Objektorientering 2

vt 25

Innehåll

Repetition

Klassen Car

```
1  class Car():
2      def __init__(self, brand, year, color):
3          self.brand = brand
4          self.year = year
5          self.color = color
6      def drive(self):
7          print(self.brand + ": Kör framåt")
8      def honk(self):
9          print(self.brand + ": Tut tut!")
10     def breaking(self):
11         print(self.brand+ ": Bromsar...")
```

Repetition

Begrepp

- `class Car` är en ny *datatyp* som vi har skapat
- `__init__` är en *konstruktor*
- `def drive(self):` är en *metod*
- `self.brand` är en *instansvariabel*

Repetition

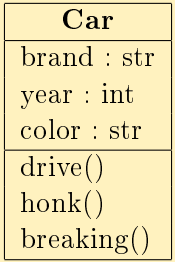
Skapa en instans

```
1 bil1 = Car("Volvo", 2018, "Vit")
2 bil2 = Car("BMW", 2005, "Black")
3
4 bil1.honk()
5 bil2.drive()
```

```
Volvo: Tut tut!
BMW: Kör framåt
```

Klassen Car

Klassdiagram



Klassen Car

self

Som du märkt inleds varje *metod* med parametern `self`. Exempelvis `honk(self)`

```
1     def honk(self):
2         print(self.brand+": Tut tut!")
```

Men `self` dyker inte upp i *metodanropet* senare.

```
1 bil1.honk() # Inget mellan paranteserna
```

Det är för att Python skickar med en *referens* till *instansen* varje gång man anropar en *metod*.

Klassen Car

self

Som du säkert också har märkt så står det `self.` framför alla *instansvariabler*.

```
1     def honk(self):
2         print(self.brand+": Tut tut!")
```

Innehåll

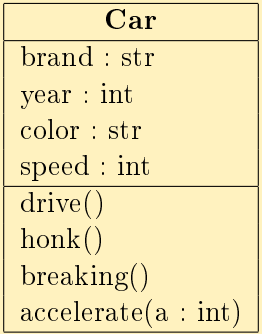
Ändra i klasser

Klassen bil

```
1  class Car():
2      def __init__(self, brand, year, color):
3          ...
4          self.speed = 0 # Bilen står still
5      def drive(self):
6          print(self.brand+": kör "+str(self.speed)+" km/h framåt
               ")
7      def honk(self):
8          print(self.brand+": Tut tut!")
9      def breaking(self):
10         self.speed = 0
11         print(self.brand+": Bromsar...")
12     def accelerate(self, a):
13         self.speed += a
```

Ändra i klasser

Klassdiagram



Innehåll

Klassdiagram

- För att beskriva vad en klass ska innehålla använder man av något som kallas för klassdiagram (UML).
- I ett klassdiagram så har man tre rutor.
- Den översta rutan innehåller klassens namn.
- Den andra rutan innehåller klassens *attribut* (variabler).
- Den nedersta rutan innehåller alla klassens *metoder*.

Klassdiagram

Attributen

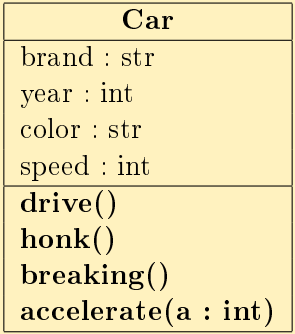
- I diagramet anger man varje attributs datatyp, (int, float, str m.m.)
- Det skrivs vanligtvis: `attribut: datatyp`

Car
brand : str **year : int** **color : str** **speed : int**
drive() honk() breaking() accelerate(a : int)

Klassdiagram

Metoderna

- I diagramet anger man vad varje metod tar emot och skickar tillbaka
- Det skrivs vanligtvis: `metod(arg1: datayp, arg2: datatyp): datatyp`



Innehåll

Övningar

1. Skapa klasserna `Karaktär`, `Vapen` och `Sköld` enligt klassdiagramen i `oot2.py`
2. Följ resten av instruktionerna i filen.