*Skolan för Datavetenskap och kommunikation*

# PROGRAMMERINGSTEKNIK

FÖRELÄSNING 18

# Dagens föreläsning

- Betygskriterier
- P-redovisning
- Komplettering
- Vad händer om man inte hinner klart?
- Plussa

**för betyg E ska du visa att du kan:**

- följa reglerna i programspråkets syntax,
- tillämpa och redogöra för regler för god programmeringsstil (såsom användarvänlighet, kommentarer, felhantering, strukturering, flexibilitet),
- upptäcka och korrigera programmeringsfel,
- modifiera givna program,
- överföra data mellan fil och program,
- identifiera behovet av och använda styrstrukturer (villkorssatser och slingor),
- dela upp ett större problem i hanterliga delar och konstruera funktioner för dessa,
- använda de datastrukturer som finns inbyggda i programspråket, samt välja datastrukturer som passar för det aktuella problemet,
- använda och konstruera egna klasser,
- granska andras program

**för betyg D ska du dessutom visa att du kan:**

- följa en given tidsplan för arbetet,
- skriva ett perfekt program som i alla avseenden uppfyller givna krav på användarvänlighet, begriplighet och strukturering

**för betyg C ska du dessutom visa att du kan:**

- infoga felhantering för att få ett program som inte kraschar för felaktiga indata

**för betyg B ska du dessutom visa att du kan:**

- konstruera och implementera en svårare algoritm

**för betyg A ska du dessutom visa att du kan:**

- sätta dig in i och använda en större modul (t ex tkinter, pygame, urllib) för att utveckla ditt program

# GRANSKNING

- ✓ Ditt program måste granskas.
- ✓ Du måste granska ett program.

Den som granskar ska fylla i granskningsprotokollet, se länk under ”P-uppgiften”

Granskaren ska vara med vid redovisningen.

Alla ska granska *exakt ett* program (inte fler)

# BOKA TID

- Du måste boka en redovisningstid, se länk under ”Bokning”.
- Skriv också upp vem som granskar ditt program (men hen måste också boka en egen redovisningstid).

# REDOVISNING

- Assen kollar leg
- Granskaren får berätta
- Assen provkör
- Assen tittar på koden
- Assen frågar
- Assen fyller i protokollet
- Assen kollar att programmet inte är kopierat
- Du får betyg (E,D,C,B,A eller komplettering)

# BONUS

- Du samlar ihop bonuspoäng från
  - Labbarna 1-6 (max 24 bp)
  - Provet (max 4 bp)
  - Specen (max 4 bp)
  - Prototypen (max 4 bp)
  - *Webbdelen (max 4+4+4+2 bp)*
- Om du får 35 bp eller mer höjs ditt slutbetyg ett steg, från E->D osv
- ...men om du får F höjs inte betyget till E

# KOMPLETTERING

- Om ditt program inte uppfyller kraven kan du få komplettera:
  - ✓Antingen senare samma dag
  - ✓Eller boka ny tid
- Granskaren behöver inte vara med vid kompletteringen (om inte granskaren också ska komplettera sin granskning).

# INTE KLAR I TID?

- Efter kursens slut anordnas uppsamlingstillfällen (som omtentor)
- 2-9 juni labbvecka
- Länkar/mer info kommer på KTH Social
- Om du inte blir godkänd på P-uppgiften i tid (med minst E) missar du chansen att plussa.

# PLUSSA

- Om du blir godkänd i tid med minst E ...
- ... så har du möjlighet att *plussa* för högre betyg på uppsamlingstillfällen fram t o m **30 januari 2017**
- Du behöver inte göra ny spec/prototyp/granskning när du plussar.

# OBS!

- Det finns ett moment kvar efter P-uppgiften:
- LAB4: Webbdelen (html/css/javascript/php)

# FELHANTERING, T EX

- Felaktig inmatning:
  - Tecken istället för tal
  - För stort/för litet tal
- Filer:
  - Infil saknas
  - Felaktiga data i filen
- Lista/dictionary:
  - Index saknas
  - Nyckel saknas

# EXCEPTION - REPETITION

- När något blir fel i ett Python-program uppstår ett särfall, t ex NameError:

```
>>> print sko


Traceback (most recent call last):
  File "<pyshell#17>", line 1, in -toplevel-
    print sko
NameError: name 'sko' is not defined
```

- Man kan ta hand om särfall genom att införa `try-except-else`-satser för de delar i programmet som kan krascha.

# SÄRFALL - EXEMPEL

```
try:
    tal = input("Vad ska inverteras? ")
    invers = 1.0/tal
except (ZeroDivisionError):
    print "Noll kan inte inverteras"
except (NameError):
    print "Du borde ha skrivit ett tal!"
else:
    print "Inversen blev", invers
```

# EXEMPEL I SLINGA

```
def lasPengar():
    """ Läser in tills man ger ett heltal"""
    pengar = None
    while not pengar:
        try:
            svar = raw_input(”Ange belopp: ")
            pengar = int(svar)
        except(ValueError), e:
            print "Felaktigt belopp, försök igen"
    return pengar
```

# FELHANTERING I TKINTER

- I messagebox finns "popupfönster" som lämpar sig för felhantering:
  - showinfo
  - showwarning
  - showerror
  - askquestion
  - askyesnocancel
- Alla tar två parametrar: title och message
- Vissa har returvärde (askyesnocancel)

# EXEMPEL: SHOWERROR

```
from tkinter import *

rot = Tk()

messagebox.showerror(title="Fel",    \
        message="Du har just gjort fel.")


rot.mainloop()
```

# GETATTR

- För att hämta värdet på ett namngivet atribut:
- värde = getattr(objekt, attributnamn)
- Exempel:

```
djur = Husdjur()
w = getattr(djur, "namn")
```

# SETATTR

- För att ändra värdet på ett namngivet atribut:
- setattr(objekt, attributnamn, värde)
- Exempel:

```
djur = Husdjur()
setattr(djur, "namn", "Leo")
```