Skolan för Datavetenskap och kommunikation

DD1311 Programmeringsteknik med PBL

Föreläsning 5

Funktioner (kap 6)

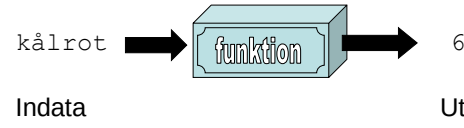
- Kap 6 i Dawson
- Funktioner du redan använt
- Anropa funktioner
- Definiera egna funktioner
- Parameter & returvärde
- Lokala och globala variabler
- Rekursion

Funktioner du redan använt

Funktion	Indata (parametrar)	Utdata (returvärde)
`ut=input("Gissa:")`	strängen "Gissa"	den inlästa gissningen
`ut=str(bredd)`	talet i variabeln bredd	talet som sträng
`ut=len("kålrot")`	strängen "kålrot"	antal tecken i strängen
`ut=range(0,100,2)`	startvärde 0, övre gräns 100, steg 2	en lista med tal

Anropa funktioner

- Så här ser anrop ut: `utdata = funktion(indata)`
- Indata skickas in via *parametrar* till funktionen
- Utdata returneras via *return-sats* ur funktionen
- Programmet *fortsätter* efter anropet



Snigelexempel

- http://www.csc.kth.se/utbildning/kth/kurser/DD1311/forel11/snigelrace.py

Hur man definierar en funktion

- Funktioner definieras överst i programmet!
- Skriv först
  `def` *funktionsnamn*`(`*parametrar*`):`
- Sen, indenterat:
  - En kommentarrad som beskriver vad funktionen gör, inom tredubbla citationstecken, tex `"""Beräknar arean"""`.
  - Satserna som funktionen ska utföra.
  - Allra sist `return` *returvärde/returvärden*
  - Anger man inget returvärde blir det `None`

Parameter & returvärde

Funktionen definieras så här:

```
def renta(pengar):
    """Beräknar och returnerar räntan."""
    if pengar > 100000:
        r = pengar*0.75/100
    else:
        r = pengar*0.40/100
    return r
```

returvärde `r`

parameter `pengar`

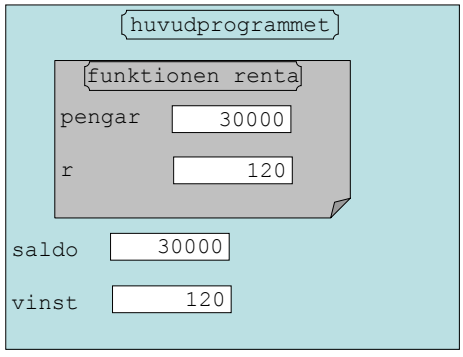
Funktionen anropas så här:

```
vinst = renta(saldo)
```

Uppgift: Du vill skriva en funktion som avgör om en låneansökan ska beviljas.
- Vad är indata (parametrar)?
- Vad är utdata (returvärden)?

Lokala och globala variabler

- Varje gång en funktion anropas reserveras minne för just det anropet.
- Parametrar och variabler som definieras inuti funktionen blir *lokala,* dvs dom finns bara i funktionen.
- Variabler som definierats i *huvudprogrammet* kallas *globala* variabler.
- En funktion som ska förändra en variabels värde bör *returnera* det ändrade värdet.



Början till Labb 3

```
#Skapar en dictionary med mat på två språk
svenska = "spenat/gröt/kött/fil/pumpa"
setswana = "morogo/bogobe/nama/madila/lerotse"
svenskaOrd = svenska.split("/")
setswanaOrd = setswana.split("/")
ordlista = {}
antal = len(svenskaOrd)
for i in range(antal):
    s = setswanaOrd[i]
    ordlista[s] = svenskaOrd[i]
print(ordlista)
```

Labb 3: funktion 1

```
def skapaLista(rad):
    """Delar upp en rad i ord"""
    lista = rad.split("/")
    return lista
```

Labb 3: funktion 2

```
def flytta(uppslagsord,definition):
    """Skapar en ordlista"""
    ordlista = {}
    antal = len(uppslagsord)
    for i in range(antal):
       ordlista[uppslagsord[i]] = definition[i]
    return ordlista
```

Labb 3: huvudprogrammet

```
#Huvudprogram
svenska = "spenat/gröt/kött/fil/pumpa"
setswana = "morogo/bogobe/nama/madila/lerotse"
svenskaOrd = skapaLista(svenska)
setswanaOrd = skapaLista(setswana)
ordlista  = flytta(svenskaOrd,setswanaOrd)
print ordlista
```

Rekursion

- En rekursiv funktion i Python har:
  - Ett anrop av funktionen själv inuti funktionsdefinitionen.
  - Ett basfall som inte är rekursivt.
  - En if-sats som väljer mellan basfallet och det rekursiva anropet.
- Använd bara rekursion om du vet vad du gör!

Rekursivt exempel

```
def dubbel(x):
    if x<4:
        print x
        dubbel(x+1)
        print x, "igen!"

dubbel(1)
```