DD1331

Grundläggande programmering för F1

Några bilder till föreläsning 2

Innehåll

Innehåll

- Snabbrepetition av Fahrenheit-Celsius

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper,

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler,

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer:

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer: aritmetiska,

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer: aritmetiska, relationer,

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer: aritmetiska, relationer, logiska

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer: aritmetiska, relationer, logiska
- if – satser

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer: aritmetiska, relationer, logiska
- if – satser med else och elif

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer: aritmetiska, relationer, logiska
- if – satser med else och elif
- Repetition med while

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer: aritmetiska, relationer, logiska
- if – satser med else och elif
- Repetition med while
- Biblioteksmoduler:

Innehåll

- Snabbrepetition av Fahrenheit-Celsius
- Datatyper, variabler, tilldelning
- Operatorer: aritmetiska, relationer, logiska
- if – satser med else och elif
- Repetition med while
- Biblioteksmoduler: math, time, random

Fahrenheit-Celsius-programmet från F1

```
# Fran Fahrenheit till Celsius

print()

f = float(input("Fahrenheit = "))
c = (f-32)*5/9

print("Celsius = ",round(c,1))
print()
```

Pythons inbyggda datatyper

Pythons inbyggda datatyper

- Tal

  int, float, complex, bool

Pythons inbyggda datatyper

- Tal

  int, float, complex, bool

- Texter (Strängar)

  str

Pythons inbyggda datatyper

- Tal

  int, float, complex, bool

- Texter (Strängar)

  str

- Samlingar av ordnade element: tuple, list

Pythons inbyggda datatyper

- Tal

  int, float, complex, bool

- Texter (Strängar)

  str

- Samlingar av ordnade element: tuple, list

- Samlingar utan ordning: set, dictionary

Pythons inbyggda datatyper

- Tal

  int, float, complex, bool

- Texter (Strängar)

  str

- Samlingar av ordnade element: tuple, list

- Samlingar utan ordning: set, dictionary

Blått: NU Grått: SENARE

Pythons inbyggda datatyper

- Tal

  int, float, complex, bool

- Texter (Strängar)

  str

- Samlingar av ordnade element: tuple, list

- Samlingar utan ordning: set, dictionary

Blått: NU Grått: SENARE

Variabel = namn som refererar till data

Pythons inbyggda datatyper

- Tal

  int, float, complex, bool

- Texter (Strängar)

  str

- Samlingar av ordnade element: tuple, list

- Samlingar utan ordning: set, dictionary

Blått: NU Grått: SENARE

Variabel = namn som refererar till data

Typ är en egenskap hos data, ej hos variabler.

Aritmetiska operatorer

Aritmetiska operatorer

$a = 5 \quad b = 3$

Aritmetiska operatorer

a = 5 b = 3

Operator	Beskrivning	Exempel
+	addition	`a+b` → 8
-	subtraktion	`a-b` → 2
*	multiplikation	`a*b` → 15
/	division	`a/b` → 1.6666666666666667
%	modulo, resten	`a%b` → 2
//	*floor division*	`a//b` → 1
**	exponentiering	`a**b` → 125

Aritmetiska operatorer

a = 5 b = 3

Operator	Beskrivning	Exempel
+	addition	`a+b` → 8
-	subtraktion	`a-b` → 2
*	multiplikation	`a*b` → 15
/	division	`a/b` → 1.6666666666666667
%	modulo, resten	`a%b` → 2
//	*floor division*	`a//b` → 1
**	exponentiering	`a**b` → 125

Alla aritmetiska operatorer kan kombineras med tilldelning, t.ex.

Aritmetiska operatorer

a = 5 b = 3

Operator	Beskrivning	Exempel
+	addition	a+b $\rightarrow$ 8
-	subtraktion	a-b $\rightarrow$ 2
*	multiplikation	a*b $\rightarrow$ 15
/	division	a/b $\rightarrow$ 1.6666666666666667
%	modulo, resten	a%b $\rightarrow$ 2
//	*floor division*	a//b $\rightarrow$ 1
**	exponentiering	a**b $\rightarrow$ 125

Alla aritmetiska operatorer kan kombineras med tilldelning, t.ex.

```
a *= 7        ⇔    a = a*7
```

Aritmetiska operatorer

a = 5 b = 3

Operator	Beskrivning	Exempel
+	addition	`a+b` → 8
-	subtraktion	`a-b` → 2
*	multiplikation	`a*b` → 15
/	division	`a/b` → 1.6666666666666667
%	modulo, resten	`a%b` → 2
//	*floor division*	`a//b` → 1
**	exponentiering	`a**b` → 125

Alla aritmetiska operatorer kan kombineras med tilldelning, t.ex.

```
a *= 7            ⇔   a = a*7
antalBarn += 1    ⇔   antalBarn = antalBarn + 1
```

