Table of contents

Inheritance

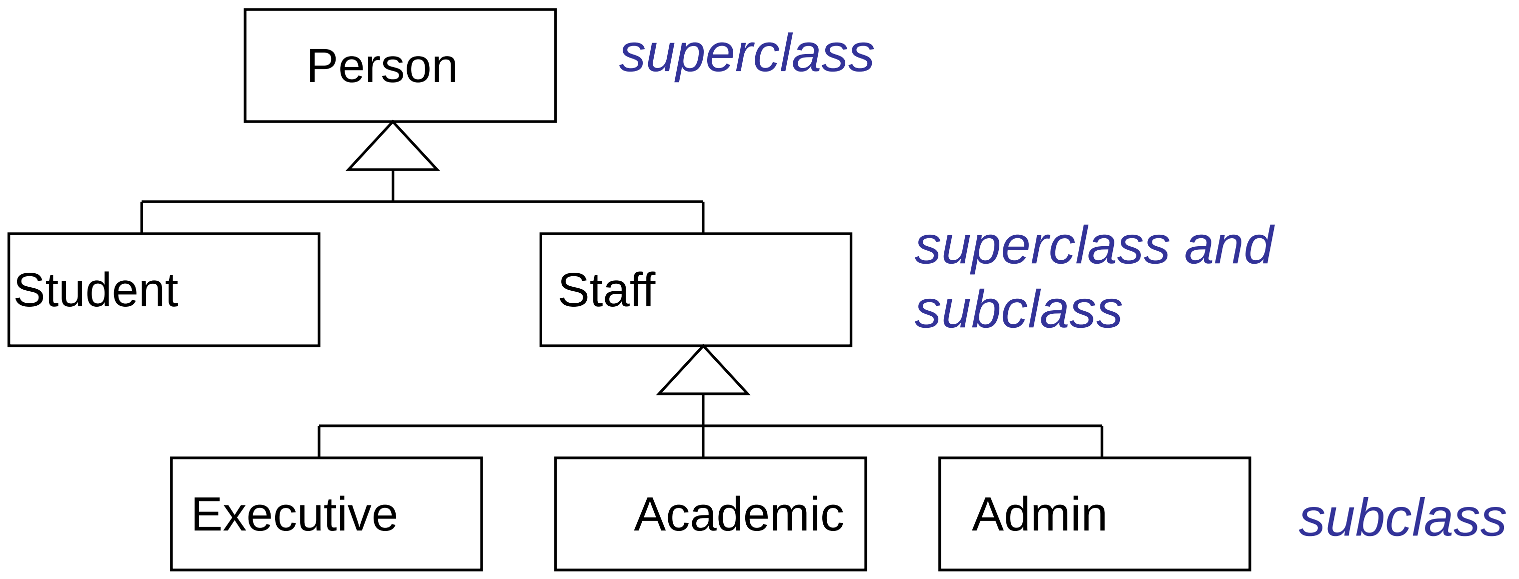
- "a student is a person"
  - a Student class inherits a Person class

- Person is said to be
  - the <u>parent</u> class, <u>super</u> class or <u>base</u> class of Student

- Student is said to be
  - a <u>child</u> class, <u>sub</u> class or <u>derived</u> class of Person

