TDP002 – Imperativ programmering

Human Centered Systems
Inst. för datavetenskap
Linköpings universitet

Översikt

- Välkommen till IP
- Om kursen
- Datorns grundbegrepp
- Historien om Python
- Hej värld i IDLE
- Principer vs Python
- Hitta rätt attityd
- Labb-gruppindelning
- Schema

"En lång vandring börjar
med att man tar på sig skorna"

Innovativ programmering

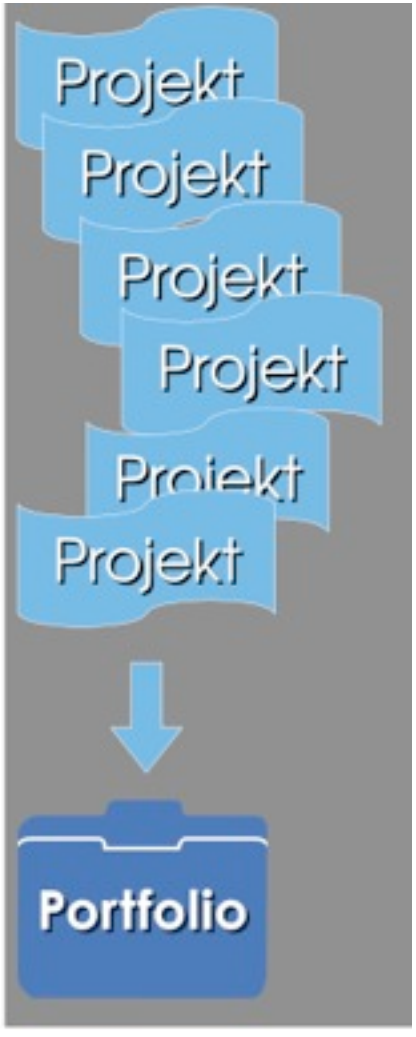
- Akademisk utbildning **och** hantverkskunnande
- Yrkesutbildning med rum för kreativitet och egna intressen
- Tillämpningsorienterad – lyhördhet mot teknikfront och industri
- Utbildning av programutvecklarproffs
  - Som kan hantverket
  - Fungera i grupp och kommunicera
  - Yrkesmässig attityd

Programplan för IP – översikt

ÅRSKURS 1
Programspråk och plattformar

ÅRSKURS 2
Tillämpad programmering

ÅRSKURS 3
Industrinära fördjupning



IP Årskurs 1

HT	VT
Handhavande av datormiljö, 4hp Imperativ programmering, 6hp Projekt: Egna datormiljön, 6hp	Innovativ prog.varudesign, 6hp Konstruktion av datorspråk, 6hp Projekt: Datorspråk, 6hp
Tentaperiod	Tentaperiod
Handh. av datormiljö (forts) OO programmering, 8hp Projekt: OO system, 6hp	Innovativ prog.varudesign (forts) Algoritm och systemkonstr, 6 hp Projekt: interaktivt system, 6hp
Tentaperiod	Tentaperiod

30 + 30 högskolepoäng (hp) per läsår
ca 1,5 hp per arbetsvecka

Du förväntas...

- ...delta i schemalagd undervisning
- ...läsa och arbeta flitigt på egen hand **varje dag**
  - Gör många egna övningar
  - Läs och experimentera på egen hand
  - Jobba 8-10 timmar
- ...fullfölja kurser på bra sätt
  - PP-tips 35: Finish What You Start
  - PP-tips 69: Gently Exceed Your User's Expectations
- ...ha en bra attityd till intensivt arbete och andra människor

Kursens syfte (Mål 1) – Teori

Att Du lär Dig:

- Programmera i ett imperativt språk
- De grundläggande principerna för imperativa språk
- Att kommunicera om program och programmering
- Skilja på generella principer och program- och språkspecifika
- Imperativ programmering så att du enkelt kan byta språk, t ex från Python till C++

