Repetition inför tentamen

DA361A

Vad är UML?

UML

“UML (Unified Modeling Language) is a standard notation for the modeling of real-world objects as a first step in developing an object-oriented design methodology.”

- Standard för att kunna visualisera designen av ett system
  - Olika diagram beroende på vad som behövs visualiseras
- Kan ses som förarbete inför implementation av ett system

Vad är ett klassdiagram?

Klassdiagram

“In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.”

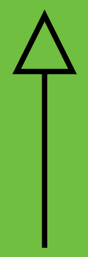
- Klass, attribut och metoder
- Relationer (association, aggregat & komposition, arv)
- Kardinalitet (antalsförhållande)

Relationer

- **Association** defines a relationship between classes of objects that allows one object instance to cause another to perform an action on its behalf ———
- An association with an **aggregation** relationship indicates that one class is a part of another class ———◇
- The **composition** aggregation relationship is just another form of the aggregation relationship, but the child class's instance lifecycle is dependent on the parent class's instance lifecycle ———◆

Relationer forts.

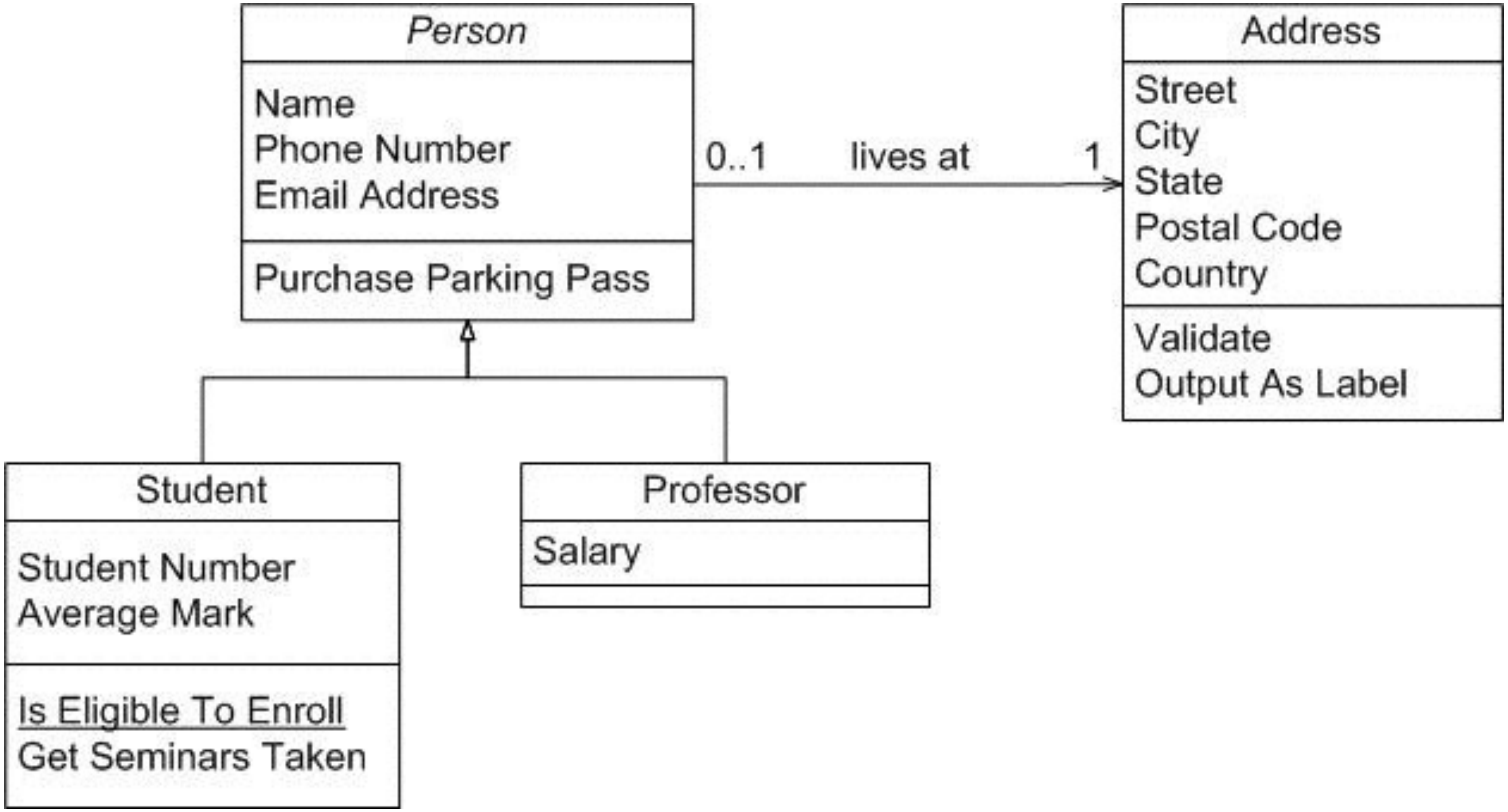
- **Inheritance**, refers to the ability of one class (child class) to inherit the identical functionality of another class (super class), and then add new functionality of its own



