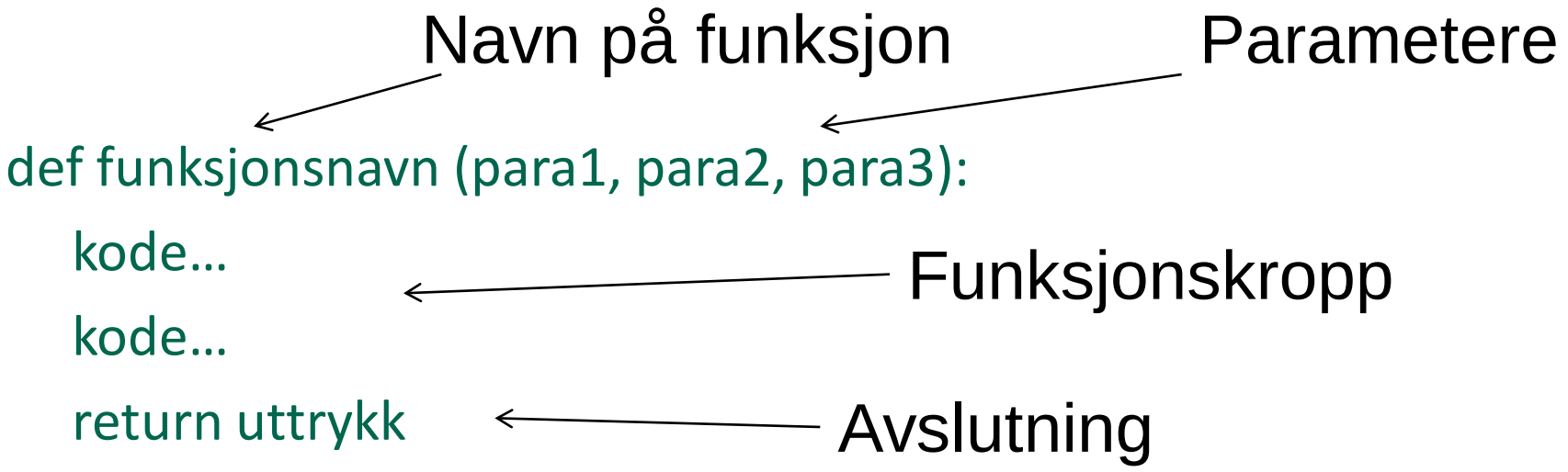
IPO Chart for the `discount` Function		
Input	Processing	Output
An item's regular price	Calculates an item's discount by multiplying the regular price by the global constant `DISCOUNT_PERCENTAGE`	The item's discount

```
DISCOUNT_PERCENTAGE = 0.20

def discount (price):
    discount = price * DISCOUNT_PERCENTAGE
    return discount
```

Oppbygning av en funksjon.

- Når man definerer/lager en funksjon bruker man det reserverte ordet def
- All koden i funksjonen (kroppen) skrives i *innrykk*.
- Avslutter med return (når noe skal returneres)



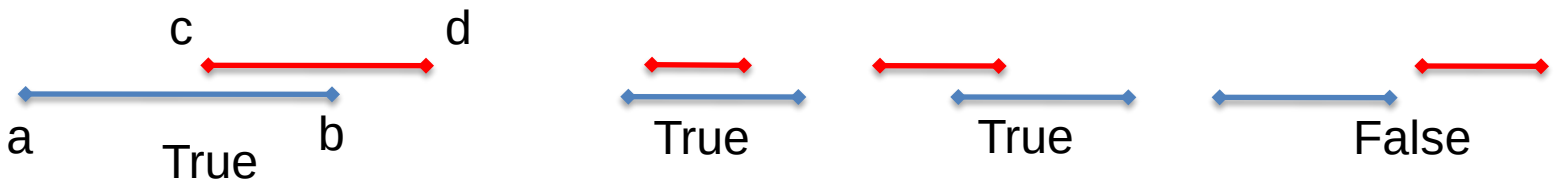
Å returnere *boolske verdier*

- Boolsk funksjon: returnerer enten True eller False
- Typiske Bruksområder
  - teste et kriterium for beslutninger (if…, while…)
  - forenkle sjekk av inndata for gyldighet
    - spørre om verdier
    - returnere **True** hvis inndata er en lovlig verdi

