Design

OO
Class Diagram
Sequence Diagram

What is the first P.L. you learned?

Object Oriented Programming

Video: https://drive.google.com/open?id=1Sshz2G5EUJouW8Tugn6cU-M6eRHIvY4s

Log in your Uchicago account to access

Object-Oriented Programming, Classes

- Class
  - Data + Operation

- Encapsulation
- Polymorphism
- Inheritance

- Enhance modularity!

Encapsulation

- “the packing of data and functions into a single component. The features of **encapsulation** are supported using classes. It allows selective hiding of properties and methods in a class by building an impenetrable wall to protect the code from accidental corruption.”

Encapsulation

- “the packing of data and functions into a single component. The features of **encapsulation** are supported using classes. It allows selective hiding of properties and methods in a class by building an impenetrable wall to protect the code from accidental corruption.”
- Implication to design?

Polymorphism

- “to process objects differently depending on their data type or class. More specifically, it is the ability to redefine methods for derived classes”
- “the provision of a single interface to entities of different types.”
- Examples

Polymorphism

- “to process objects differently depending on their data type or class. More specifically, it is the ability to redefine methods for derived classes”
- “the provision of a single interface to entities of different types.”

- Implication to design?
- Benefits?
- Problems?

Inheritance

- “a mechanism for code reuse and to allow independent extensions of the original software via public classes and interfaces.”

- Examples

Inheritance

- “a mechanism for code reuse and to allow independent extensions of the original software via public classes and interfaces.”

- Implication to design?
- Benefits?
- Problems?

Class diagram

The video clips for Class Diagram explanation can be found at
Video 1: https://drive.google.com/open?id=11UUWy915XeR7nTnv23W-gsEur2CilTfW
Video 2: https://drive.google.com/open?id=1Y4Myv0J4azfsnZQUChCCzYDITm_jxwA6
Video 3: https://drive.google.com/open?id=1VZVMkjqVASWVECvMjYZUXVssHLpj6pSA
Video 4: https://drive.google.com/open?id=1jotZC4RAiV3UDAOaL_JmJ1CjJfjl_9_j
You need to log into your UChicago account to watch

http://en.wikipedia.org/wiki/Class_diagram

Class diagram

- Describes the types of objects in the system

http://en.wikipedia.org/wiki/Class_diagram

Class diagram

- Describes the types of objects in the system
- Describes the static relationships among them

http://en.wikipedia.org/wiki/Class_diagram

How to decide/design classes?

How to decide/design classes?

- Data
- Operations

How to decide/design classes?

- Data
  - What are the data?
    - Attributes
    - Association
  - What are the properties of the data?
    - Visibility (public or private)
    - Type
    - Default value
    - Is it a container or just a single item?

```
Class student{
 private:
    final string name;
    int age;
    Set enrolledSet<CSClass>;
}
```

How to decide/design classes?

- Operation
  - What are the operations?
  - What are the properties of each operation?
    - Visibility (public or private)
    - Type

```
Class student{
private:
   final string name;
   int age;
   Set enrolledSet<CSClass>;




}
```

How to decide/design classes?

- Relationship among classes
    - Association
    - Inheritance relationship

How to decide/design classes?

- Relationship among classes
    - Association
    - Inheritance relationship
        - Should this class inherits from another class?
        - Should we create a super class for multiple classes?

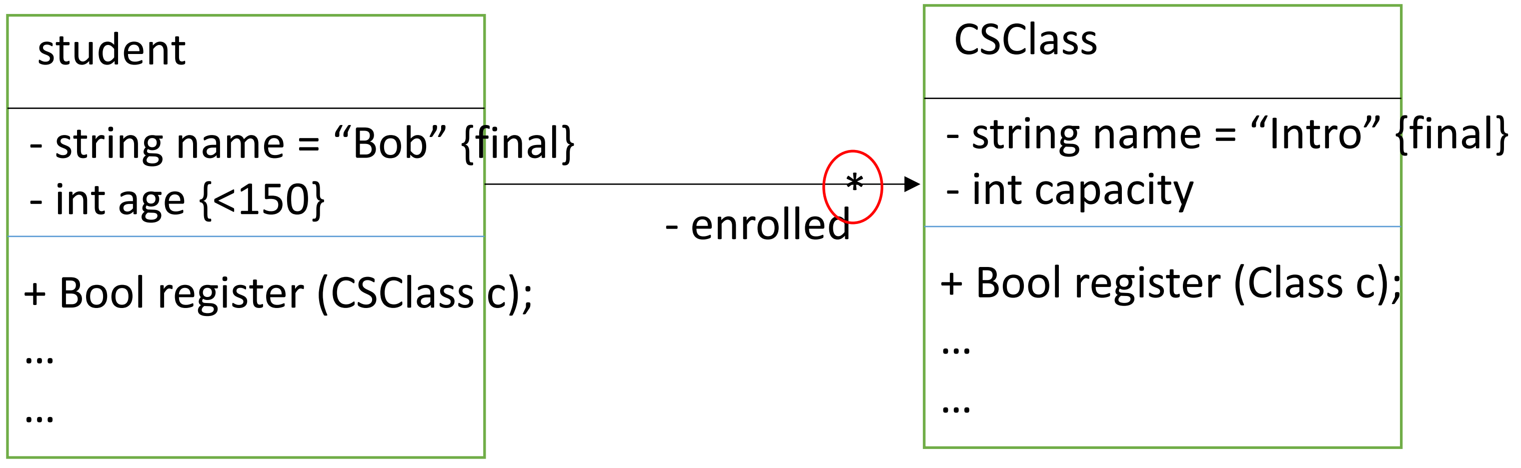
How to represent the class design?