Kursens syfte (Mål 2): Hantverk

Att Du lär Dig:

- Hantverket att programmera imperativt
- Använda programmeringspråket Python (för mindre program)
- Använda programmeringens grundelement bra

- Gå från givet problem till en egen lösning (som även andra uppskattar)
- Genom egen erfarenhet

Kursens syfte (Mål 3): Verktyg

Att Du lär Dig:

- Att utnyttja en generell utvecklingseditor
- Att du lär dig arbeta med en versionshanterare
- Genom egen erfarenhet
  - Hur **Emacs** kan användas på ett avancerat sätt
  - Hur **Subversions eller liknande** grundläggande funktioner fungerar
  - Arbeta i (**Ubuntu**) Linuxmiljön under utveckling

Examinationsmoment

- Datortenta, 3hp
  - teorifrågor och individuell problemlösning (+ verktyg)
  - Halvtids-dugga , samla poäng till sluttentan
- Laborationer, 3hp
  - (muntlig och) skriftlig redovisning
  - 3 obligatoriska seminarier
  - 3 obligatoriska dojos/programmeringsstugor
- Egna övningar
  - Ett “måste” för att klara datortentan

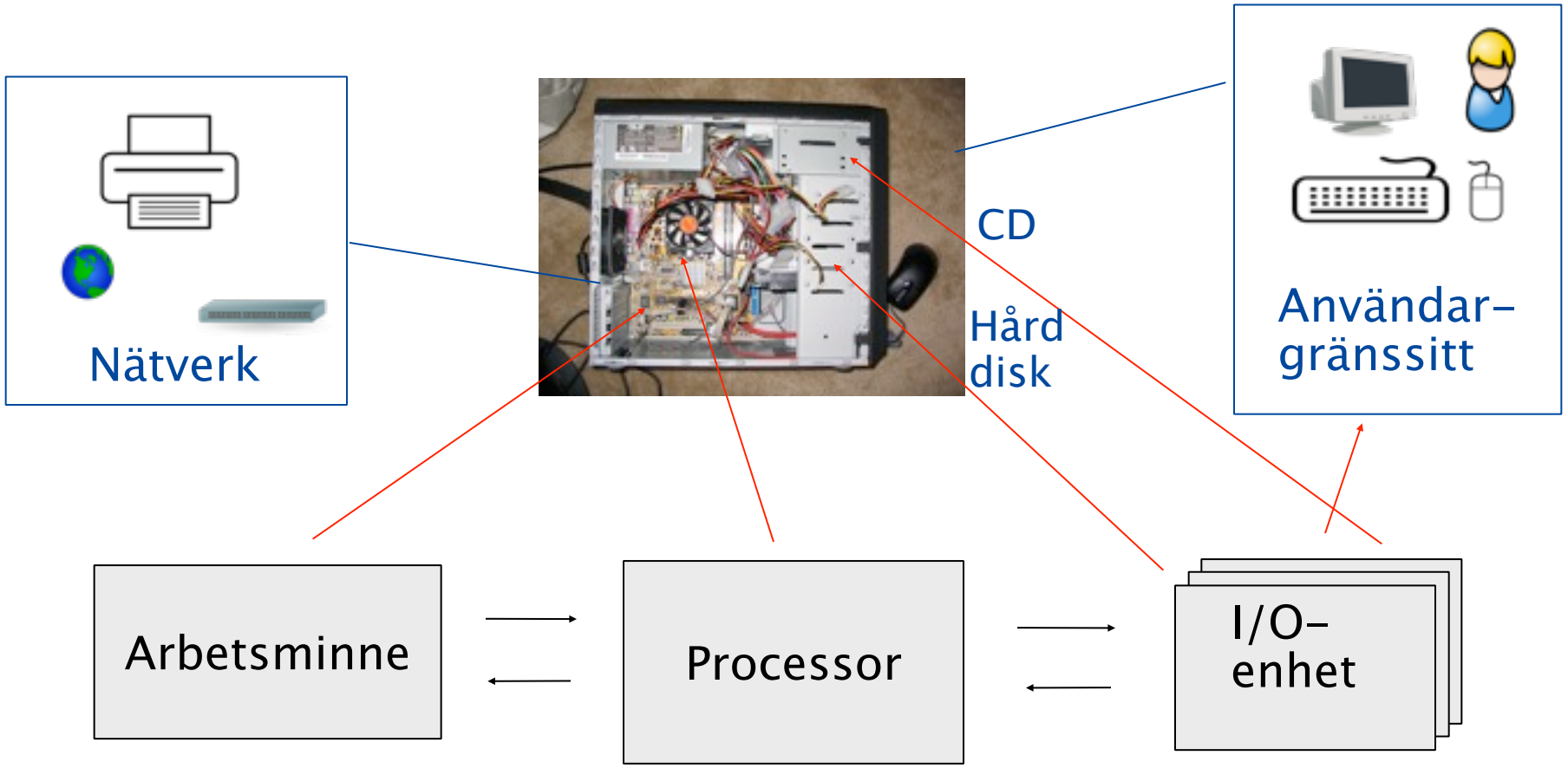
Tillgodoräknanden-Dugga

- Ger **3 poäng** på tentamen (för betyg 4/5).

Kurslitteratur: LP + PL

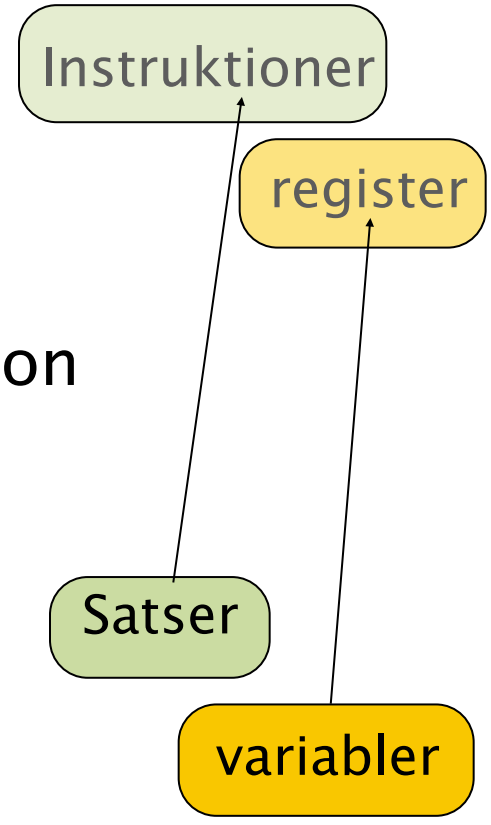
- Learning Python, 4th Edition. Mark Lutz O'reilly 2009
- Concepts of Programming Languages, 9th Edition. Robert W Sebesta, Addison Wesley 2009

Vad är en dator? Exempel: PC



Vad är ett program?

- processor med von-Neumann-arkitektur
  - samma **arbetsminne** för data och program
  - program är en sekvens av **instruktioner**
  - **register** för aktuell instruktion och data
- imperativa högnivåspråk, t ex C, Java, Python
  - programmet består av **satser, block** och **deklarationer**
  - exekvering av satser sker sekvensiellt
  - Data hålls med **tillståndsvariabler**