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

a = 5 b = 3

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

$a = 5 \quad b = 3$

```
a == b
```

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

a = 5 b = 3

`a == b` är **False**

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

a = 5 b = 3

`a == b` är **False**

`a > b`

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

a = 5 b = 3

`a == b` är **False**

`a > b` är **True**

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

a = 5 b = 3

`a == b` är **False**

`a > b` är **True**

`a != b`

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

a = 5 b = 3

`a == b` är **False**

`a > b` är **True**

`a != b` är **True**

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

a = 5 b = 3

`a == b` är **False**

`a > b` är **True**

`a != b` är **True**

```
svar = a >= 0
```

Relationsoperatorer

Operator	Beskrivning
==	lika med
!=	skilt från
>	större än
<	mindre än
>=	större än eller lika med
<=	mindre än eller lika med

a = 5 b = 3

`a == b` är **False**

`a > b` är **True**

`a != b` är **True**

`svar = a >= 0` svar blir **True**

Logiska operatorer kombinerar villkor

Operator	Beskrivning
`and`	sant om båda villkoren uppfyllda
`or`	sant om ett av villkoren uppfyllda
`not`	sant om villkoret EJ är uppfyllt

Logiska operatorer kombinerar villkor

Operator	Beskrivning
`and`	sant om båda villkoren uppfyllda
`or`	sant om ett av villkoren uppfyllda
`not`	sant om villkoret EJ är uppfyllt

```
kvadrant1 = x > 0 and y > 0

passed = labmarks > 100 or exammarks > 75

failed = not passed
```

Styrstrukturer

Styrstrukturer

- Villkor

Styrstrukturer

- Villkor
- Repetitioner

Styrstrukturer

- Villkor

  if

- Repetitioner

Styrstrukturer

- Villkor

  if if...else

- Repetitioner

Styrstrukturer

- Villkor

  if if...else if...elif...else

- Repetitioner

Styrstrukturer

- Villkor

  if if...else if...elif...else

- Repetitioner

  while

Styrstrukturer

- Villkor

  if if...else if...elif...else

- Repetitioner

  while for

Styrstrukturer

- Villkor

  if if...else if...elif...else

- Repetitioner

  while for

  med villkor

Styrstrukturer

- Villkor

  if if...else if...elif...else

- Repetitioner

  while for

  med villkor över sekvens

if-satsen

```
if villkor:
    block
```

if-satsen

```
if villkor:
    block
```

```
if villkor:
    block1
else:
    block2
```

if-satsen

```
if villkor:
    block
```

```
if villkor:
    block1
else:
    block2
```

Ett block består av en eller flera satser

if-satsen

```
if villkor:
    block
```

```
if villkor:
    block1
else:
    block2
```

Ett block består av en eller flera satser

Indenteringen (indraget) är nödvändig

if-satsen

```
if villkor:
    block
```

```
if villkor:
    block1
else:
    block2
```

Ett block består av en eller flera satser

Indenteringen (indraget) är nödvändig

Fler än två alternativ?

if-satsen

```
if villkor:
    block
```

```
if villkor:
    block1
else:
    block2
```

Ett block består av en eller flera satser

Indenteringen (indraget) är nödvändig

Fler än två alternativ? Använd **elif** !

if-satsen med **elif**

```
if villkor1:
    block1
elif villkor2:
    block2
else:
    block3
```

Flera **elif** – delar går bra

if-else-exempel

```
guess = int(input(
  "What year was Marie Curie born? "))

if guess == 1867:
    print("Correct")
else:
    print("Wrong answer")
```

if-elif-else-exempel

```
guess = int(input(
  "What year was Marie Curie born? "))

if guess == 1867:
    print("Correct")
elif abs(guess - 1867) <= 10:
    print("Close but not correct")
else:
    print("Entirely wrong")
```

if-elif-else-exempel

```
guess = int(input(
  "What year was Marie Curie born? "))

if guess == 1867:
    print("Correct")
elif abs(guess - 1867) <= 10:
    print("Close but not correct")
else:
    print("Entirely wrong")
```

Om vi har tid bygger vi ut exemplet på föreläsningen

if-else med sammansatt villkor

```
answer = \
int(input("What␣year␣did␣Marie␣Curie␣\
␣␣␣␣␣␣␣␣␣␣␣␣receive␣the␣Nobel␣Prize?␣"))

if answer == 1903 or answer == 1911:
    print("Correct")
else:
    print("Wrong")
```

while – repetition

while – repetition

```
while villkor:
    block
```

while – repetition

```
while villkor:
    block
```

Så länge som villkoret är sant,
utför satsen eller satserna i blocket!

while – repetition

```
while villkor:
    block
```

Så länge som villkoret är sant,
utför satsen eller satserna i blocket!

Vilken multipel av 17 är närmast större än 40?

while – repetition

```
while villkor:
    block
```

Så länge som villkoret är sant,
utför satsen eller satserna i blocket!

