1. Objektorienterad programmering med Python

DA361A

Agenda

- Repetition
- OOP

Repetition

Värden, typer och variabler

- Vad är ett värde?
- Vad är en typ?
- Vad är en variabel?

Operatorer och uttryck

- Vad är en operator?
- Vad är ett uttryck?

Kontrollstrukturer

- Vad är en `if`-sats?
- Vad är en loop?

Kontrollstrukturer

- Vad är en `if`-sats?
- Vad är en loop?

Datastrukturer

- Vad är en `list`?
- Vad är en `dict`?

Funktioner

- Vad är en funktion?
- Hur skapar vi en funktion?
- Vad är ett argument?
- Varför använder vi oss av funktioner?

PEP 8

- Vad är PEP 8
- Varför använder vi detta?

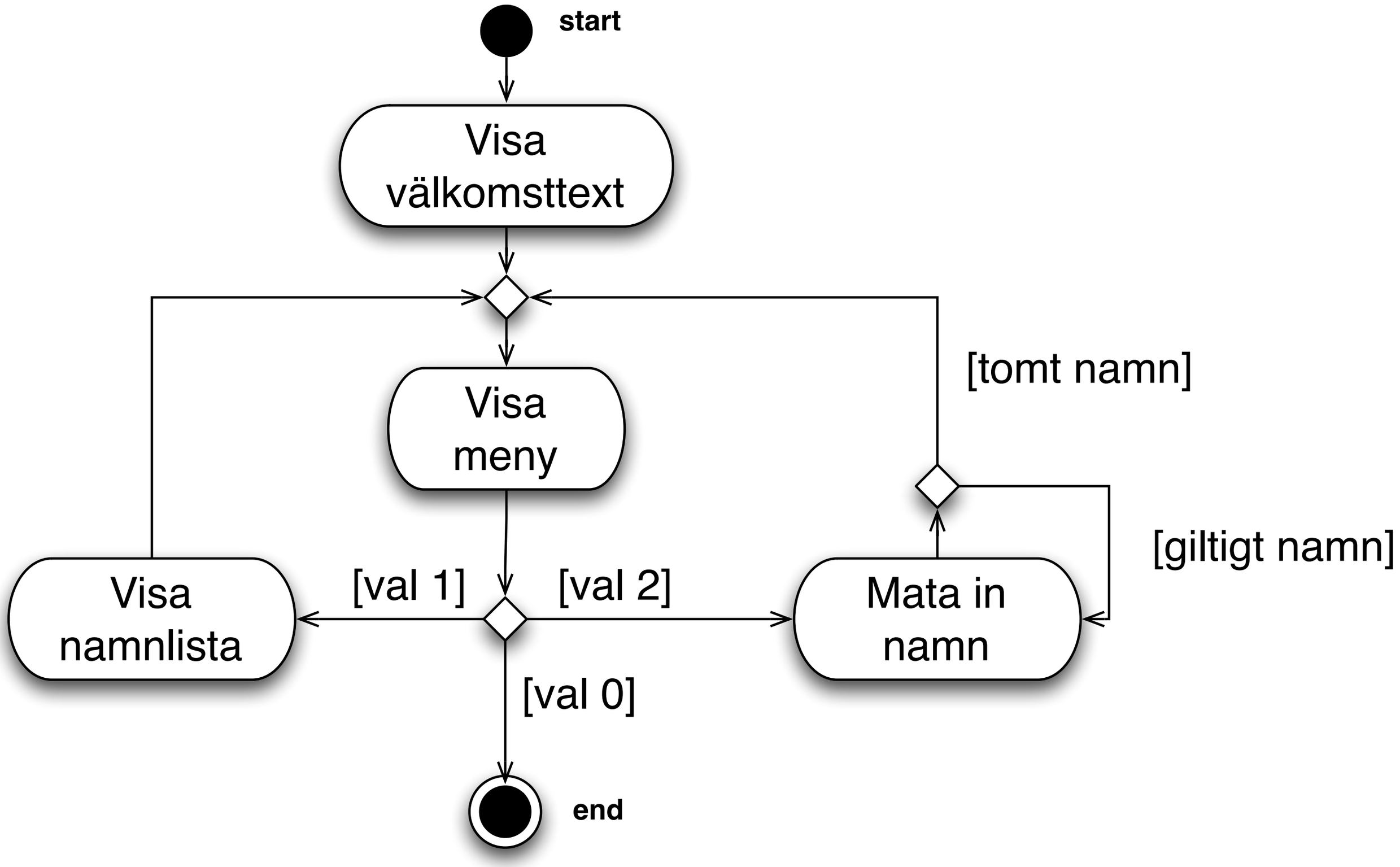
https://www.python.org/dev/peps/pep-0008/

