Skolan för Datavetenskap och kommunikation

DD1311 Programmeringsteknik med PBL

Föreläsning 3

- Kap 4 i Dawson
- Slumptal
  - fler random-funktioner
- En ny *styrstruktur*:
  - for-slingan
- Mer om strängar:
  - metoden split()
  - index

Talgissning 3.1

```
# Talgissning, version 3.1
import random
tal = random.randint(1,100)
gissning = int(input("Gissa mitt tal: "))
n = 1
while gissning != tal:
    if gissning > tal:
        gissning = int(input("Lägre:"))
    elif gissning < tal:
        gissning = int(input("Högre:"))
    n += 1
print("Bravo, du gissade rätt!")
print("...på bara", n, "försök!")
```

Random-funktioner

Funktion	Resultat
`random.randint(1,6)`	Slumpar ett heltal: 1,2,3,4,5 eller 6
`random.randrange(1,6,2)`	Slumpar ett udda heltal: 1,3 eller 5
`random.random()`	Slumpar ett decimaltal mellan 0.0 och 0.999…
`namn="Kit","Nour","Robin"` `random.choice(namn)`	Slumpar ett av namnen

Talföljder

Funktionen range() ger en lista av heltal.
`range(10)` ger `[0,1,2,3,4,5,6,7,8,9]`

Man kan också ange både start och slut:
`range(8,15)` ger `[8,9,10,11,12,13,14]`

Och även steg:
`range(44,55,3)` ger `[44,47,50,53]`

for-slingan

En for-slinga upprepar ett antal satser för varje element i en följd. Exempel:

- För varje tal i en följd (*range*):
  - Beräkna kvadraten!
- För varje bokstav i en sträng:
  - Skriv ut bokstaven!
- För varje telefonnummer på listan:
  - Ring upp personen!

följd=sequence

Vad blir ditt namn baklänges?

```
# Vänder namnet baklänges
namn = "tunström"
bakfram = ""
for bokst in namn:
    bakfram = bokst + bakfram
print(bakfram)
```

Hur fungerar det?

bokst	bakfram
t	t
u	u+t
n	n+ut
s	s+nut
t	t+snut
r	r+tsnut
ö	ö+rtsnut
m	m+örtsnut

Uppgift: Vad händer här?

```
summa = 0
for i in range(5):
    summa += i
print(summa)
```

…Och här?

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

Strängmetoden split()

- Det är enkelt att dela upp en mening i ord:
  `mening.split()`
- Metoden split delar vid mellanslag
- Varje del läggs i en numrerad låda
- Numret kallas *index*

Strängar - index

- Varje tecken i en sträng har också index.
- Exempel:

`mat="pizza"`

p	i	z	z	a
0	1	2	3	4

mat[0]	mat[1]	mat[2]	mat[3]	mat[4]
"p"	"i"	"z"	"z"	"a"

Strängar: skivning

mat[0]➔"p"	Titta går bra…
~~mat[0] = "m"~~	...men inte ändra värdet.
mat[1:3]➔"iz"	Delsträng (3:e ingår inte).
mat[:4]➔"pizz"	Från början till 4.
mat[2:]➔"zza"	Från 2 till slutet.

strängskivning = string slicing

Algoritm för att skapa anagram

1. Börja med ett tomt anagram
2. Slumpa ett index i ursprungsordet
3. Kopiera bokstaven från den platsen och lägg till den till anagrammet
4. Plocka bort den från ursprungsordet
5. Upprepa punkt 2-4 så länge som det finns bokstäver kvar i ursprungsordet

Exempel

slumpad plats	namn = "LISA"	anagram = ""
1	"LSA"	"I"
2	"LS"	"IA"
1	"L"	"IAS"
0	""	"IASL"

Anagramprogrammet (Python 2)

```
import random
namn = raw_input("Vad heter du? ")

anagram =""
while namn != "":
    antal = len(namn)
    index = random.randrange(antal)
    anagram = anagram + namn[index]
    namn = namn[:index] + namn[(index+1):]
    print index, namn, anagram

print "Ett anagram på ditt namn är ", anagram
```

Anagramprogrammet (Python 3)

```
import random
namn = input("Vad heter du? ")

anagram =""
while namn != "":
    antal = len(namn)
    index = random.randrange(antal)
    anagram = anagram + namn[index]
    namn = namn[:index] + namn[(index+1):]
    print(index, namn, anagram)

print("Ett anagram på ditt namn är ", anagram)
```