Programmering i Python

	27 oktober	Grunder i Python
⇨	**10 november**	Slingor, funktioner
	17 november	Klasser, metoder
	1 december	Praktisk tillämpning
	8 december	?

Repetition

- int, float, string, list, tuple
- print()
- input()
- Variabeltilldelning
- if-satser

Mer om if-satser

```
if age < 12:
    print(“Du är för ung för att åka Jetline”)
elif age > 80:
    print(“Du är för gammal för att åka Jetline”)
else:
    print(“Välkommen in till Jetline”)
```

Kan lägga till hur många elif som helst, men bara en else för varje if-sats.

Kan ha en if-sats i en if-sats:

```
if name[0] == “E”:
    if name == “Elvira”:
        print(“Du är Elvira”)
    else:
        print(“Du är inte Elvira men ditt namn börjar på E”)
```

For-loopar på listor

Låt säga att vi vill utföra samma operation på alla element i en lista

Exempel:
Vi vill printa alla siffror i en lista på en varsin rad

```
lista = [1, 2, 3, 5, 6, 3, 2, 6]
```

Jobbigt att göra:

```
print(lista[0])
print(lista[1])
print(lista[2])
```

Osv… om det är en lång lista eller om vi inte vet längden på listan innan

Istället använder vi en “for”-loop:

```
for siffra in lista:
    print(siffra)
```

For-loopar på ett spann (range)

Låt säga att vi vill upprepa något ett visst antal gånger gånger, t.ex. Printa “Hej” 1000 gånger. Jobbigt om vi skulle behöva en 1000 element lång lista.
range(n) skapar ett spann som fungerar som listan [0, 1, …, n - 1]
Ex. `range(1000) = [0, 1, 2, …, …, 998, 999]`

```
for i in range(1000):
    print(“Hej nummer “ + str(i) )
```

Kommer printa:

```
Hej nummer 0
Hej nummer 1
Hej nummer 2
.
.
.
Hej nummer 999
```

For-loopar i for-loopar?

```
for i in range(5):
    print(i)
for i in range(5):
    print(i)
```

```
for i in range(5):
    for j in range(5):
        print(i)
```

Skillnad?

If-satser i for-loopar

Låt säga att vi har en lista med de 10 största städerna i Sverige (strängar)
och att vi vill printa alla städer som har mer än 8 bokstäver i sorterad ordning

Idé:

1. Sortera listan
2. Gör en for-loop som går igenom hela listan
    a. Printa staden om dess ordlängd är >8

```
cities = [“Norrköping”, “Stockholm”, “Malmö”, “Göteborg”, “Jönköping”,
“Uppsala”, “Helsingborg”, “Linköping”, “Örebro”, “Västerås”]
cities.sort()
for city in cities:
    If len(city) > 8:
        print(city)
```

While-loopar

Används om vi vill upprepa något medan ett villkor stämmer.
T.ex. Vi vill addera 1 till ett tal fram tills att det talet är större än 10.

```
tal = 1
while tal <= 10:
    tal = tal + 1
```

Vad hade hänt om vi skrev `while tal > 0` ?

Oändlig loop!

Moduler

Python har många inbyggda funktioner

Bra - man behöver inte uppfinna något som redan någon annan uppfunnit

Det finns även en hel del funktioner som man lätt kan importera till sitt pythonprogram

Random

Ibland vill man generera slumptal. För att göra det i python så måste man importera modulen random.

```
import random
```

Slumpvis tal mellan 0 och 10:

```
random.randrange(10)
```

Timer delay

Om man vill “pausa” sitt program, eller göra något annat med tid så finns det en modul för det med: time

```
import time
```

Vänta i en sekund:

```
time.sleep(1)
```

Funktioner

Om vi hinner

Uppgifter shorturl.at/grtCX

Psst.. x % 2 = 0 om x är Delbart med 2

1. Skriv ut alla jämna tal från 0 till och med 100

2. Addera alla jämna tal från 0 till och med 1000 och printa resultatet

3. Gör en nedräknare till en raketuppskjutning, som räknar/printar ned från 10 till 1 och sedan printar “LIFT OFF!” (använd sleep)

4. Skapa ett gissningsspel:
   a. Programmet genererar ett hemligt slumptal
   b. Användaren gissar på talet
   c. Programmet svarar om det hemliga talet är större eller mindre än gissningen
   d. Upprepa steg b och c tills användaren gissat rätt
   e. Fråga om du behöver hjälp!

5. Skapa sten sax påse där programmet väljer ett och användaren väljer ett (tips: representera sten sax påse som siffrorna 0, 1, 2 och låt programmet slumpa en). Printa tydliga instruktioner till användaren.