Data- och informationsvetenskap: Objektorienterad programmering och modellering för IA

DA361A 7,5hp LP2

Lärare i kursen

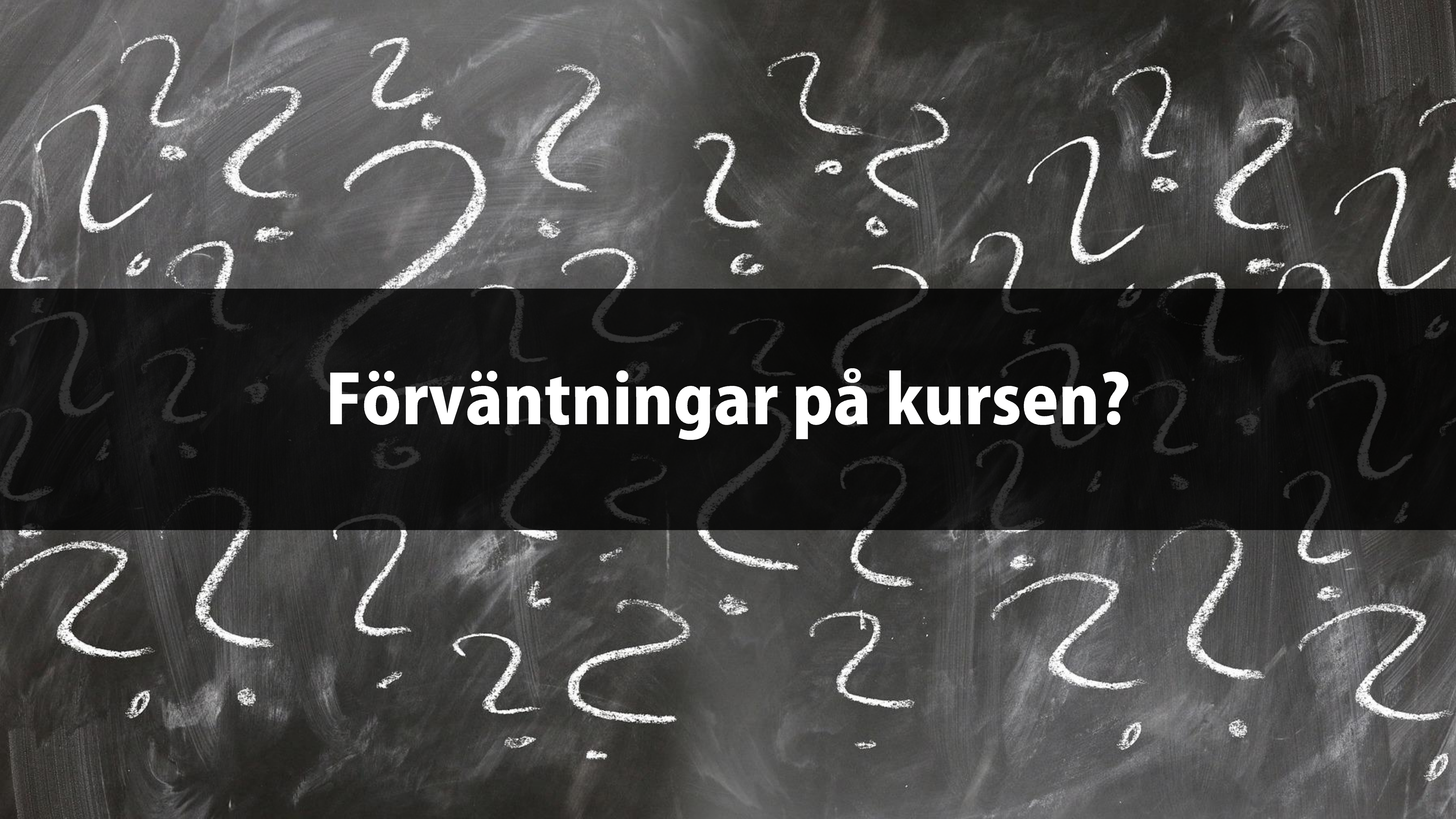
Anton Tibblin

anton.tibblin@mah.se