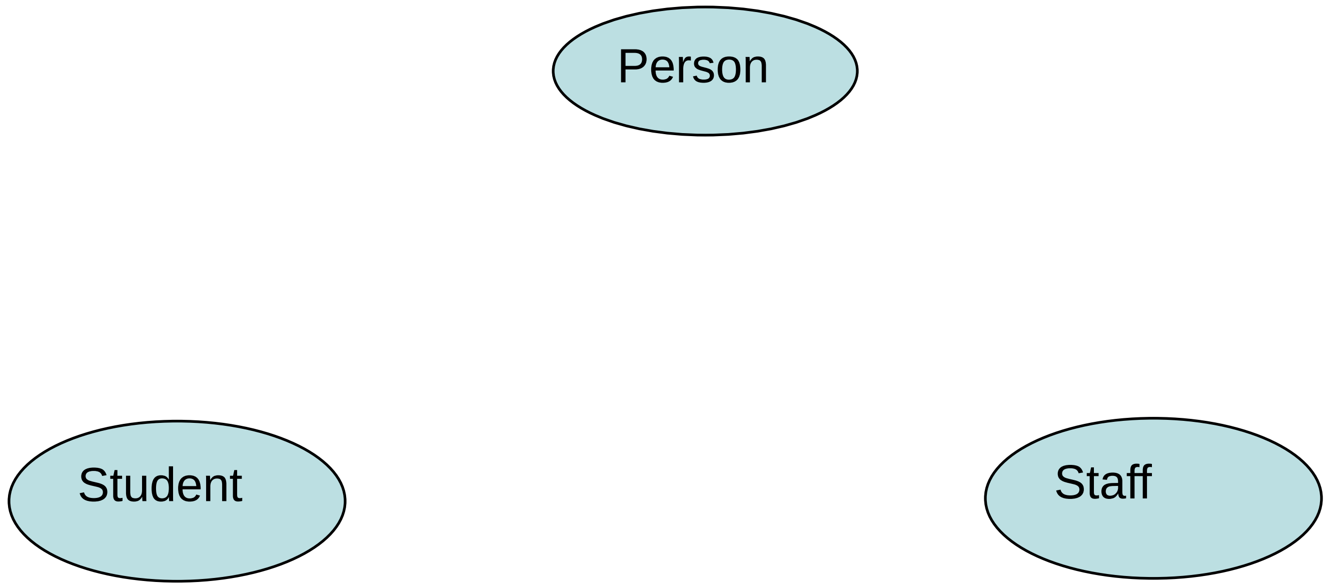
Inheritance

- Class Hierarchy in UML
  - Organize super- & sub-classes into a class diagram (UML)
    - place superclasses on the top of the hierarchy and
    - place subclasses toward the bottom of the hierarchy



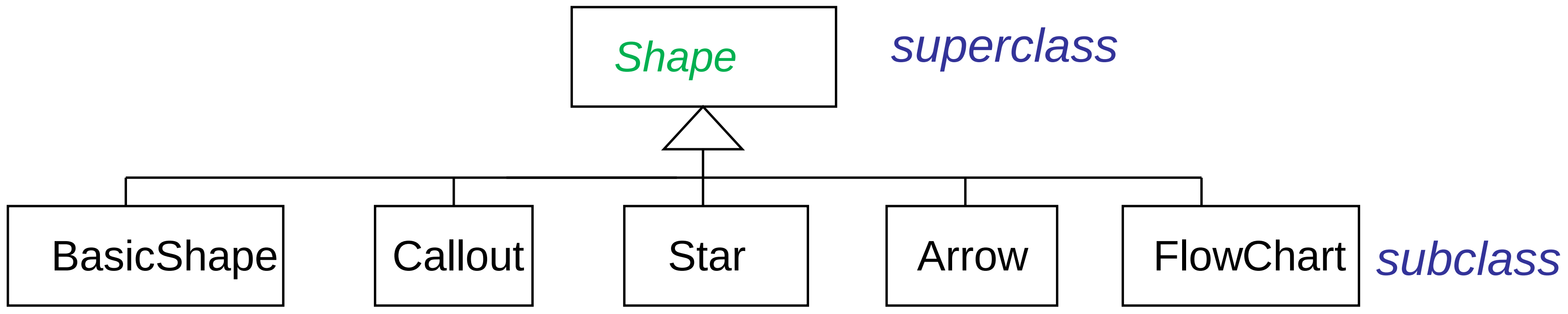
Inheritance

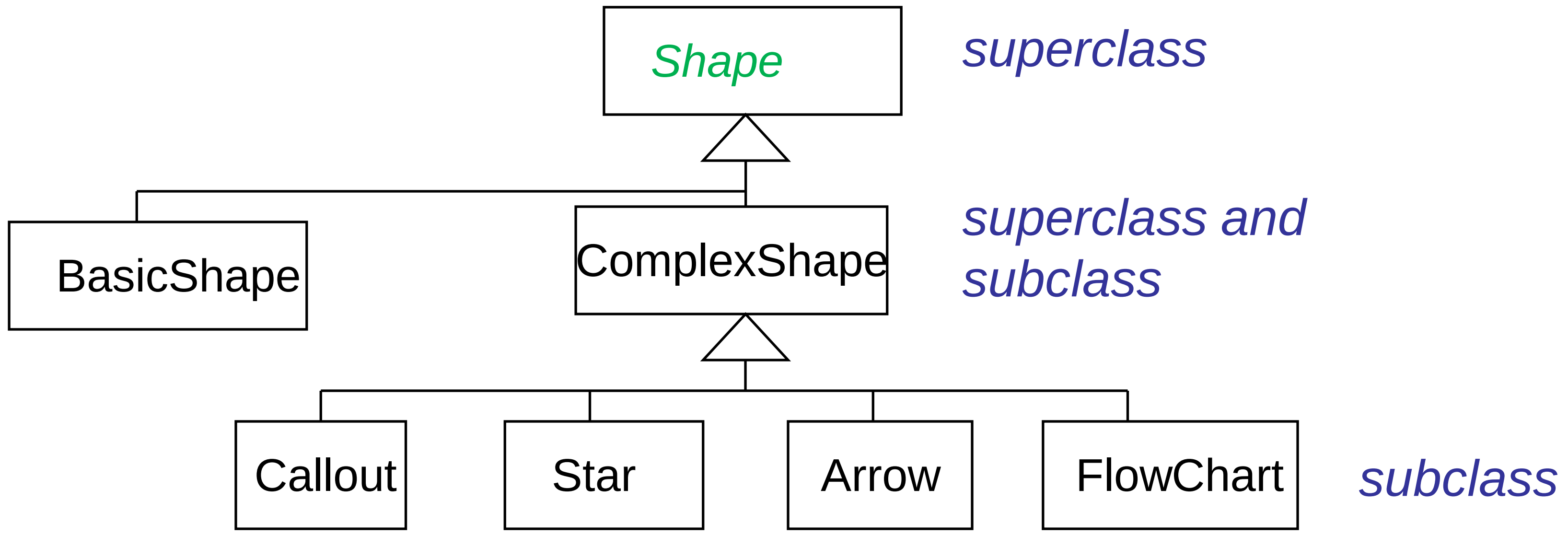
- Two types of formulation:
  - # 1 - Specialization:
    - From Person (parent/super) to Student/Staff (child/sub)
  - # 2 - Generalization:
    - From Student/Staff to Person