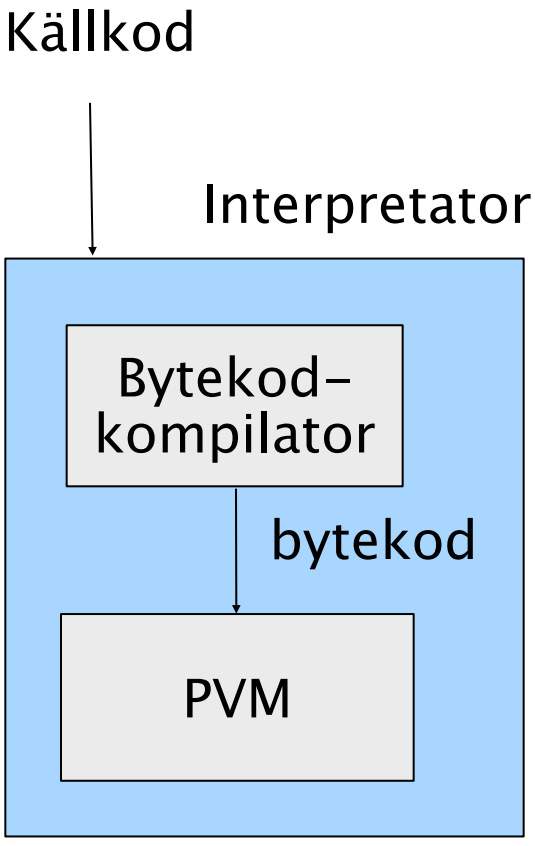


Tre grundbegrepp (Källa: wikipedia)

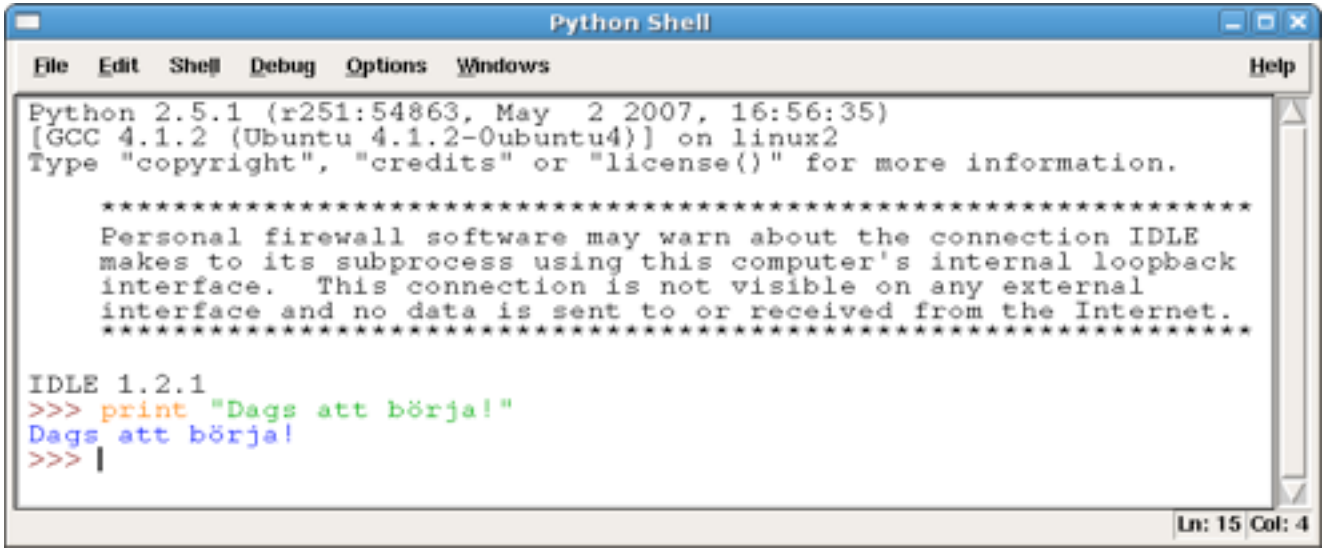
- Källkod (eng source code):
  - "is any sequence of statements and/or declarations written in some **human-readable** computer programming language."
  - "The source code may be converted into an executable file by a **compiler**, or executed on the fly from the human readable form with the aid of an **interpreter**."
- Interpretator: ett program som exekverar direkt från källkod
- Kompilator: ett program som översätter från ett format till ett annat, t ex från källkod till exekverbar kod