Aleksander Fabijan

aleksander.fabijan@mah.se

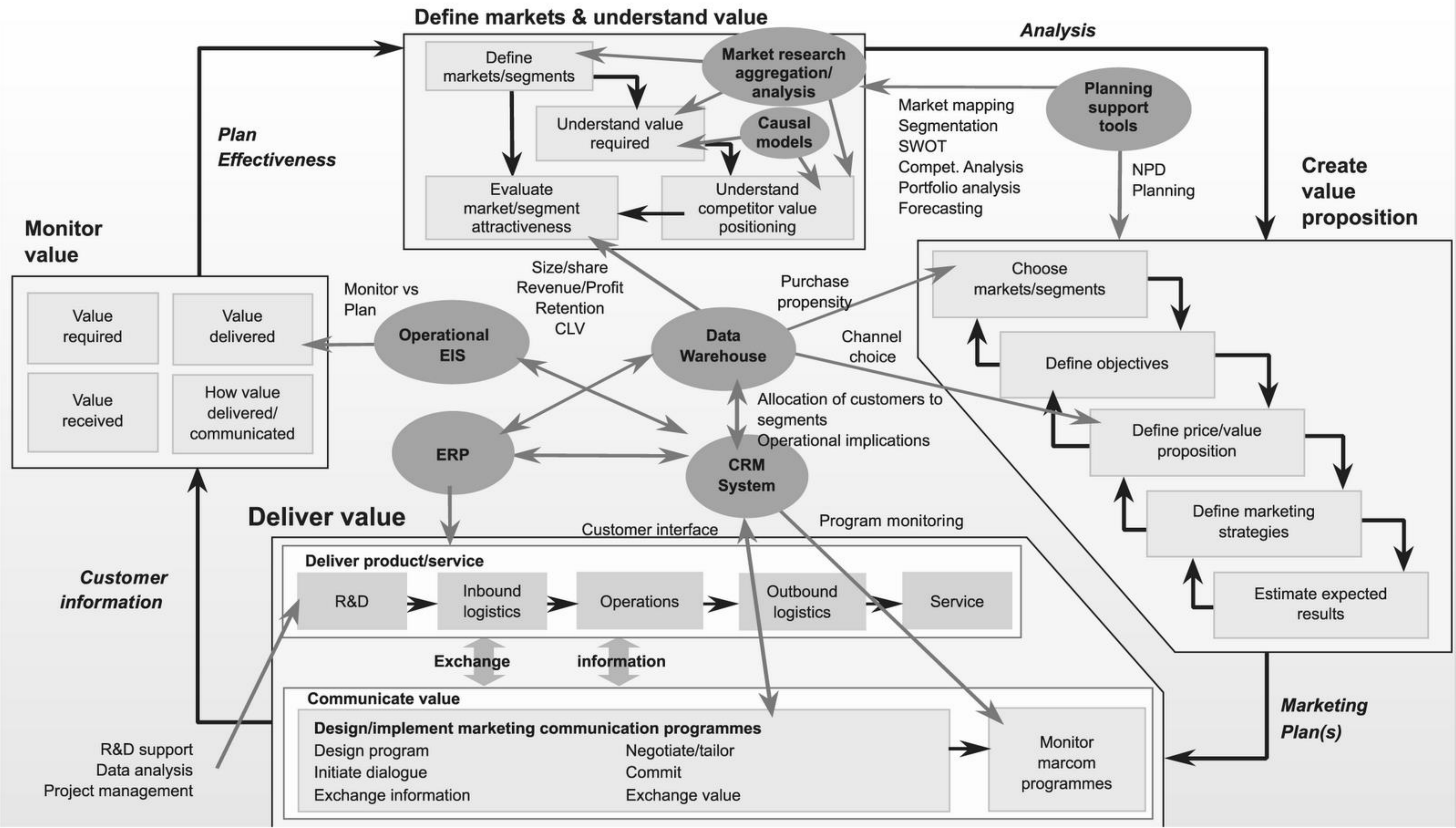
I ♥
PYTHON



Förväntningar på kursen?

Hur går ni till väga idag?

När ni jobbar med era projekt?