Kardinalitet

0..1	Zero or one
1	One only
0..*	Zero or more
*	Zero or more
1..*	One or more
3	Three only
0..5	Zero to Five
5..15	Five to Fifteen

Exempel



Person
Name
Phone Number
Email Address
Purchase Parking Pass
0..1
lives at
1
Address
Street
City
State
Postal Code
Country
Validate
Output As Label
Student
Student Number
Average Mark
Is Eligible To Enroll
Get Seminars Taken
Professor
Salary

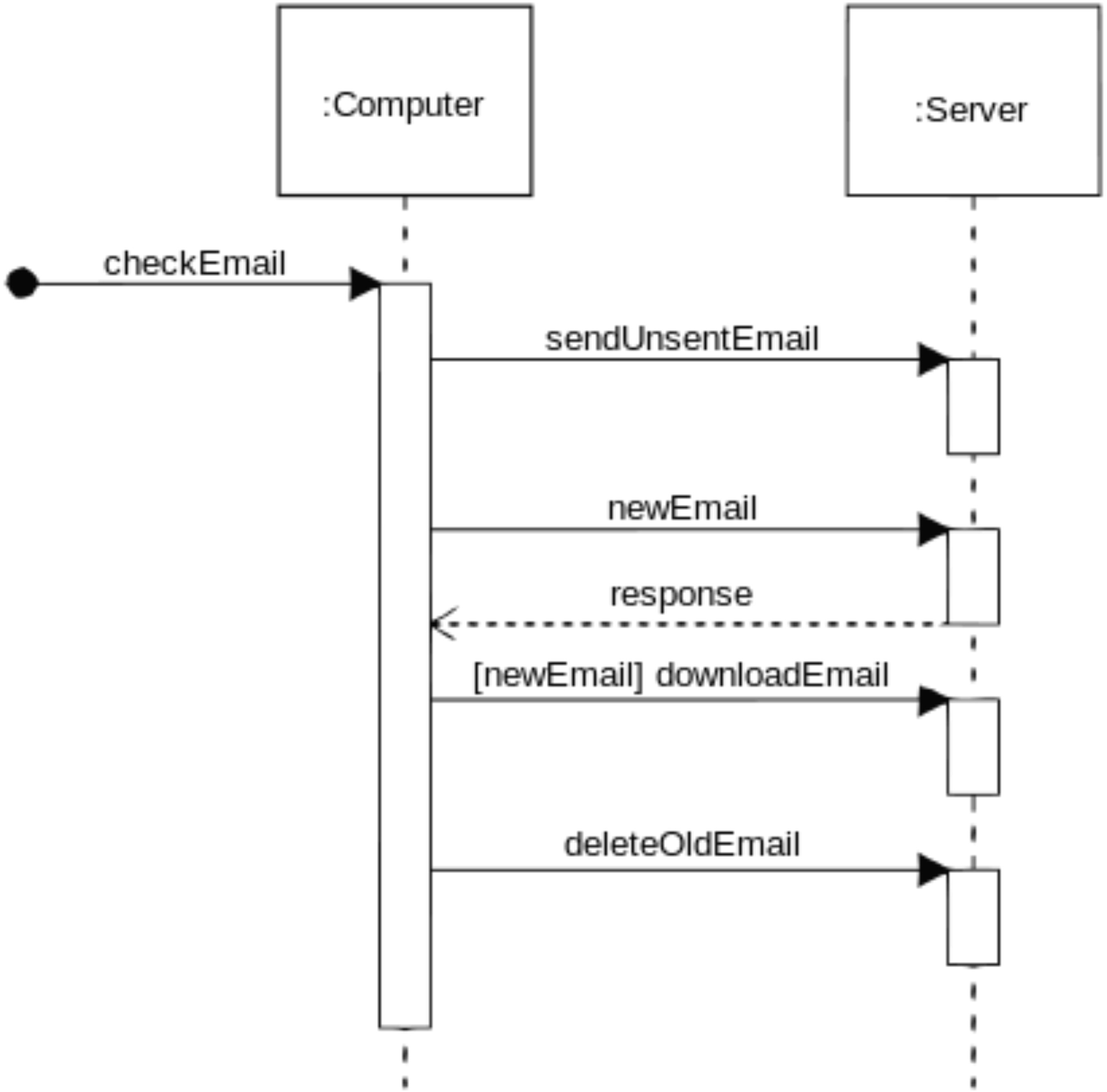
Vad är ett sekvensdiagram?