Hur kör man Pythonprogram (Kap 3)

- Python körs (normalt) via en Python-interpretator
- Python-interpretatorn har en inbyggd bytekod-kompilator
  - byte-kod är s k intermediär kod
  - Även s k objekt-kod stöds
- Python-byte-kod exekveras med python virtual machine (PVM)
- Moderna språk ofta en mix av interpretering/kompilering



IDLE: standard-IDE för Python



- IDE: Integrated Development Environment
- interaktiv Python via IDLE-skalet

- Även en editor för programfiler
- Inbyggd debugger

Olika sätt att exekvera Pythonkod

- Interaktivt via kommandorad: kommandoskal
  - DOS, IDLE-skal, Linux-skal
  - print-kommandot onödigt; ladda fil möjligt; inga blanka rader
  - Testa kod interaktivt...användbart vid experiment
- Anropa ett fil myprogram.py från kommandorad
  - % python myprogram.py
  - Linux: #!/usr/bin/python el. bättre #! /usr/bin/env python (utläses hash bang eller shebang-rad på unix-språk)
  - % chmod a+x myprogram.py
- Generell editor/IDE: t ex Emacs och C-c C-c

Så hur ser du Pythonkod ut?

Med tal:

```
>>> 4 + 3
7
>>> (42 - 33) / 4
2
>>> (42 - 33) / 4.5
2.0
>>> (42 - 33) / 5.3
1.698113207547169
>>> (42 - 33) / 5
1
>>> length = 3.6
>>> depth = 4
>>> length * depth
14.4
```

Med strängar:

```
>>> "What's this?"
"What's this?"
>>> "What's this?" + ' Not "that"'
'What\'s this? Not"that"'
>>> message = "Let the Game Begin"
>>> message[8:12]
'Game'
>>>"Error %(code)i \ ... %(msg)s" % \
... {'code': 334, 'msg': 'Couldn\'t parse'}
"Error 334 Couldn't parse"
>>> "test".islower()
True
>>> "Test".islower()
False
```

Vårt första Pythonprogram

Filen helloworld.py:

```
#!/usr/bin/env python
# -*- coding: utf-8

print "Hej Värld!"
raw_input("\nTryck valfri tangent för att avsluta")
```

```
larde@~:./helloworld.py
Hej Värld!

Tryck valfri tangent för att avsluta
larde@~:
```

Enkla Python-program är som **skript** dvs kommandon på fil

Program = fil = modul

Kommando = sats

Programmeringsspråk

- **Maskinspråk** är datorns språk. Det är binärkod (1 och 0)
- **Assemblerspråk** är mnemoniskt maskinspråk
  - Ex: “MOV A,#0FFH”
- **Högnivåspråk** är utvecklade för att passa programmeraren

1943 Eniac Code
1950 Short Code
1954 FORTRAN
1958 Algol 58,Lisp
1959 COBOL
1963 Algol 68
1964 Basic
1967 Simula
1971 Pascal
1972 C, Prolog,ML
1976 Smalltalk
1977 Bourne Shell
1983 C++, Ad
1987 Perl
1991 Python, VB
1993 Ruby
1994 PHP
1995 Java
2000 C#

Principer: syntax och semantik

- **Syntax**: formatet på språkets konstruktioner
- **Semantik**: betydelsen hos språkets konstruktioner
- Varje programspråk har sin egen **unika syntax**
- **Paradigm**: språken har en gemensam semantisk kärna men kan variera i olika detaljer
- I våra kurser: Vi lär oss principerna – så kan vi paradigmet
  - Språket som vi väljer är ett exempel...om än valt med omsorg
  - Men då måste vi lära oss se **principerna i exemplet**

Python vs Java – en skillnad i syntax...

