



UPPSALA
UNIVERSITET

Programmeringsteknik I vt20

- Introduktionskurs i programmering
 - Med Python som språk





Kursmål

Efter godkänd kurs ska studenten kunna:

- förklara vad ett givet program i Python utför;
- använda befintliga moduler såsom Pythons standardbibliotek och numpy utifrån deras dokumentation och andra resurser;
- analysera och lösa problem med hjälp av programmeringskonstruktioner;
- använda en programutvecklingsmiljö;
- testa och felsöka program;
- redogöra för de grundläggande begreppen modul, funktion, klass, objekt och därtill hörande underbegrepp;

Vad är programmering?

- ”Att få datorn att göra det vi vill”
 - För att slippa låta människor göra det
- ”Att analysera och dela upp problem i väldefinierade, kända moment”
- Att skriva kod
- Att använda kod
- Att läsa kod
 - Kommunikation med *människor*, inte datorer
 - Den människan kan vara du själv om ett halvår... eller i morgon
- Att använda alla tillgängliga resurser för att uppnå detta
 - Läroböcker, referensdokumentation, Google, Stackoverflow
 - Men om något används är det viktigt att man ändå förstår hur och varför
 - Lätt att hitta ”smarta tips” som är fel eller används fel



Vad är Python?

- Ett generellt programmeringsspråk
 - Finns språk som bara är tänkta för ett användningsområde och främst finns där
 - MATLAB exempel på språk avsett för tekniska beräkningar
 - R för statistik
- Ett fritt programmeringsspråk
 - Mer än bara "gratis"
 - Pythonspecifikationen är öppet tillgänglig
 - Standardimplementationen av Python är öppen källkod
- Ett tolkat programmeringsspråk
 - Textkommandon kan köras direkt
 - Program måste inte byggas/kompileras innan de kan köras
 - Går (ofta) fort att skriva, långsammare att köra



Vad är Python?

- Ett populärt programmeringsspråk
 - Vetenskap: SciPy, NumPy, Matplotlib, OpenCV
 - AI: PyTorch, TensorFlow
 - Skapande: InkScape (vektorgrafik), OpenShot (videoredigering), Blender (3D)
 - Spel (vissa delar): bl.a. Sims 4
 - Webbapplikationer: Django, Bittorrent
- Det finns färdiga Pythonpaket för "det mesta" som du kan använda i dina program!
 - Skapa/läsa bilder, filmer, Excelfiler är t.ex. inga problem



Upplägg

- Programmering är en färdighet
 - Finns naturligtvis en omfattande teori
 - (Nya) begrepp
 - Gamla begrepp som får ny innebörd jämfört med
 - Matematik
 - Vardagsspråk
- 7 föreläsningar ger ett stöd
- ~30 labbtillfällen är kärnan (3 hp)
- Skriftlig tentamen 20 mars (2 hp)



Labbar

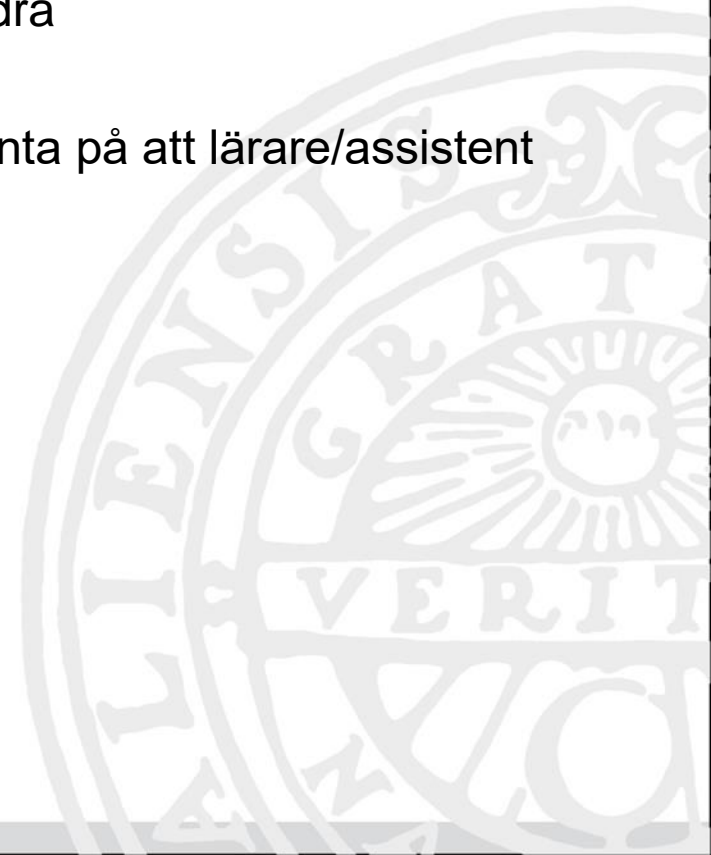
- *Alla* labbar är frivilliga
- Under labbarna arbetar ni med 10 *nätlektioner*
 - Fem av dem är obligatoriska uppgifter (OU) som ska redovisas muntligen
 - På ett pass när den grupp du tillhör är schemalagd
 - Deadline för varje OU
 - Exakt datum beror på när *din* labbgrupp är schemalagd
 - Läser du fristående? Välj själv.



