Föreläsning 2

DD1310
Programmering / OPEN
6 hp

Datatyper

Heltal (int):
- Lagrar godtyckligt stort helttal.
- Binära talsystemet.
- T ex 8 bitars heltal ger 2^8 = 256 kombinationer => -128 till +127

Flyttal (float):
- Lagras med mantissa och exponent.
- T ex 5.9722x10^24

Boolean (bool):
- En boolesk variablen kan endast anta värdena True och False.
- True kan betraktas som 1 och False som 0.
- Det är en 1-bit variabel.

Sträng (str):
- Är en följd av godtyckliga tecken, även siffertecken.
- T ex ”Laboration 2”

Pythonspråket

- Datatyper
- Referenser
- Konstanter
- Funktioner
- Typkonverteringar
- Inläsning

Referenser

- Om man deklarararerar en variabel av en godtycklig datatyp t ex
  - *program = ‘OPEN’*
- så kommer *program* att vara en **referens** till en position i datorns minne där ett **objekt** med strängen ‘OPEN’ finns lagrad.
- Det är ointressant var objektet lagras men kom ihåg detta! (Bild)
- Till objektet finns ett antal **metoder** associerade, vilka dessa är bestäms av objektets datatyp (sträng, int, float, etc) .
- Exempel:
  - *kurs.upper()*
  - *kurs.capitalize()*
- En variabel kan tilldelas värdet *None* om man vill att den ska ha ett värde som är tomt:
  - *kurs = None*

Datatyper

- Varje värde som lagras i en variabel tillhör en för python förståelig datatyp. Vad som lagras i en variabel avgör datatypen, för att ta reda på en variabels datatype kan man skriva
  type(variabelnamn)
- Python har bl a följande fördefinierade datatyper:
  - Heltal: *int*
  - Flyttal: *float*
  - Boolesk: *bool*
  - String: *str*

Konstanter

- Variabler kan tilldelas ett startvärde i deklarationen:
  pi = 3.14159
- Variabelns värde kan ändras senare i programmet.
- För att skilja variabler från konstanter kan man skriva dem med versaler:
  PI = 3.14159

Funktioner

- Vissa beräkningar behöver man göra ofta i ett program. Istället för att skriva satserna som utför beräkningen flera gånger kan man definiera en funktion som gör beräkningen. Dessa liknar matematikens funktioner men kan hantera mycket mer än bara tal.
- Funktionens första rad kallas **funktionshuvud** och resten för **funktionkropp**.
- I funktionshuvudet anger man det reserverade ordet *def* följt av funktionens namn, eventuella indata som funktionen är beroende av och slutligen ett kolon:
  - *def funktionsnamn(eventuella parametrar):*
  - *T ex def cirkelarea(r):*
- Parametrarna i funktionshuvudet kallas **formella parametrar** och de man anger vid anropet kallas **anropsparametrar**.
- Ska något returneras måste en *return*-sats ingå i funktionskroppen.
- Samtliga variabler som införs i en funktion är **lokala**, vilket innebär att de bara existerar inuti funktionen.

Inläsning

- Användaren kan ge en variabel ett nytt värde om programmet läser in detta från tangentbordet.
  - input()
    - Läser in en rad som en sträng, parameter kan ges. Eventuell typkonvertering görs efteråt.
- Programkörningen haltar vid anrop av input() och fortsätter först då returtangenten tryckts. Den eventuella sträng man knappat in före returtryckningen returneras.

Exempel

```
# Funktioner

# import math
from math import pi

def cirkelarea(r):
    # pi = 3.14
    return pi*r*r


print ('Cirkelns area =', cirkelarea(2))
print ('pi = ', pi)
```

Exempel

```
# Inläsning

pi = 3.14

def cirkelarea(r):
    return pi*r*r

namn = input('Vad heter du? ')
print ('Hej', namn)

r = input('Vad är cirkelns radie? ')
print ('Cirkelns area = ', cirkelarea(float(r)))
print ('pi = ', pi)
```

Typkonverteringar

- Används när man vill ändra datatypen för ett uttryck.
- Exempel:
  - *s = '2.75'*
  - *f = float(s)*
- *i* får här värdet 2.75
- Fungerar bara för rimliga konverteringar.
- *int(), float(), str()*