Sekvensdiagram

"A Sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario."

- Objekt (klasser)
- Metoder
- Ordning (tidslinje)

Exempel


:Computer
:Server
checkEmail
sendUnsentEmail
newEmail
response
[newEmail] downloadEmail
deleteOldEmail

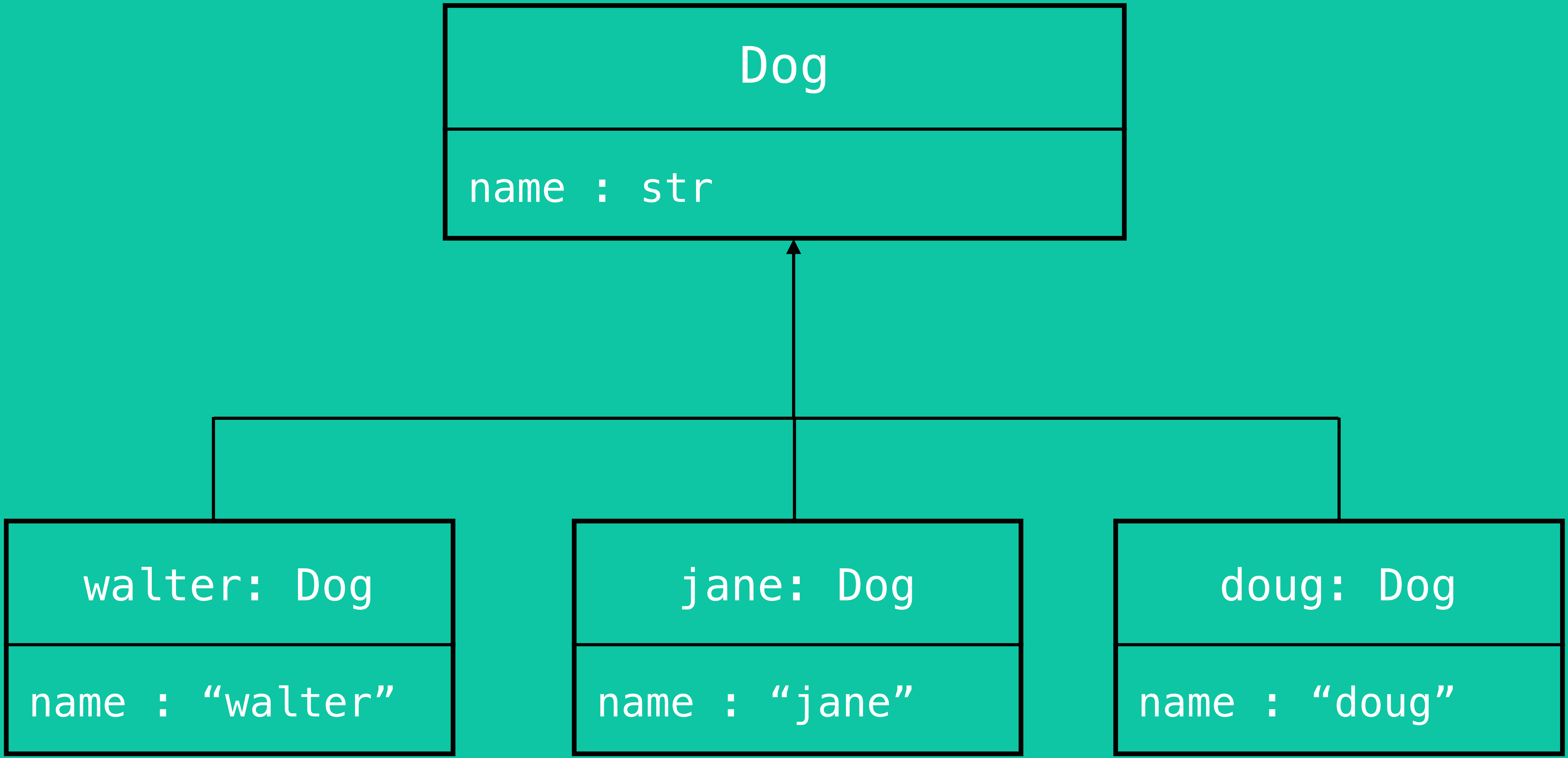
Vad är OOP?