Python:

```
If x == 2 and y == 3
    print “hej”
```

Java:

```
If (x == 2 && y == 3) {
    System.out.println(“hej”);
}
```

“villkor”

```
for x in range(7):
    print x
```

```
for (int x=0; x < 7; x++) {
    System.out.println(x);
}
```

“iteration”

...men i många avseenden **samma semantik**

Program **in** or **into a Language**

> “Programmers who program "in" a language limit their thoughts to constructs that the language directly supports. If the language tools are primitive, the programmer's thoughts will also be primitive.”

“Programmers who program **into a language** first decide what thoughts they want to express, and then they determine how to express those thoughts using the tools provided by their specific language.”

-- Steve McConnell, paraphrasing David Gries, Code Complete

Historien om Python

- Version 0.9 lanserades 1991 av Guido van Rossum
  - Evolverat ur ABC (jmf Basic) och Modula-3 (jmf Pascal)
  - Python 2.6.2 nuvarande stabila version
  - Öppen källkod (GPL-kompatibel licens)
- Programming for Everybody
  - Multiparadigm: Imperativt, funktionellt, objekt-orienterat
  - Skriptspråksfamiljen: Perl, Python, Tcl, bash, etc
  - rapid development: bra för nybörjare och proffs
- Python 3.1.1 – "nästa" utgåva

Om språket Python

- Kraftfullt skriptspråk
  - eng. script = manus, manuskript
  - kommandospråk, dvs används även interaktivt
  - Unix-begrepp: skalfönstrets språk, skalskript
  - Bra för “enradare” men även större program
  - Klister med bra stöd för filhantering, strängar, operativsystemsanrop
  - Inbäddat i andra språk t ex C och Java
- Generellt lättanvänt imperativt, objekt-orienterat språk
  - Enkla konstruktioner: typsystem, modulhantering, exkvering
  - Korta program; liten mängd standardbibliotek

Hitta rätt attityd

Aah! Äntligen semester!!

Regel nummer 1:
Attityd är allt

Programmering – “Likt Sherlock Holmes”

- Mord/problem är intressant
- Sökandet är jobbet
- Ju svårare desto bättre
- Ett fall i taget: fokus
- Välj dina fall med omsorg
- Klienterna kommer till dig för att det är svårt
- Varje detalj är viktig
- Arbeta systematiskt och outtröttligt

Till sist några allmänna råd...

Slösa inte med din lånade tid här

- Sätt upp ditt mål och följ det
- Ta ansvar för din egen utbildning
- Var flitig och uthållig
- Fullfölj – en sak (kurs) i taget

- Gå din egen väg – låt dig inte distraheras

Summering

- Learning Python, Kap 2-3
- Programming Languages, Kap 1, (frivilligt 2), 3.1
- Wikipedia:
  - http://en.wikipedia.org/wiki/Source_code
  - http://en.wikipedia.org/wiki/Compiler
  - http://en.wikipedia.org/wiki/Interpreter_%28computer_software%29
- Rekommenderade övningar
  - Se övningsmaterialet, föreläsning för föreläsning
  - Även gamla tentauppgiftproblem

Labbgrupper

- Jobba två och två
- Hitta någon på samma nivå
  - Nybörjare nästan ingen eller ingen erfarenhet av programmering
  - Gjort lite på gymnasiet eller som hobby
  - Tagit någon/några kurs på Högskola/Universitet
- Skriv upp er i webreg (3 grupper gr1, gr2, gr3)

Schema

- Er webreg grupp är er grupp i schemat
- Vid schemalagda tillfällen enbart de grupper i schemat.
- “Handledd laboration” i schemat innebär assistent på plats
- Schemat kommer att förändras
- Imorgon 13-15 utgår
- Imorogn 15-17 gr3 och gr1
- Första veckan
  - Komma igång
  - Registrera i webreg
  - Installera det som behövs
  - Prova emacs/svn
  - Prova på lite python