UPPSALA
UNIVERSITET

Labbar

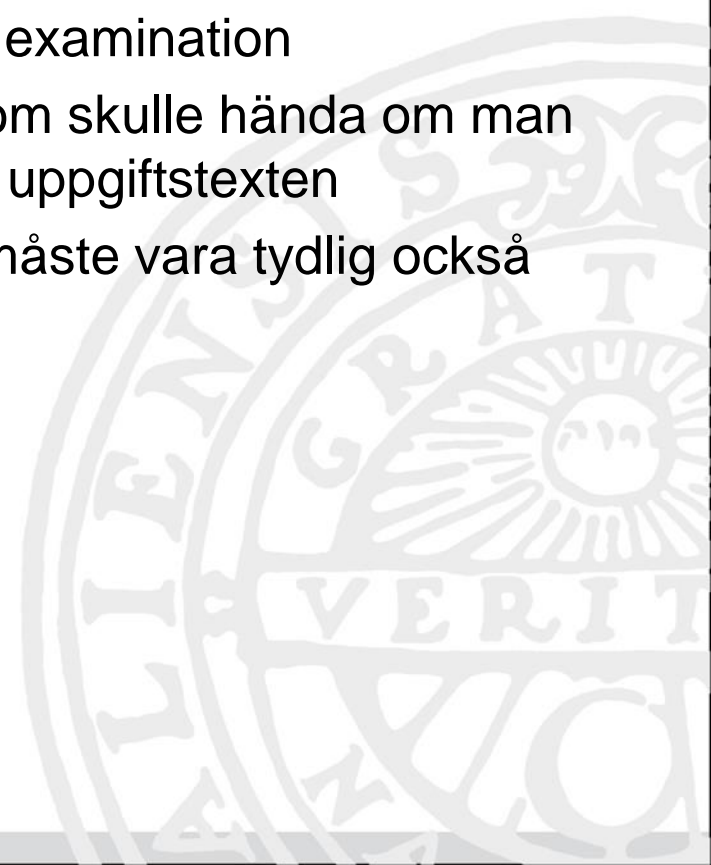
- Se till att arbeta själv med datorn
 - Egen laptop eller terminalen i datorsalen
- Helt okej att ta hjälp och diskutera med varandra
- Kör du fast?
 - Skriv upp ditt namn på lista "Hjälp" och vänta på att lärare/assistent ger tips om hur du kommer vidare