Define markets & understand value
Analysis
Define markets/segments
Market research aggregation/ analysis
Understand value required
Causal models
Evaluate market/segment attractiveness
Understand competitor value positioning
Planning support tools
Market mapping
Segmentation
SWOT
Compet. Analysis
Portfolio analysis
Forecasting
NPD Planning
Create value proposition
Plan Effectiveness
Monitor value
Value required
Value delivered
Value received
How value delivered/ communicated
Monitor vs Plan
Operational EIS
Size/share
Revenue/Profit
Retention
CLV
Data Warehouse
Purchase propensity
Channel choice
Allocation of customers to segments
Operational implications
ERP
CRM System
Choose markets/segments
Define objectives
Define price/value proposition
Define marketing strategies
Estimate expected results
Deliver value
Customer interface
Program monitoring
Deliver product/service
R&D
Inbound logistics
Operations
Outbound logistics
Service
Exchange
information
Communicate value
Design/implement marketing communication programmes
Design program
Initiate dialogue
Exchange information
Negotiate/tailor
Commit
Exchange value
Monitor marcom programmes
Customer information
R&D support
Data analysis
Project management
Marketing Plan(s)

Kursplanen

Kursens syfte

Kursens syfte är att studenten ska utveckla kunskaper och färdigheter inom objektorienterad programutveckling och -design. Därigenom ska studenten även vidareutveckla sina programmeringsförmågor.

Innehåll

- Från strukturerad till objektorienterad programmering
- Design och analys med principer för objektorientering
- Objektorientering i det aktuella programspråket (Python)
- Unified Modeling Language (UML)

Innehåll

- Objektorienterad systemanalys och design (OOSAD)
- Objektorienterad programmering (OOP)

Objektorienterad systemanalys och design

“Object-oriented analysis and design (OOAD) is a popular technical approach for **analyzing**, **designing an application**, system, or business by applying the object-oriented paradigm **and visual modeling** throughout the development life cycles to foster better stakeholder communication and product quality.”

Objektorienterad programmering

"Object-oriented programming (OOP) is a programming paradigm based on the concept of **objects**, which are data structures that contain **data**, in the form of fields, often known as **attributes**; and code, in the form of procedures, often known as **methods**."

Kursmaterial

- It’s Learning - inlämningar, resultat, meddelande
  - *http://mah.itslearning.com/elogin/*
- Mah Webb - all annan information
  - *http://da361a.ia-mah.se/*
- Kursplan
  - *http://edu.mah.se/sv/Course/DA361A?v=1#Syllabus*

Kurslitteratur

- Think Python (O'Reilly)
  - ISBN: 1491939362
  - Finns gratis här: http://greenteapress.com/wp/think-python-2e/
- Object-Oriented Systems Analysis and Design Using UML (2010)
  - ISBN: 9780077125363
- *Problem Solving with Algorithms and Data Structures Using Python*
  - *ISBN: 9781590282571*
- http://pythonbooks.revolunet.com/

Schema

Vecka	Moment
45	Kursintroduktion + Föreläsning + WS
46	Föreläsning*2 + Labb
47	Föreläsning + Labb
48	Föreläsning + Labb
49	Föreläsning + Labb
50	Föreläsning + Labb
51	Tenta
1	Labb (extra)
2	Föreläsning + Labb (extra)

Bedömningsformer

- 2st Inlämningsuppgifter, 3.5 hp, U-G
- Tentamen, 4 hp, U-VG

Närvaro!

Frågor?

Python Exercise

- In the previous courses, we learned to use variables. For example:
  - NumberOfTeachers = 2
  - City= "Malmö"
  - Teachers = ['Anton', 'Aleksander'] .
  - Aleksander = {'Name': ‘Aleksander’, 'Age': 27, ‘Teacher:’ ‘’Yes’}

- Exercise: With your existing python knowledge, describe one of your friends in code.

- Exercise 2: Describe another friend in code. Also, change one thing about your first friend (e.g. their age).

One Simple Solution

friend1 = "My first friend is Mladen and he is 23 years old. He plays volleyball and does kickboxing"

friend2 = "My second friend is Lilly and she is 24 years old. She also plays volleybal, however, does not play any other sport"

How are you going to print/change Lilly's age?

Another Simple Solution

```
friend1Name = "Mladen"
friend1Gender = "male"
friend1Age=23
friend1Activities = ['Volleyball','Kickboxing']


friend2Name = "Lilly"
friend2Gender = "female"
friend2Age = 24
friend1Activities = "Volleyball"
```

Challenges with these approaches

- The code can become very long and difficult to read

- It is easy to make mistakes that make our program crash
  - The activity variable with friend1 is a list
  - The activity variable with friend2 is a string
  - When I later try to use these variables, I need to know that I'm operating once with a list, and the second time with a string

A better way: Objects are left Objects

- One way to describe objects from real life in code is to leave them as objects also in the code.

