

## PROGRAMMERINGSTEKNIK

FÖRELÄSNING 10

## ARV

- Läs kap 9 i boken!

## KODÅTERVINNING

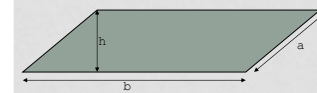
- Samma satser om igen?  
*Skriv en funktion!*
- Samma data som skickas in i alla funktionerna?  
*Skriv en klass!*
- Flera klasser med liknande attribut och metoder?  
*Skriv en superklass och låt klasserna ära från den!*

## ARV

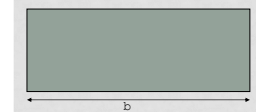
- Arv låter oss återanvända klasser.
- Exempel: `class StartKnapp(Button)` :
- Klassen `StartKnapp` ärver alla
  - attribut
  - metoderfrån klassen `Button`.
- `StartKnapp` är *subklass* till `Button`
- `Button` är *superklass* till `StartKnapp`

## EXEMPEL: GEOMETRI

- Ett enkelt exempel som visar hur arv fungerar:
  - Parallelogram är den mest generella figuren - den får bli superklass
  - Rektangel är en sorts parallelogram med rätta vinklar - vi låter den vara subclass till Parallelogram
  - Kvadrat är en sorts rektangel med lika sidor - den får vara subclass till Rektangel



Parallelogram:  
area =  $h \cdot b$   
omkrets =  $2 \cdot (a+b)$



Rektangel:  
area =  $a \cdot b$   
omkrets =  $2 \cdot (a+b)$



Kvadrat:  
area =  $a \cdot a$   
omkrets =  $4 \cdot a$

```
# Modul med geometri-klasser
#*****
#***** superklassen Parallelogram *****
#*****
class Parallelogram(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 subclass till Parallelogram *****
#*****
class Rektangel(Parallelogram):

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

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

```
#*****
#***** Kvadrat är subclass 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()
```

## MODULER

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

```
import random
from random import *
```
- En egen fil med klasser och funktioner kan importeras på samma sätt
 

```
from geometri import *
```

## SPELKORT

- Klassen Kort representerar ett spelkort



- Klassens attribut är `valor` (2-10, knekt, dam, kung, ess), `farg` (Klöver, Ruter, Hjärter och Spader) och `framsidanUpp` (True om kortet är uppvänt)
- Metoderna är `__init__` (konstruktorn), `__str__` (kortet som sträng), `vand` (vänder)

```
class Kort(object):
    """ Ett spelkort. """
    VALORER = ["ess", "2", "3", "4", "5", "6", "7",
              "8", "9", "10", "knekt", "dam", "kung"]
    FARGER = ["Klöver", "Ruter", "Hjärter", "Spader"]

    def __init__(self, valor, farg, synligt = True):
        self.valor = valor
        self.farg = farg
        self.framsidanUpp = synligt

    def __str__(self):
        if self.framsidanUpp:
            rep = self.farg + "-" + self.valor
        else:
            rep = "XX"
        return rep

    def vand(self):
        self.framsidanUpp = not self.framsidanUpp
```

## HAND

- Klassen Hand representerar en korthand



- Klassens attribut är `korten` (en lista med kort).
- Metoderna är `__init__`, `__str__`, `bort` (tar bort korten), `stoppaIn` (lägger till ett kort), `ge` (ger ett kort till en annan korthand)

```
class Hand(object):
    def __init__(self):
        self.korten = []

    def __str__(self):
        if self.korten:
            rep = ""
            for card in self.korten:
                rep += str(card) + "\t"
        else:
            rep = "<tom>"
        return rep.ljust(14)

    def bort(self):
        self.korten = []

    def stoppaIn(self, card):
        self.korten.append(card)

    def ge(self, kort, annanHand):
        self.korten.remove(kort)
        annanHand.stoppaIn(kort)
```

## HAND-OBJEKT

- Ett Hand-objekt har bara ett attribut: `korten` som är en lista med Kort-objekt.
- Rita ett exempel:

korten

valor	"4"	valor	"5"	valor	"8"	valor	"9"
farg	Hjärter	farg	Spader	farg	Hjärter	farg	Spader
framsidan	True	framsidan	True	framsidan	True	framsidan	True
	0		1		2		3

## LEK

- Vi vill ha en klass `Lek` som representerar en hel kortlek.
- Vi skriver `class Lek (Hand)`
- Klassen `Lek` ärver då alla attribut och metoder från `Hand`.
- Vi definierar också tre nya metoder i klassen `Lek`: `fill`, `blanda` och `delatUt`.

```
class Lek(Hand):
    """ En kortlek. """
    import random

    def fyll(self):
        for farg in Kort.FARGER:
            for valor in Kort.VALORER:
                self.stoppaIn(Kort(valor, farg))

    def blanda(self):
        random.shuffle(self.korten)

    def delaUt(self, personer, perHand = 1):
        for runda in range(perHand):
            for person in personer:
                if self.korten:
                    oversta = self.korten[0]
                    self.ge(oversta, person)
                else:
                    print("Slut på kort!")
```