Python, del 1

Daniel Bosk[1]

Avdelningen för informations- och kommunikationssytem (IKS),
Mittuniversitetet, Sundsvall.

python1.tex 1343 2013-09-25 16:49:17Z danbos

Översikt

Översikt

Mittuniversitetet
MID SWEDEN UNIVERSITY

Vad är programmering?

Ett enkelt exempel

```
1 print( "hello␣world!" )
2 name = input( "who␣are␣you?␣" )
3 print( "hello␣" + name + ",␣how␣nice␣to␣meet␣
    you!" )
```

Översikt

Datatyper

- Man kan säga att nästan allt i Python är av någon datatyp.
- Dessa datatyper håller Python själv reda på, men man måste även som programmerare hålla koll på dessa för att programmet ska fungera som man vill.
- De datatyper som Python definierar är bl.a.

int	Heltal (Integers),
float	Flyttal (Floating point numbers),
str	Strängar (Strings).

Aritmetiska operatorer

- På dessa datatyper finns operatorer definierade, t.ex. + är en operator definierad för heltal.
- Andra operatorer som finns är `+ - * / // %`.

Aritmetiska operatorer

Några exempel (som körs i Pythons tolk från terminalen):

```
1  \$ python
2  Python 2.2.3 (#1, Jan  5 2005, 16:36:30)
3  [GCC 3.4.2] on sunos5
4  Type "help", "copyright", "credits" or
       "license" for more information.
5  >>> 5+5
6  10
7  >>> 3*2
8  6
9  >>> 5//3
10 1
11 >>> 5\%3
12 2
13 >>> "hej"+"svejs"
14 'hejsvejs'
15 >>> "hej"*3
16 'hejhejhej'
17 >>>
```

Aritmetiska operatorer

Alla operatorer är dock inte definierade för alla datatyper, en mycket kort och ej fullständig sammanfattning är:

Heltal	kan använda operatorerna + - * / // %,
Flyttal	kan använda operatorerna + - * /, heltalsdivision och modulo går inte att beräkna för flyttal.
Strängar	har bara operatorerna + *.

Identifierare

- Vilka regler gäller då för namnet på konstanterna?
- De får bestå av bokstäver, siffror och även understreck (_).
- Men de får dock inte börja med siffror.
- De får heller inte vara något av följande reserverade ord: and assert break class continue def del elif else except exec finally for from global if import in is lambda not or pass print raise return try while
- Viktigt att tänka på är att man skiljer på gemener och versaler, d.v.s. `PI` är inte samma sak som `Pi`.

Variabler

- En variabel är en identifierare.
- Den representerar ett ett minnesutrymme i vilket man kan lagra data.
- Namnet uppfyller kravet för en identifierare, men får inte vara enbart versaler (för det tolkar Python som en konstant).

Variabler

```
>>> x=5
>>> y=3
>>> z=x*y*3
>>> print( x, y, z )
5 3 45
>>> print( x )
5
>>> print( y )
3
>>> print( z )
45
>>> x=x+1
>>> print( x )
6
>>> print( z )
45
>>>
```

Variabler

- Notera att värdet på `z` inte ändras när vi ändrar värdet på `x`.
- Detta för att det är värdet `45` som lagras i `z` och inte relationen `x*y*3`.

Typkonvertering

- Ibland kan man vilja konvertera vissa typer till andra.
- Det kan vara att man läst in en sträng från tangentbordet och vill konvertera den till ett tag (om användaren matade in ett tal).

Typkonvertering

```
>>> x="3.14"
>>> pi=float(x)
>>> print( pi )
3.14
>>> print( x+2 )
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
  TypeError: cannot concatenate 'str' and 'int'
     objects
>>> print( pi+2 )
5.14
>>>
```

Typkonvertering

Ett annat bra exempel är procentberäkningar,

```
1 >>> 99/100*100
2 0
3 >>> float(99)/100*100
4 99.0
5 >>>
```

där man får fel svar om man inte explicit typkonverterar.

Översikt

Översikt

- Funktioner utgör en viktig byggsten i programmeringen.
- Liksom inom matematiken kan den användas för att dela upp större problem i mindre och bidra till en bättre ordning.
- De används dessutom på samma sätt.
- $f(x) = 2 \cdot g(x) + 3$ och $g(x) = x^2$ inom matematiken skulle kunna skrivas i Python som

```
1 def g(x):
2     return x*x
3
4 def f(x):
5     return 2*g(x)+3
```

Översikt

- När vi talar om funktioner består de av flera delar, funktionen består av ett funktionshuvud och en funktionskropp.
- Funktionshuvudet är den första raden i funktionsdefinitionen.
- Den består av funktionsnamn och formella parametrar. D.v.s.

$$\overbrace{\text{def functionname}(\underbrace{\text{argument1, argument2, ...}}_{\text{formella parametrar}}):}^{\text{funktionshuvud}}$$

- Resten av funktionen är funktionskroppen, d.v.s. all kod som hör funktionen till.

Översikt

Beräkning av energi hos en boll I

```
#encoding: utf8

import math

def kinetic_energy(m,v):
    return 0.5*m*math.pow(v,2)

def potential_energy(m,h):
    G = 9.82
    return m*G*h

def skriv(e1, e2):
    print "Bollens rörelseenergi = ", e1,
        "Joule"  # alt + str(e1)
    print "Bollens potentiella energi", e2,
        "Joule"
```

Beräkning av energi hos en boll II

```
svar = raw_input('Ange␣bollens␣massa:␣')
m = float(svar)

svar = raw_input('Ange␣bollens␣höjd:␣')
h = float(svar)

svar = raw_input('Ange␣bollens␣hastighet:␣')
v = float(svar)

e1 = kinetic_energy(m,v);
e2 = potential_energy(m,h);

skriv(e1,e2)
```

Beräkning av energi hos en boll

En körning av detta program (från en terminal) kan se ut så här:

```
\$ python 01.py
Ange bollens massa: 2
Ange bollens höjd: 3
Ange bollens hastighet: 4
Bollens rörelseenergi =  16.0 Joule
Bollens potentiella energi 58.92 Joule
\$
```

Tips och kommentarer

Några punkter att tänka på:

- Blanda aldrig språk, antingen skriver ni alla variabelnamn och funktioner på engelska eller på svenska – aldrig båda!
- Använd förklarande namn till alla variabler och funktioner!
- Skriv kommentarer!
- Dela upp programmen i mindre delar – funktioner! Det blir lättare att följa och det blir snyggare kod, men framför allt mycket enklare att programmera!

Referenser