Oppgave: Boolsk funksjon

- Lag funksjonen overlapp_intervall(a,b,c,d)
  - Gitt intervallene (a,b) og (c,d): overlapper de hverandre?
  - Returnere True hvis overlapp, ellers False
  - Om ett begynner der det andre slutter: ikke overlapp
- Bruk parametre og returverdi
  - IKKE global variabel, IKKE input og print i funksjonen

kode: overlapp_intervall_v1.py …v2 …v3



Returnere flere verdier

- I Python kan en funksjon returnere flere verdier
  - Spesifisert etter return-uttrykket, atskilt av kommaer
    - Format: return uttrykk1, uttrykk2, etc.
  - Antall variable i kallet må stemme, dvs.
    - En variabel på venstresiden av = for hver returnert verdi:

```
def get_name():
    first_name=input("First name? ")
    last_name=input("Last name? ")
    return  first_name,  last_name

first_name,  last_name = get_name()
```

Modulen math

- Modulen math: inneholder
  - Funksjoner for å utføre matematiske beregninger
  - Matematiske konstanter som pi og e
  - Bruk av modulen:
    - må ha skrevet import math
      - Eller: from math import *funksjonsnavn* # de funksjonene vi trenger
      - Eller: from math import * # alle funksjoner
    - Kalle funksjon, for eksempel y = math.sin(x)
    - Bruke konstant, for eksempel omkr = 2 * math.pi * r
    - Vi kan slippe å skrive math. foran sin og pi hvis vi brukte
      - Enten from math import sin, pi
      - Eller from math import *

Noen funksjoner i modulen math:

`math` Module Function	Description
`acos(x)`	Returns the arc cosine of `x`, in radians.
`asin(x)`	Returns the arc sine of `x`, in radians.
`atan(x)`	Returns the arc tangent of `x`, in radians.
`ceil(x)`	Returns the smallest integer that is greater than or equal to `x`.
`cos(x)`	Returns the cosine of `x` in radians.
`degrees(x)`	Assuming `x` is an angle in radians, the function returns the angle converted to degrees.
`exp(x)`	Returns e^x
`floor(x)`	Returns the largest integer that is less than or equal to `x`.
`hypot(x, y)`	Returns the length of a hypotenuse that extends from (0, 0) to (`x`, `y`).
`log(x)`	Returns the natural logarithm of `x`.
`log10(x)`	Returns the base-10 logarithm of `x`.
`radians(x)`	Assuming `x` is an angle in degrees, the function returns the angle converted to radians.
`sin(x)`	Returns the sine of `x` in radians.
`sqrt(x)`	Returns the square root of `x`.
`tan(x)`	Returns the tangent of `x` in radians.