Labbar

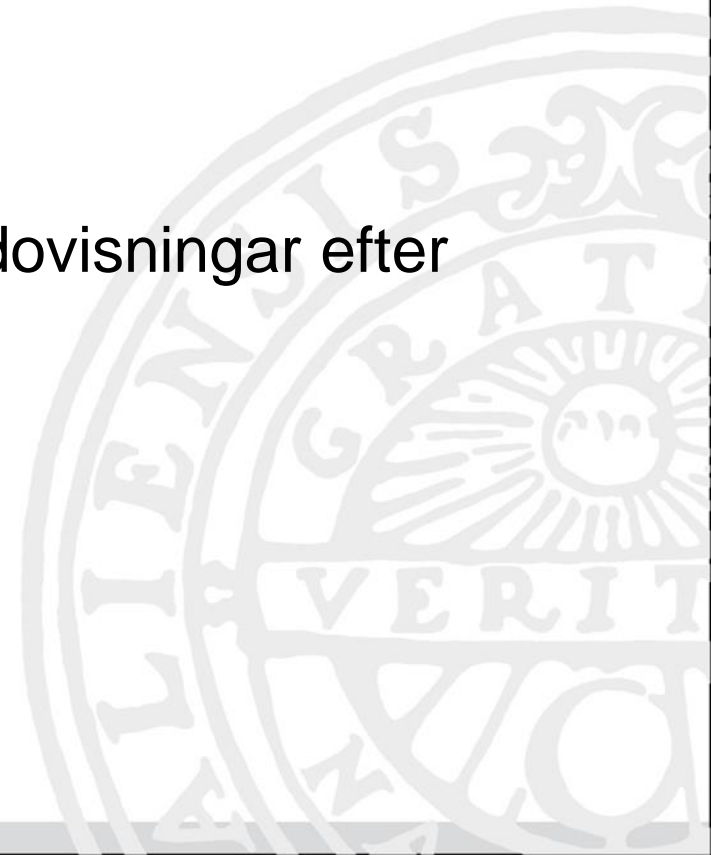
- Klar att redovisa?
 - Skriv upp ditt namn på lista "Redovisa"
 - Se till att ha leg, eftersom detta är examination
 - Du förklara din kod individuellt, vad som skulle hända om man gjorde ändringar, svara på frågor från uppgiftstexten
 - Koden måste inte bara fungera, den måste vara tydlig också





Deadlines

- Redovisning måste ske senast vid deadline
- Absolut omöjligt?
 - Be mig om dispens.
 - Assistenten kan *inte* ta emot redovisningar efter deadline utan dispens.





Konkret

- OU1 Objekt, parameteröverföring, returvärden
– 2020-01-31/2020-02-03
- OU2 Listor 2020-02-06/2020-02-07 (**TÄTT!**)
- OU3 Arbeta med text 2020-02-14/2020-02-17
- OU4 Studentdokumentation 2020-02-27/2020-02-28
(uppgiften kan ändras!)
- OU5 Trafiksimulering 2020-03-10/2020-03-11

Variabler

- I matematik uttrycker vi samband.
 - Vi säger hur världen "är"
- Om ett samband ($a = 3$, $a = b$, $b > c$) etablerats fortsätter det gälla
- Python är *imperativt*
 - Vi säger inte hur världen *är*, vi säger hur den ska *bli*
 - Kan testa hur den *är*



UPPSALA
UNIVERSITET

Tilldelning

```
a = 3
```

```
b = a
```

```
a = a + 1
```

```
c = 'Python är skoj'
```

```
b > a
```

```
c = 5
```

```
a > c
```

```
b = 6
```

```
b > a
```





Jämförelse

- Tilldelning `a = b`
 - "Gör så att variabeln `a` får det värde variabeln `b` har just nu"
- Jämförelse `a == b`
 - "Har `a` samma värde som `b`, i så fall sanningsvärdet `True`, annars sanningsvärdet `False`"



Programflöde

- Vi vill inte bara stapla kommandon på varandra.

```
if a > b:
```

```
    print('a är störst')
```

```
elif a == b:
```

```
    print('a lika med b')
```

```
else:
```

```
    print('b är störst (kanske?)')
```



Indentering

- Vanligt i många språk att man gör indrag (indenterar) för att visa vilka rader som hör ihop
- I Python är indenteringen *semantisk*
 - Indraget styr vad koden faktiskt betyder
 - Inget "end" eller liknande i slutet av ett if-block



Slingor

```
while a < b:  
    a = a + 1  
    print('Ökar a med 1')
```

- En `if`-sats kontrollerar villkoret en gång
 - Utför innehållet om det uppfylls
- En `while`-sats kontrollerar villkoret
 - Utför innehållet om det uppfylls
 - Och igen om det fortfarande uppfylls...
 - Och igen om det fortfarande uppfylls...
 - ...