OOP

Vad är ett programmeringsparadigm?

Procedural programming is a method of writing software.

It is a programming practice centered on the *procedures* or *actions* that take place in a program


start
Visa välkomsttext
Visa meny
[tomt namn]
[giltigt namn]
Visa namnlista
[val 1]
[val 2]
Mata in namn
[val 0]
end

Procedural programming is made up of one or more procedures

- Procedures operate on data items that are separate from the procedure
- Data items are passed from one procedure to another
- Focus is on the creation of procedures that operate on the program's data

Objektorienterad programmering

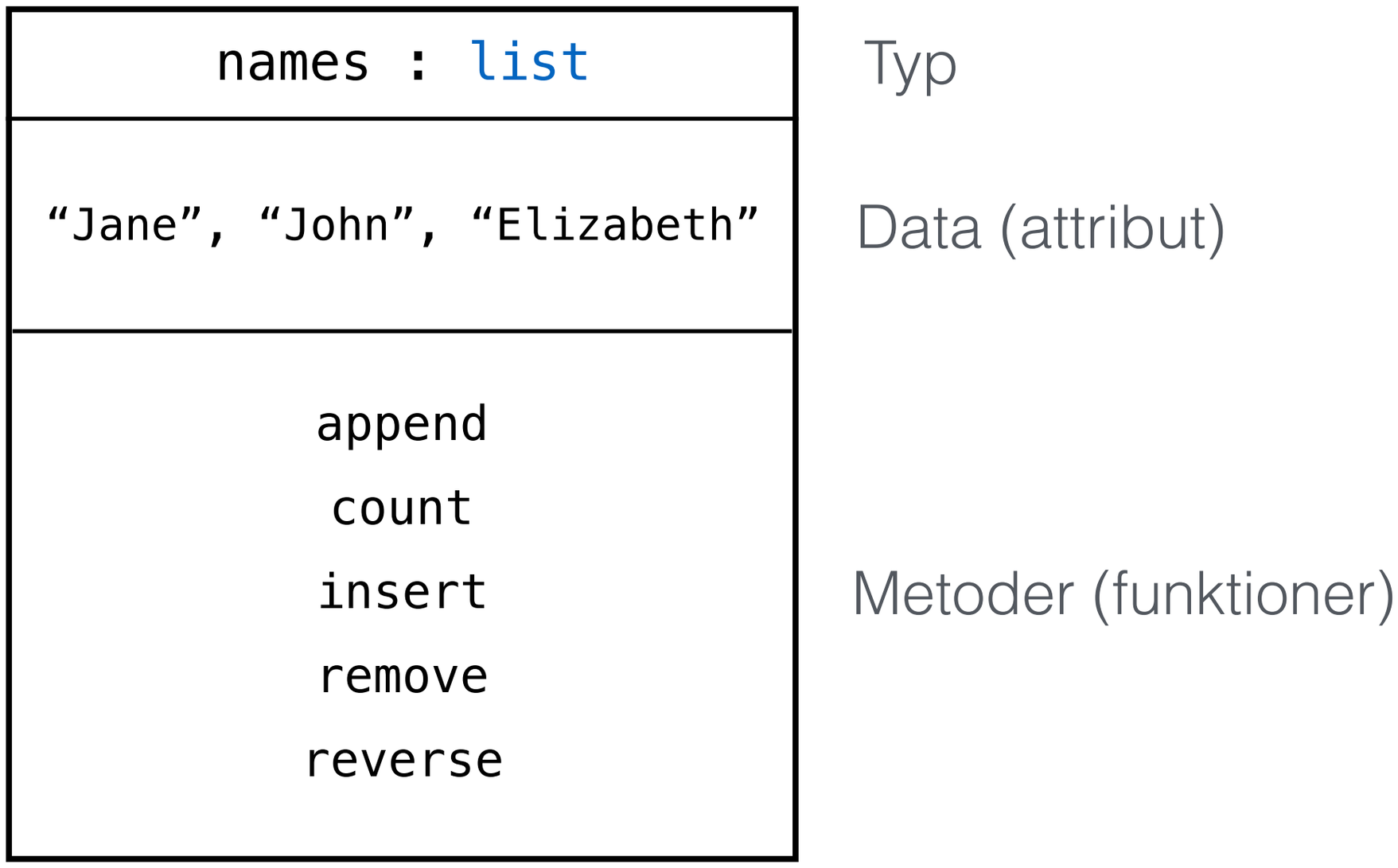
“objekt”, “klass”, “instans”