Components of class diagrams

- Class name
- Class properties
  - Attributes
  - Associations (could be bi-directional) $\longrightarrow$

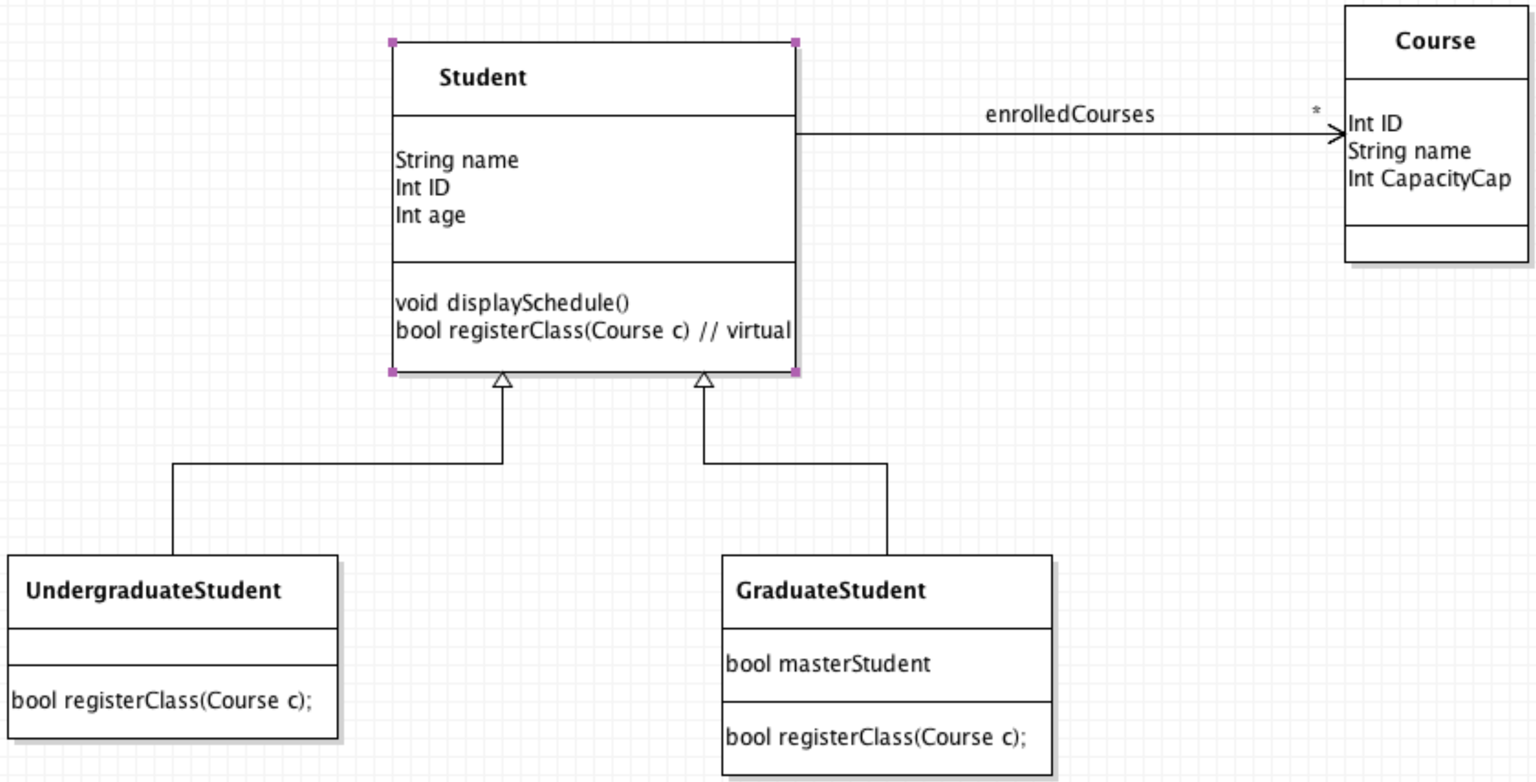
  visibility name : type [multiplicity] = default {property-string}
- Class operations

  Visibility name (parameter list) : return-type {property-string}
- Generalization $\longrightarrow$
  - Inheritance (subclass, super class, interface, …)
- Dependency $\dashrightarrow$
- Constraints {}


student
- string name = “Bob” {final}
- int age {<150}
+ Bool register (CSClass c);
…
…
- enrolled
*
CSClass
- string name = “Intro” {final}
- int capacity
+ Bool register (Class c);
…
…

```
Class student{
private:
   final string name;
   int age;
   Set enrolledSet<CSClass>;
public:
   student (string n, int a);
   bool register (CSClass c);
   …
}
```

- * represents unknown number of CSClass property objects of a student object
- If we put a constant number, like 4, here, we should replace the “Set” data structure into Array



UndergraduateStudent and GraduateStudent are subclasses of Student, and inherit all the attributes and methods of Student.
They both re-implement the registerClass function (polymorphism), and both inherit the super-class' implementation of displaySchedule.

How to turn class diagram to code

- A private attribute → ??
- A * attribute/association → ??
- Class declaration
  - Some attributes may not map to fields

What are the constraints to set?

- Assertion
    - Pre-condition
    - Post-condition
    - Invariant