- You make new friends in one line of code by populating their properties.

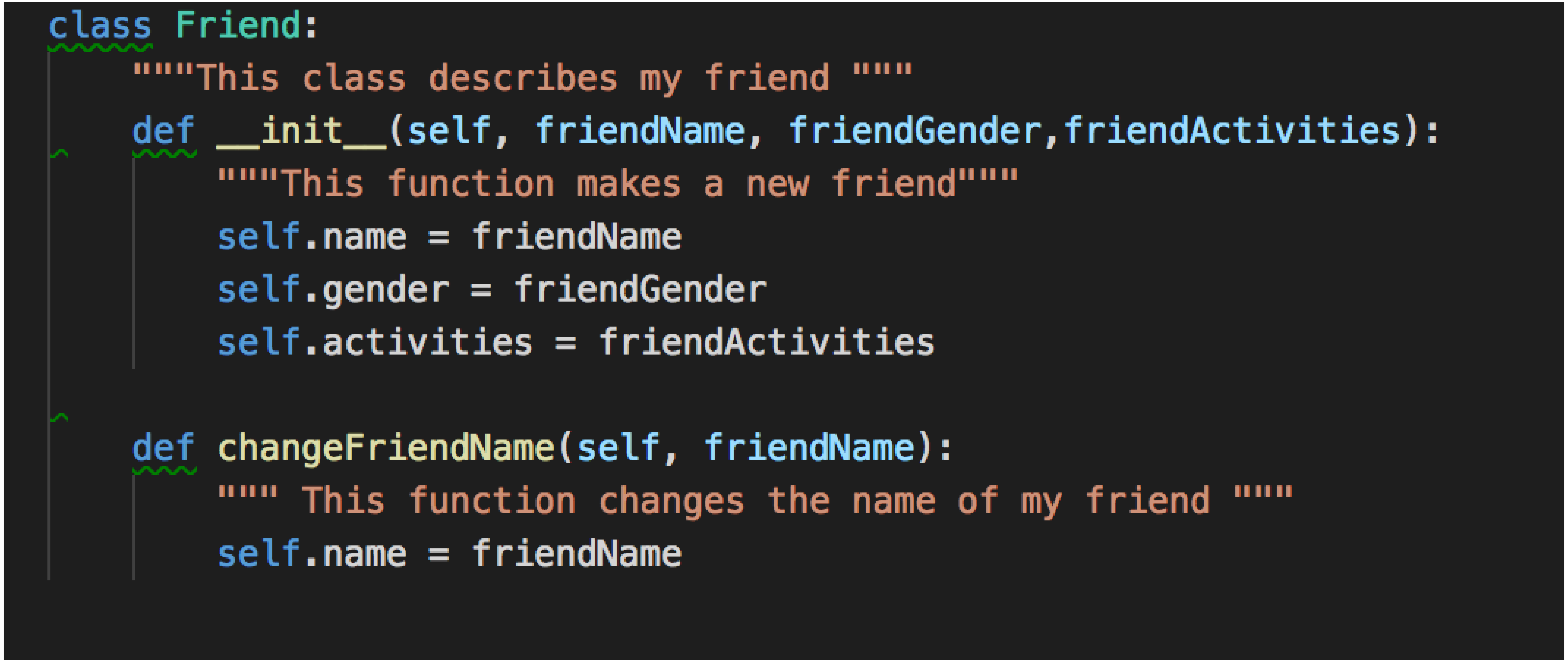
At the end of the day...

- All we want is to make new friends... in one line of code:

```
MyFriendMladen = friend("Mladen", "Male", 23)
MyFriendLilly = friend("Lilly", "Female", 24)
```

Wouldn't making friends this way be much easier?

Example OOP friends in python



```
class Friend:
    """This class describes my friend """
    def __init__(self, friendName, friendGender,friendActivities):
        """This function makes a new friend"""
        self.name = friendName
        self.gender = friendGender
        self.activities = friendActivities

    def changeFriendName(self, friendName):
        """ This function changes the name of my friend """
        self.name = friendName
```

Frågor?

Informationsarkitekt
ESTD
2006
Malmö högskola