Object-oriented programming is centered on objects.

Objects are created from abstract data types that encapsulate data and function together.

- An **object** is a software entry that contains **both data and procedures**
- Data contained in an object is known as the object's **data attributes**
- Procedures that an object performs are known as **methods**

```
names = [“Jane”, “John”, “Elizabeth”]
```

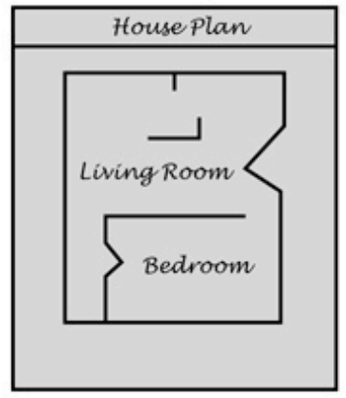


A ***class*** is code that specifies *data attributes* and *methods* for a particular type of data.

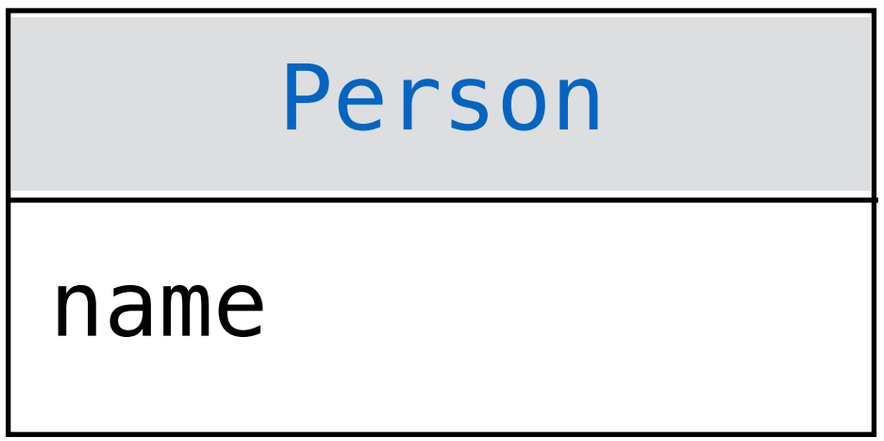
Class

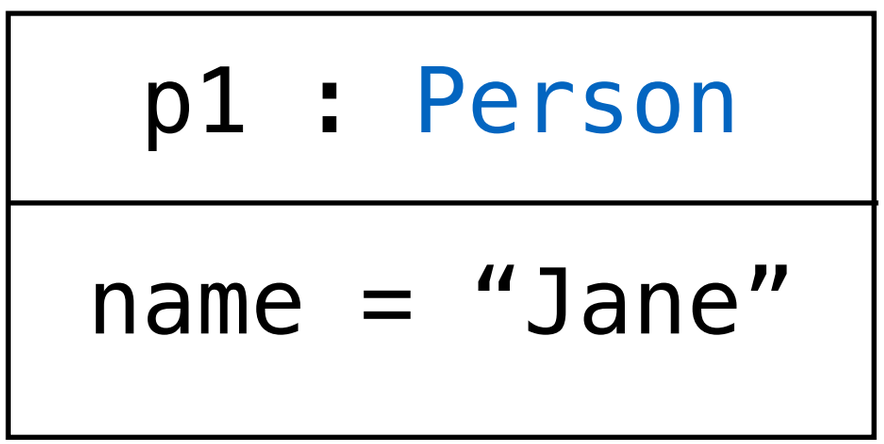
3 objects /
instances /
individuals

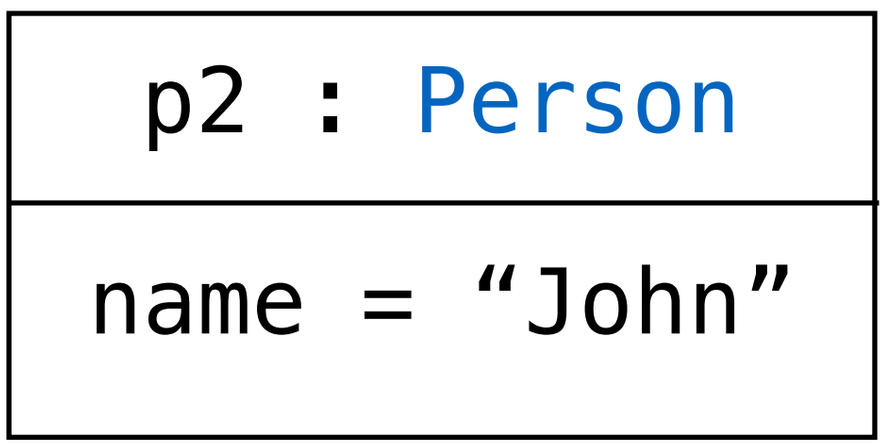
Blueprint that describes a house



Instances of the house described by the blueprint


Person
name


p1 : Person
name = “Jane”


p2 : Person
name = “John”

Class Definition

- It is a set of statements that define a class's methods and data attributes
- Starts with the keyword `class`, followed by the class name
- The `self` parameter is required in every method of the class
  - It is a way of knowing which object's data attributes the method is supposed to operate on
- Most python classes have the `__init__` method
  - Known as the ***initializer method*** because it initializes the object's data attributes

Inkapsling

- Objektet har ett gränssnitt — en tydlig definition över **vad** som kan göras.
- Exakt **hur** saker och ting utförs spelar ingen roll utifrån.
