COSC 2306
Data Programming

OOP and Class

Working on large projects

- A large project is often the result of work of different teams
- Each team is in charge of a section
  - (the other teams don't need to know the details)
- All sections integrate together

What is OOP

- Two basic programming paradigms:
  - Procedural Programming
    - organizing programs around functions or blocks of statements which manipulate data
  - Object-Oriented Programming (OOP)
    - combining data and functionality and wrap it inside what is called an object

Why OOP?

- More intuitive modeling of data/world
  - Better code readability
  - Better code maintainability
- Intensive modularization
  - Better code reusability
  - Better code extensibility
- Advanced features
  - Inheritance

Example

Track student in a class (name, year, ID, email)

fran = [”Frances Allen", "junior", 2300, "fra@uh.edu"]
tom = ["Tom Cruise", "freshman", 2301, "tomc@uh.edu"]
feng = ["Feng Yan", "senior", 2302, "fengy@uh.edu"]
…

Potential issues
- difficult to manage and extend (e.g., index 2 is what? what if I want to add phone number later)
- prone to errors (e.g., what if ID is missing for tom?)

How to perform OOP?

- Know and use the classes and objects
- Class: Abstract of data model
  - Define the information and activities of certain data organization
  - Abstract of object characteristics
- Object: Instance of class
  - Carries real information and methods
  - Follow work patterns defined in the class

Object-Oriented Framework

- Classes and objects are the two main aspects of object oriented programming
- A **class** creates a new *type* using the keyword class
- Where **objects** are *instances* of the class
- An analogy: we can have variables of type int -- variables that store integers are instances (objects) of the int class