Lagre funksjoner i moduler

Kapittel 5.10

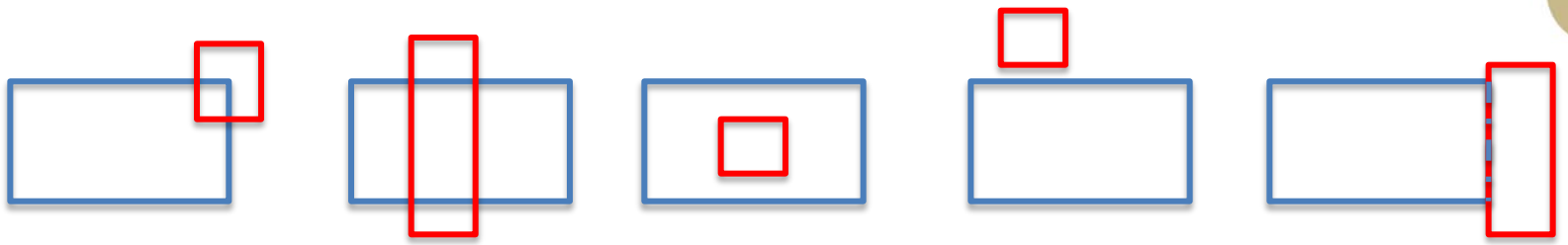
Å lagre funksjoner i moduler

- Modul : fil som inneholder Python-funksjoner
  - Programmer kan importere modulene og kalle funksjonene
    - import modulnavn
- Filnavn for modul skal slutte med .py
  - Ikke nøkkeluttrykk i Python (~~if.py, while.py~~, …)
- Hvorfor moduler?: gruppere relaterte funksjoner
  - program blir lettere å forstå, teste og vedlikeholde
  - nyttige funksjoner kan brukes i mange program
  - EKSEMPEL: overlappende_intervall() som vi nettopp lagde
    - stor generell bruksverdi
    - La oss putte den i en modul!

kode: intervall.py

bruk: booking.py

Bruk av overlapp_intervall()
Overlappende rektangler



- Gitt to rektangler parallelle med x- og y-aksen: Overlapper de?
- Lag en funksjon som sjekker dette!
  - 8 parametre, x1,y1, x2,y2, x3,y3, x4,y4
    - (x1,y1) og (x2,y2) er koordinatene for nedre venstre og øvre høyre hjørne i ett rektangel
    - (x3,y3) og (x4,y4) er tilsvarende koordinater for det andre rektanglet
- Dette er ganske tricky!
  - Ligner på oppgave 2d, eksamen i TDT4102 (C++), juni 2015
  - Potensielt jobbintervjuspørsmål…
- Blir vesentlig enklere med gjenbruk av **overlapp_intervall( )**
  - Bare vi skjønner hvordan vi skal bruke den…

kode: overlapp_rektangel_halvferdig.py
…_ferdig.py ; …_import.py

Oppsummering (1)

- Dette kapittelet dekket:
  - Returverdifunksjoner, inkludert
    - Å skrive returverdifunksjoner
    - Å bruke returverdifunksjoner
    - Å returnere flere verdier fra en funksjon
  - Å bruke bibliotekfunksjoner og import-uttrykket
  - Moduler, inkludert
    - Modulene random og math
    - Å gruppere dine egne funksjoner i moduler

Oppsummering (2), mange slags funksjoner

	Innebygde	I bibliotek	Egendefinerte
Uten returverdi - eksempel:…	print( )	color()	abso_v0()
- hvordan kalle? Tilsvarer hel setning	print('Hei!')	turtle.color('red')	abso_v0()
Med returverdi - eksempel: …	abs() input()	random() sin()	abso_v2()
- hvordan kalle? Tilsvarer en verdi	a=input('Tall? ') print(abs(a))	x=random.random() y=math.sin(x)/2	x=abso_v2(-5.1)
Hva må skrives lenger oppe i programmet?	ingenting	import turtle import random import math eller… from … import * (slippe prefiks ved kall)	def funknavn(par): setninger …. #hvis returverdi **return** verdi

Neste uke: Lister og tupler (kap 7)

- Aktuelle spørsmål for quiz:
  - «What will the following code display?» (Checkpoint 7.1)
  - «What will the following code display?» (Checkpoint 7.2)
  - «What will the following code display?» (Checkpoint 7.3)
  - «What will the following code display?» (Checkpoint 7.4)
  - «What will the following code display?» (Checkpoint 7.5)
  - «What will the following code display?» (Checkpoint 7.6)
  - «What will the following code display?» (Checkpoint 7.8)
  - «What will the following code display?» (Checkpoint 7.11)
  - «What will the following code display?» (Checkpoint 7.14)
  - «Give two reasons why tuples exist» (Checkpoint 7.23)
  - «This is the last index in a list» (Review, Multiple Choice 4)
  - «Which of the following statements create a tuple» (Review, Multiple Choice 14)
  - «What will the following code display?» (Review, Short Answer 1)
  - «What does the following code display?» (Review, Short Answer 5)
  - «What will the following code print?» (Algorithm Workbench 7)
- Noen av disse gis i «kahootisert form»
- Pluss 1-2 helt uannonserte spørsmål, men også om funksjoner