- Men objektet måste ha **kontroll** över sitt tillstånd.

Person
- name : str
+ get_name : str + set_name + say_hello + __str__ : str

```python
class Person(object):

    def __init__(self, name):
        self.name = name

    def get_name(self):
        return self.name

    def set_name(self, name):
        self.name = name

    def say_hello(self):
        print self.name, "says hello!"

    def __str__(self):
        return self.name

```

Synlighet

En objektorienterad princip är att synligheten för egenskaper och operationer ska vara så liten som möjligt. Egenskaper bör ha synligheten ”privat”.

- framför en egenskap eller en operation betyder att endast klassen själv ser (kan använda) denna. Detta kallas för ”privat” synlighet.

\# framför betyder ”protected” och gör att endast klassen och de klasser som ärver av klassen kan se egenskapen eller operationen.

~ framför betyder “package” och är en utökning av “protected” till att även klasser I samma paket kan se egenskapen eller operation.

\+ betyder “publik” och alla kan se egenskapen eller operationen.

Man kan säga att en klass publika gränssnitt är dess publika operationer (inga egenskaper bör vara publika).

MALMÖ HÖGSKOLA

Synlighet inom Python (stack overflow)

http://stackoverflow.com/a/11483397

Accessor and Mutator Methods

- The ***accessor*** (getter) method returns a value from a class's attribute but does not change it.
- The ***mutator*** (setter) method stores a value in a data attribute or changes the value of a data attribute in some other way.