Skolan för Datavetenskap och kommunikation

DD1311 Programmeringsteknik med PBL

Föreläsning 9

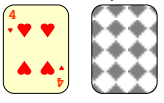
- Mer om objekt:
  - Lista av objekt
- Mer om funktioner/metoder:
  - Defaultparametrar

```
def mingel(self):
  """Låter husdjuren träffas och ev föröka sig"""
  print "Nu är det mingel i Buren!"
  n = len(self.lista)
  for i in range(n-1):
    jag = self.lista[i]
    for j in range(i+1,n):
      du = self.lista[j]
      if jag.kontakt(du):
        self.lista.append(Husdjur(jag.namn+du.namn))
```

djur 0 & djur 1	djur 0 & djur 2	djur 0 & djur 3
djur 1 & djur 2	djur 1 & djur 3	
djur 2 & djur 3		

Spelkort

- Klassen Kort representerar ett spelkort



- Klassens attribut är `valor` (2-10, knekt, dam, kung), `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
```

Defaultparametrar

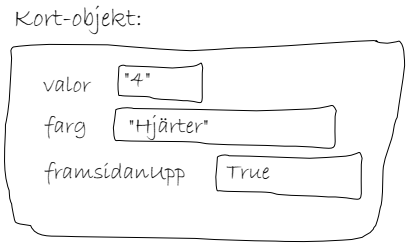
```
def __init__(self, valor, farg, synligt = True):
```

- När en funktion/metod definieras kan man ge parametrar *defaultvärden*.
- Om parametern utelämnas vid anropet är det defaultvärdet som används. Exempel:

```
hemligtKort = Kort("9","Spader",False)
synligtKort = Kort("4","Hjärter")
```

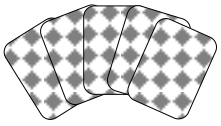
- Parametrar utan defaultvärde ska stå först i parameterlistan.

Rita upp ett objekt



Hand

- Klassen Hand representerar en korthand

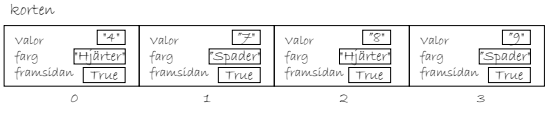


- Klassens attribut är `korten` (en lista med kort).
- Metoderna är `__init__`, `__str__`, `bort` (tar bort korten), `stoppaIn` (lägger till ett kort), `ge` (ger ett kort till en annan korthand)

```
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:



Arv

- *Arv* låter oss återanvända klasser!
- 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`: `fyll`, `blanda` och `delaUt`.

```
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!"
```