Vilken multipel av 17 är närmast större än 40?

```
x = 0
while x <= 40:
    x += 17

print("x beraknades till", x)
```

”Oändlig” repetition

```
while True:
    block
```

Avbryts med `Ctrl-C` eller break–sats inuti blocket

"Oändlig" repetition

```
while True:
    block
```

Avbryts med `Ctrl-C` eller break–sats inuti blocket

Uppgift:

- Upprepa frågan om Marie Curies födelseår, avbryt när rätt svar givits.

”Oändlig” repetition

```
while True:
    block
```

Avbryts med `Ctrl-C` eller break–sats inuti blocket

Uppgift:

- Upprepa frågan om Marie Curies födelseår, avbryt när rätt svar givits.
- Räkna också hur många försök som behövdes.

Lösning (några texter är utelämnade)

```
nGuess = 0
while True:
    guess = int(input( ....))
    nGuess += 1
    if birthyear == 1867:
        print("Correct.")
        break  # ut ur while-slingan
    elif abs(guess-1867) <= 10:
        print("Pretty close ...")
    else:
        print("Wrong")
```

Lösning (några texter är utelämnade)

```
nGuess = 0
while True:
    guess = int(input( ....))
    nGuess += 1
    if birthyear == 1867:
        print("Correct.")
        break  # ut ur while-slingan
    elif abs(guess-1867) <= 10:
        print("Pretty close ...")
    else:
        print("Wrong")

if nGuess == 1:
    print("Excellent")
else:
    print("You needed", nGuess, "tries")
```

Biblioteksmoduler

Biblioteksmoduler

- math

Biblioteksmoduler

- math
- time

Biblioteksmoduler

- math
- time
- random

Biblioteksmoduler

- math

  med sin, cos, sqrt, pi, e .....
- time

- random

Biblioteksmoduler

- math

  med sin, cos, sqrt, pi, e .....
- time

  t.ex. fördröjning, mätning av cpu-tid
- random

Biblioteksmoduler

- math

  med sin, cos, sqrt, pi, e .....
- time

  t.ex. fördröjning, mätning av cpu-tid
- random

  för att generera slumptal

Biblioteksmoduler

- math

  med sin, cos, sqrt, pi, e .....
- time

  t.ex. fördröjning, mätning av cpu-tid
- random

  för att generera slumptal

```
import math
v = math.sin(math.pi*0.37)
```

Biblioteksmoduler

- math

  med sin, cos, sqrt, pi, e .....
- time

  t.ex. fördröjning, mätning av cpu-tid
- random

  för att generera slumptal

```
import math
v = math.sin(math.pi*0.37)
```

eller

Biblioteksmoduler

- math

  med sin, cos, sqrt, pi, e .....
- time

  t.ex. fördröjning, mätning av cpu-tid
- random

  för att generera slumptal

```
import math
v = math.sin(math.pi*0.37)
```

eller

```
from math import *
v = sin(pi*0.37)
```

Exempel med time och random

Exempel med time och random

- time: Nedräkningsexempel:

Exempel med time och random

- time: Nedräkningsexempel:
  Skriv ut 10 9 8 7 6 5 4 3 2 1 0
  med 1 sekunds fördröjning efter varje siffra.
  Avsluta med GOOOOO!

Exempel med time och random

- time: Nedräkningsexempel:
  Skriv ut 10 9 8 7 6 5 4 3 2 1 0
  med 1 sekunds fördröjning efter varje siffra.
  Avsluta med GOOOOO!

- random: Tärningsexempel

Exempel med time och random

- time: Nedräkningsexempel:
  Skriv ut 10 9 8 7 6 5 4 3 2 1 0
  med 1 sekunds fördröjning efter varje siffra.
  Avsluta med GOOOOO!

- random: Tärningsexempel
  Simulera 1000 kast med två tärningar.
  Hur många gånger slås sexor med båda tärningarna?

Exempel med time och random

- time: Nedräkningsexempel:
  Skriv ut 10 9 8 7 6 5 4 3 2 1 0
  med 1 sekunds fördröjning efter varje siffra.
  Avsluta med GOOOOO!

- random: Tärningsexempel
  Simulera 1000 kast med två tärningar.
  Hur många gånger slås sexor med båda tärningarna?

Pythonkoden finns på kurshemsidan

Hitta information om biblioteksmoduler

Hitta information om biblioteksmoduler

- I Pythontolken, gör

Hitta information om biblioteksmoduler

- I Pythontolken, gör

```
import modulename
help(modulename)
```

Hitta information om biblioteksmoduler

- I Pythontolken, gör

```
import modulename
help(modulename)
```

- Python Standard Library

Hitta information om biblioteksmoduler

- I Pythontolken, gör

```
import modulename
help(modulename)
```

- Python Standard Library