OOP

“Object-oriented programming (OOP) refers to a type of computer programming (software design) in which programmers define not only the data type of a data structure, but also the types of operations (functions) that can be applied to the data structure.”

- Klass och instans
- Inkapsling och synlighet
- Polymorphism

Klass och instans



Dog

name : str

walter: Dog

name : “walter”

jane: Dog

name : “jane”

doug: Dog

name : “doug”

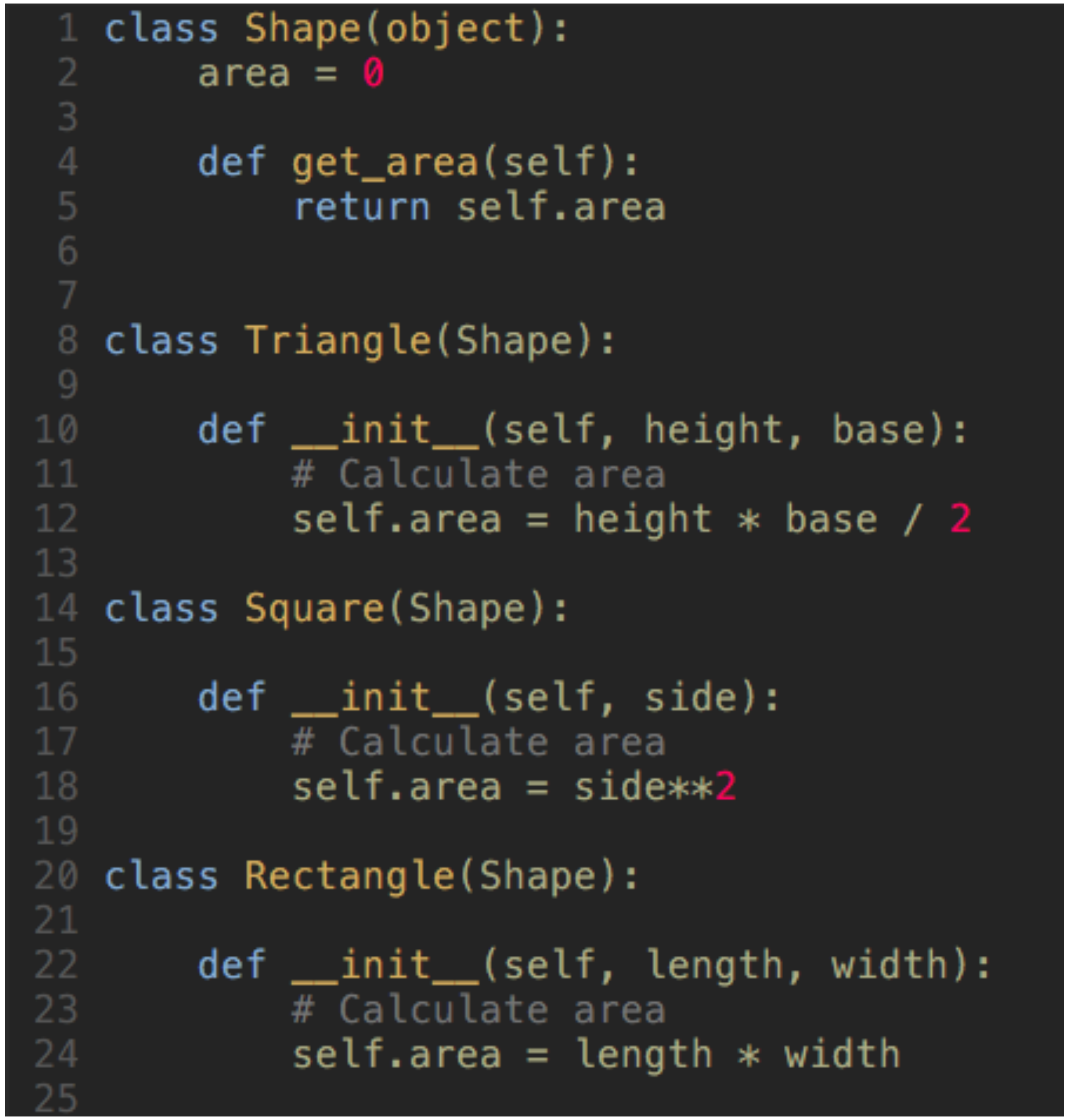
Inkapsling och synlighet

- Ett objekt ska själv hantera och ha kontroll över sitt eget tillstånd
- Tillhandahåller ett gränssnitt över vad som kan utföras