UPPSALA
UNIVERSITET

Typer

$a = 3$

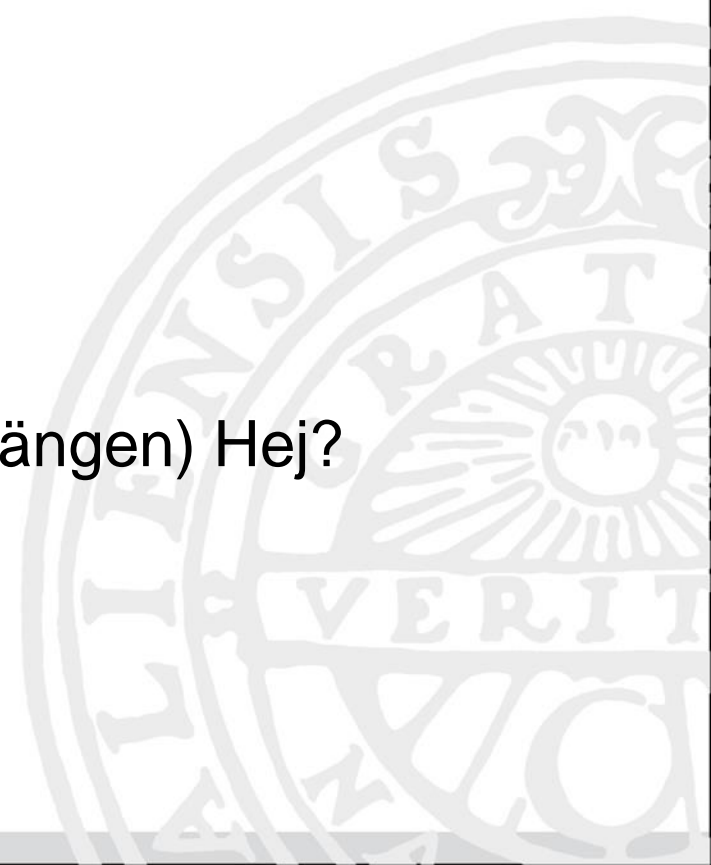
$b = \text{'Hej'}$

$a < b$

- Vad händer?

$a < b$ betyder alltså $3 < \text{'Hej'}$

Är heltalet 3 mindre än texten (strängen) Hej?





Typer

$a = '3'$

$b = 'Hej'$

$a < b$

- Vad händer?
 - $a < b$ betyder alltså $'3' < 'Hej'$
 - Är strängen 3 mindre än strängen Hej?
- Används för att sortera





Uttrycks typ

- Varje *uttryck* i Python har alltså en typ
 - Beror på typerna på ingående uttryck
 - En variabels typ beror på variabelns värde
 - En variabel kan byta typ vid tilldelning
 - `a = 3`
 - `a = 'Hej'`
 - Python är dynamiskt typat (en variabels typ kan ändras under körning, alltså dynamiskt)



Funktioner

- Vi har redan sett funktionen `print` i Python
- Vi använder funktioner för att organisera vår kod
- En funktion kan returnera värden, eller låta bli...
- Några speciella funktioner i Python används för typomvandling, exempel:
 - `str` för att skapa en sträng
 - `int` för att skapa ett heltal (tar heltalsdelen om invärdet är ett decimaltal)
 - `float` för att skapa ett decimaltal (mer exakt flyttal)



Exempel

`4 < 39`

`str(4) < 39`

`str(4) < str(39)`

`str(4 < 39)`

- Vad är typerna för de olika (del)uttrycken?
- Vilka jämförelser kan utföras? Vad blir resultaten?



UPPSALA
UNIVERSITET

Läsa in från användaren

```
a = input('Skriv in ett heltal!')
```



Dags för labb

- Vi är schemalagda i flera salar
 - När någon av 1515 och 2510 är schemalagda finns det ganska mycket plats där
 - Skriv upp er på lista för Hjälp om ingen assistent är omedelbart tillgänglig
- Försök att få igång den rena Pythontolken, jämför med ipython och Jupyter
 - Instruktioner i nätlektion 1
 - Målet i dag: gör klart nätlektion 1