

DD1311 Programmeringsteknik med PBL

Föreläsning 1

Lärandemål

Efter godkänd kurs ska du kunna

- följa reglerna i programspråkets syntax,
- tillämpa och redogöra för regler för god programmeringsstil,
- hitta och rätta fel i program,
- ändra i färdiga program,
- hämta data från fil och lagra på fil,

Fler lärandemål

- använda styrstrukturer (villkorssatser och slingor) och veta när dom behövs,
- dela upp ett större problem och konstruera funktioner för delarna,
- använda inbyggda datastrukturer,
- utveckla enkla grafiska gränssnitt,
- granska andras program.

Varför?

för att du ska kunna:

- använda programmering för att lösa problem,
- tillämpa problemlösningsmetodik även inom andra områden än programmering,
- diskutera programutveckling med experter,
- bedöma kommersiella program.

Kursens pedagogik

- PBL: Problembaserat lärande
- Labbtimmarna och grupptimmarna bör du alltid gå på.
- Koncentrerade entimmesföreläsningar med läsanvisningar. Kom förberedd för bästa resultat!
- Målrelaterade betygskriterier; välj själv betyg.

Vad ska du göra?

- Laborationer (LAB1; 1,5hp)
 - Fem labbar: program & instuderingsuppgifter
 - Kan ge bonus till provet – redovisa i tid!
- Prov (LAB2; 1,5 hp)
 - Skriftligt prov på grunderna
 - Öva med diagnostiska proven
- P-uppgift (LAB3; 3p)
 - Personlig uppgift
 - Ger betyget i kursen

Schema

- Föreläsningar:
 - Två entimmesföreläsningar i veckan på tisdagar och torsdagar, oftast i E1
- Labbar:
 - Två timmar arbete i datorsal
 - Följs av en timme egen tid för instuderingsfrågor
 - Och en frivillig frågestund efter den timmen
- Grupptimme
 - Gruppen träffas för redovisning av labb & instuderingsfrågor

Reklam för Python

- Lätt att lära sig
- Kraftfullt
- Objektorienterat
- Kan kombineras med andra applikationer
- Flyttbart
- Används av många
- Gratis

Python IDLE

- Här skriver och kör man programmen
- Fönstret *Python Shell*
 - Testa satser
 - Se programkörningen
- Redigeringsfönstret (Untitled)
 - Skriva, redigera, spara egna program
 - Köra programmet

Enklaste programmet

```
print "Hej"
```

- En *sats* som skriver ut Hej på skärmen.
- *Kommandot* heter `print` (PRINT eller Print fungerar inte)
- Texten "Hej" kallas för ett *uttryck*
sats=statement
kommando=command
uttryck=expression

Uppgift: Ge exempel på tre värden av olika typ!

Heltalsberäkningar

Operator:	Beskrivning:	Exempel:	Resultat:
*	multiplikation	3*4	12
/	division	53/10	5
%	modulo	53%10	3
+	addition	10+12	22
-	subtraktion	5-8	-3

heltal=integer

Flyttalsberäkningar

Operator:	Beskrivning	Exempel:	Resultat:
*	multiplikation	2.0*1.5	3.0
/	division	10.0/8.0	1.25
%	modulo	4.25%4.0	0.25
+	addition	0.3+0.4	0.69999999999999996
-	subtraktion	1.0-0.1	0.90000000000000002

flyttal=floating-point number

Uppgift: Hur kan man använda % för att ta reda på om ett tal är jämnt eller udda?

Strängar

- En *sträng* är en följd av tecken.
- Strängar *konkateneras* med +
 - tex blir "kus" + "lig" strängen "kuslig"
- Strängar upprepas med *
 - tex blir "nä"*3 strängen "nänänä"
- Sträng är en *datatyp*. Andra datatyper är *heltal* och *flyttal*.

sträng=string

Konkatenera (slå ihop)=concatenate

Typkonvertering

Funktion:	Beskrivning:	Exempel:	Blir:
float(x)	Konverterar till flyttal	float("3.14")	3.14
int(x)	Konverterar till heltal	int("17")	17
str(x)	Konverterar till en sträng	str(39)	"39"

parameter=argument

Variabler

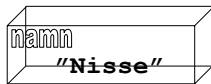
- Variabler används för att lagra data i programmet.

- En variabel skapas i en tilldelning.

```
# Variabeldemonstration  
namn = "Nisse"  
print "Grattis" + namn
```

- Variabelnamn får bara innehålla bokstäver, siffror och understreck, men får inte börja med en siffra.

tilldelning=assignment



Reserverade ord

Följande ord är reserverade i Python:

```
and      del      for      is      raise  
assert  elif    from    lambda  try  
break   else    global  not     while  
class   except  if      or      return  
continue exec  import  pass  
def     finally in      print
```

Reserverade ord har betydelse i språket och får inte användas som variabelnamn.

Uppgift: Ge förslag på tre olika variabelnamn.

Inläsning

- Funktionen `raw_input()` används vid inläsning av strängar:

```
namn=raw_input("Vad heter du?")  
print "Nämen " + namn + " då!"
```

- Vid inläsning av tal används `input()`

```
storlek=input("Ge skostorlek: ")  
print "Ta "+str(storlek+1)+" i skridskor",  
print "så får du plats med sockor också!"
```

Inläsning=user input

Strängmetoder

Metod	Exempel
<code>upper()</code>	"kanin" -> "KANIN"
<code>lower()</code>	"KANIN" -> "kanin"
<code>swapcase()</code>	"KaniN" -> "kANIn"
<code>capitalize()</code>	"kanin" -> "Kanin"
<code>title()</code>	"liten blå kanin" -> "Liten Blå Kanin"
<code>strip()</code>	" kanin " -> "kanin"
<code>replace(x, y)</code>	"kanin" -> "kinin" (om x="a",y="i")

Metodanrop

- Så här anropas en metod:

```
mening=raw_input("Skriv en förolämpning: ")
print "Så här ser den ut med versaler:"
print mening.upper()
```

metod=method
anropa=invoke

Uppgift: Spelar det någon roll i vilken ordning satserna står?

Kommentarer

Alla rader som börjar med # blir kommentarer, som datorn inte bryr sig om.

```
# Programmet som ger komplimanger
# Skrivet av Linda Kann 100119
print "Hej Linda, "
print "Vilken fin klänning",
print "och vad gott du luktar!"
```

BMI-program

```
# Läser in längd och vikt,  
# skriver ut BMI (Body Mass Index)  
print "Välkommen till BMI-beräknaren!"  
langd=input("Hur lång är du (i meter)? ")  
vikt=input("Hur mycket väger du (i kg)? ")  
bmi=vikt/(langd*langd)  
print "Din bmi är: ", bmi
```

Algoritm

1. Läser in indata
2. Gör beräkningar
3. Skriver ut resultatet