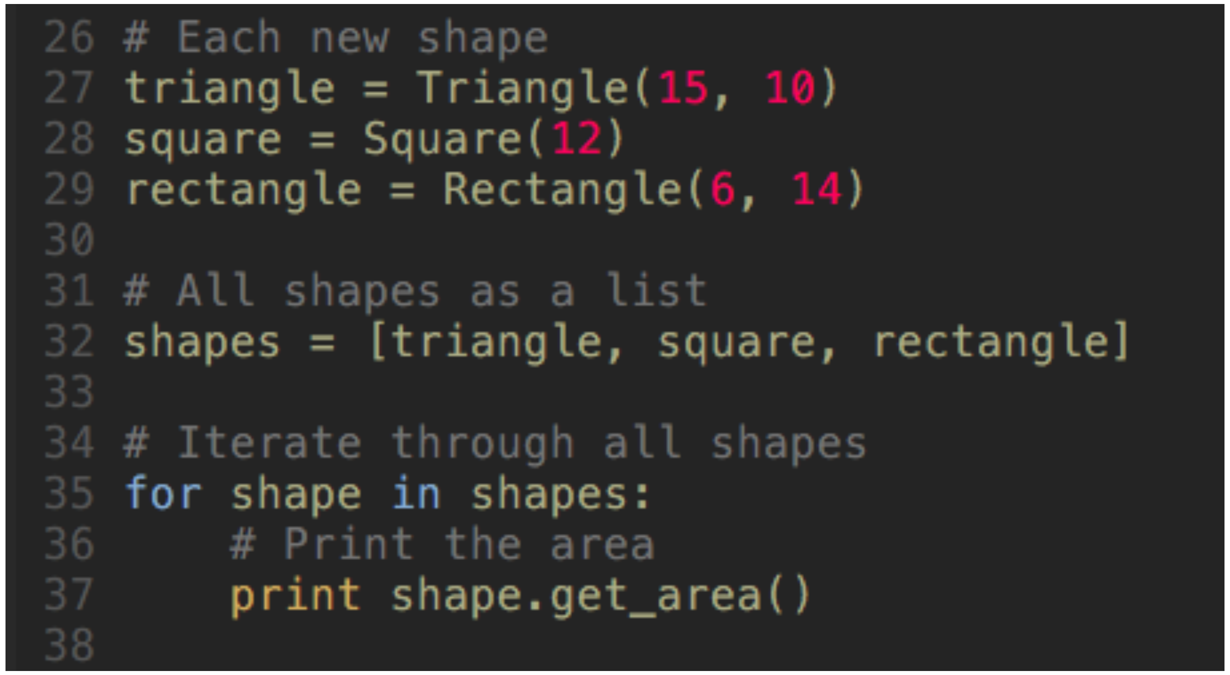
Polymorphism

“Polymorphism describes a pattern in object oriented programming in which classes have different functionality while sharing a common interface.”

Exempel



```python
class Shape(object):
    area = 0

    def get_area(self):
        return self.area


class Triangle(Shape):

    def __init__(self, height, base):
        # Calculate area
        self.area = height * base / 2

class Square(Shape):

    def __init__(self, side):
        # Calculate area
        self.area = side**2

class Rectangle(Shape):

    def __init__(self, length, width):
        # Calculate area
        self.area = length * width

```



```python
# Each new shape
triangle = Triangle(15, 10)
square = Square(12)
rectangle = Rectangle(6, 14)

# All shapes as a list
shapes = [triangle, square, rectangle]

# Iterate through all shapes
for shape in shapes:
    # Print the area
    print shape.get_area()

```

Python och OOP

- Hur implementerar vi en klass (beståndsdelar)?
- Hur implementeras attribut inom en klass?
- Hur implementeras metoder inom en klass?
- Vad innebär “`self`” inom en klass?
- Hur kan vi implementera de olika relationstyperna?
- Hur hanterar vi kardinalitet?
- Hur hanteras inkapsling inom en klass?

Exempel?

Övning

Blackjack

Resurser

- Klassdiagram
  - http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/
- Sekvensdiagram
  - http://www.ibm.com/developerworks/rational/library/3101.html
- Polymorphism
  - http://stackoverflow.com/a/1031385

Frågor?