Skolan för Datavetenskap och kommunikation

PROGRAMMERINGSTEKNIK

FÖRELÄSNING 9

ARV

- Kap 11 i boken
- moduler
- arv
- klassen object
- superklass
- subklass

arv = inheritance

MODULER

- En modul kan man hämta till programmet med en import-sats

```
import random
from random import *
```

- En egen fil med klasser och funktioner kan importeras på samma sätt

```
from geometri import *
```

KODÅTERVINNING

- Samma satser om igen?
  - *Skriv en slinga eller en funktion!*
- Samma funktioner i nytt program?
  - *Skriv en modul och importera den!*
- Samma data som skickas in i alla funktionerna?
  - *Skriv en klass!*
- Flera klasser med liknande attribut och metoder?
  - *Skriv en superklass och låt klasserna ärva från den!*

ARV

```
class Subklass(Superklass):
```

Subklassen ärver alla

attribut och metoder

från klassen Superklass.

OBJECT

När du skriver

```
class MinKlass(object):
```

betyder det att MinKlass ska ärva från *object*

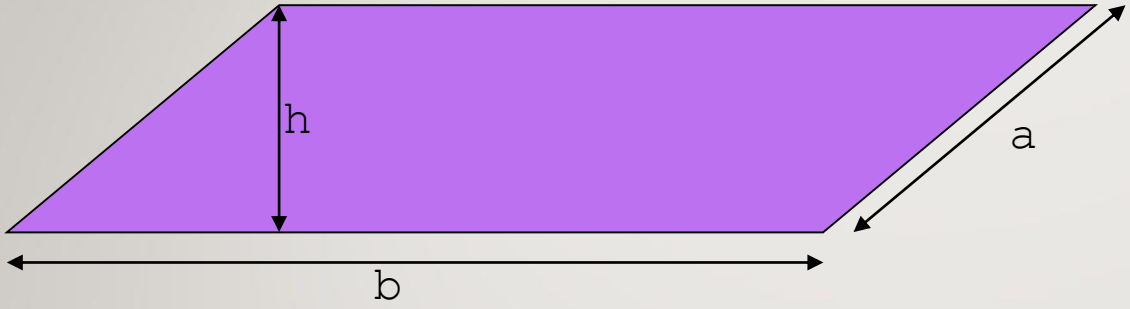
object är superklass till alla klasser i Python.

TKINTER - EXEMPEL

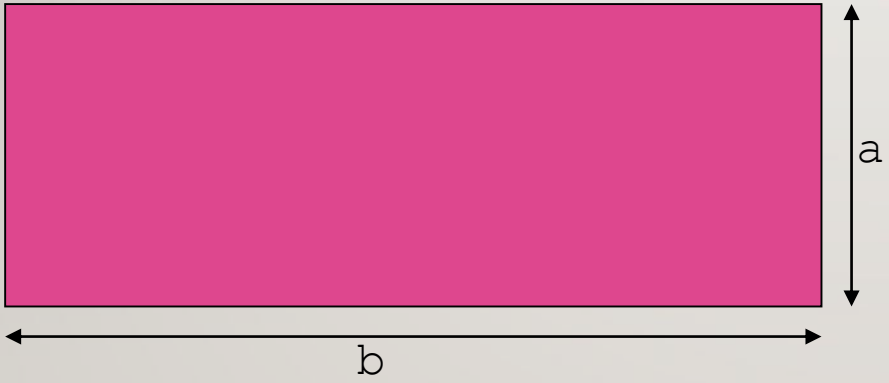
```
class Fetknapp(Button):

    def fetstil(self):
        self.config(font = ('times', 32, 'bold'))

    def kursiv(self):
        self.config(font = ('times', 32, 'italic'))
```

EXEMPEL: GEOMETRI

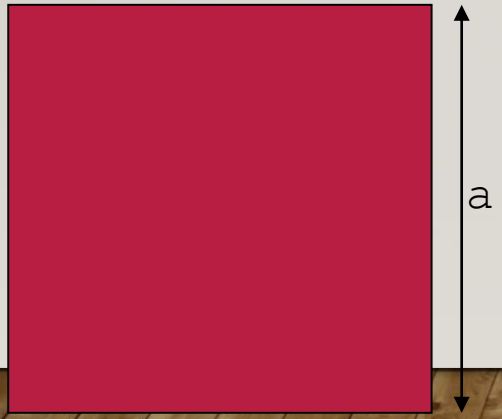
- *Parallellogram* är den mest generella figuren - den får bli superklass
- *Rektangel* är en sorts *parallellogram* med räta vinklar - vi låter den vara subklass till Parallellogram
- *Kvadrat* är en sorts *rektangel* med lika sidor - den får vara subklass till Rektangel



Parallellogram:
area = h*b
omkrets = 2*(a+b)



Rektangel:
area = a*b
omkrets = 2*(a+b)



Kvadrat:
area = a*a
omkrets = 4*a

```
# Modul med tre klasser
#***********************************************************
#************** superklassen Parallellogram **************
#***********************************************************

class Parallellogram(object):

    def __init__(self,a,b,h):
        self.kant1 = a
        self.basKant = b
        self.hojd = h

    def area(self):
        return self.hojd*self.basKant

    def omkrets(self):
        return 2*(self.kant1+self.basKant)
```

```
#*************************************************
#******  Rektangel är subklass till Parallellogram ****
#*************************************************
class Rektangel(Parallellogram):

    def __init__(self,a,b):
        self.kant1  =  a
        self.basKant = b

    def area(self):
        return self.kant1*self.basKant
```

```
#*************************************************************
#**********  Kvadrat är subklass till Rektangel ************
#*************************************************************
class Kvadrat(Rektangel):

    def __init__(self,a):
        self.kant1 = a
        self.basKant = a
```

KVADRAT-OBJEKT

- Vi skapar en instans av Kvadrat…

  … och anropar area() och omkrets().

- Vilka metoder är det som används?

```
from geometri import *

def main():
    sida = input("Hur stor är sidan på din kvadrat? ")
    kvadrat = Kvadrat(sida)
    print("Din kvadrat har arean", kvadrat.area())
    print("och omkretsen", kvadrat.omkrets())

main()
```

ÄRVA ALLT

- Det går bra att skapa en egen klass som ärver allt från superklassen.

```
class Sjutton(Exception):
    pass
```

- pass betyder "tom sats" - inget händer här.

```
class Sjutton(Exception):
    pass

def skriv_tal(antal):
    for i in range(antal):
        if i == 17:
            raise Sjutton
        else:
            print(i)

def main():
    try:
        skriv_tal(100)
    except Sjutton:
        print("Det var som sjutton!")

main()
```

SUPER

- Med hjälp av super() går det att komma åt superklassens metoder.

```
class Lila:

    def __init__(self):
        self.färg = "lila"


class Brun(Lila):

    def __init__(self):
        self.färg = "brun"
        super().__init__()
```