Inheritance

- More examples:



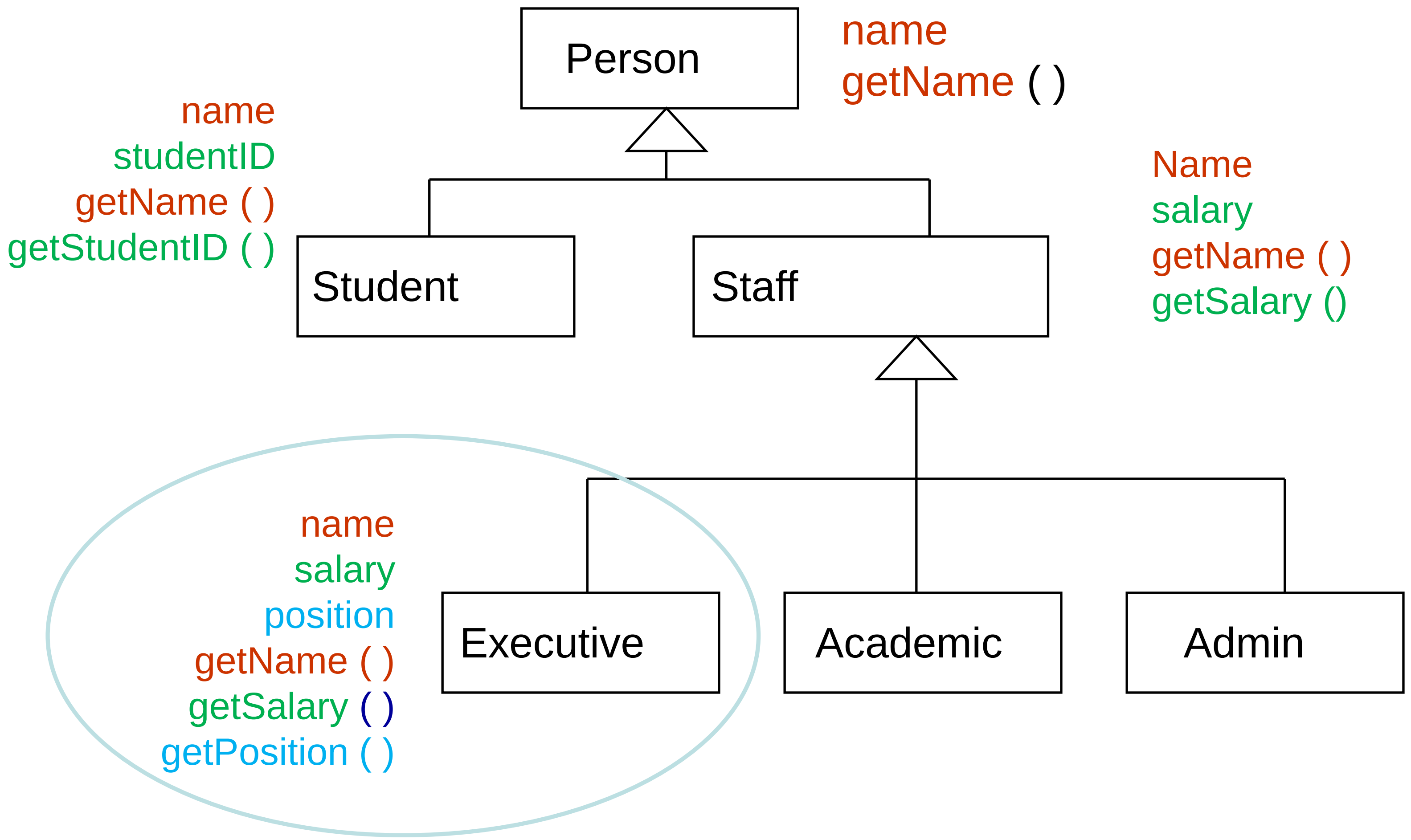


Property inheritance in class hierarchy

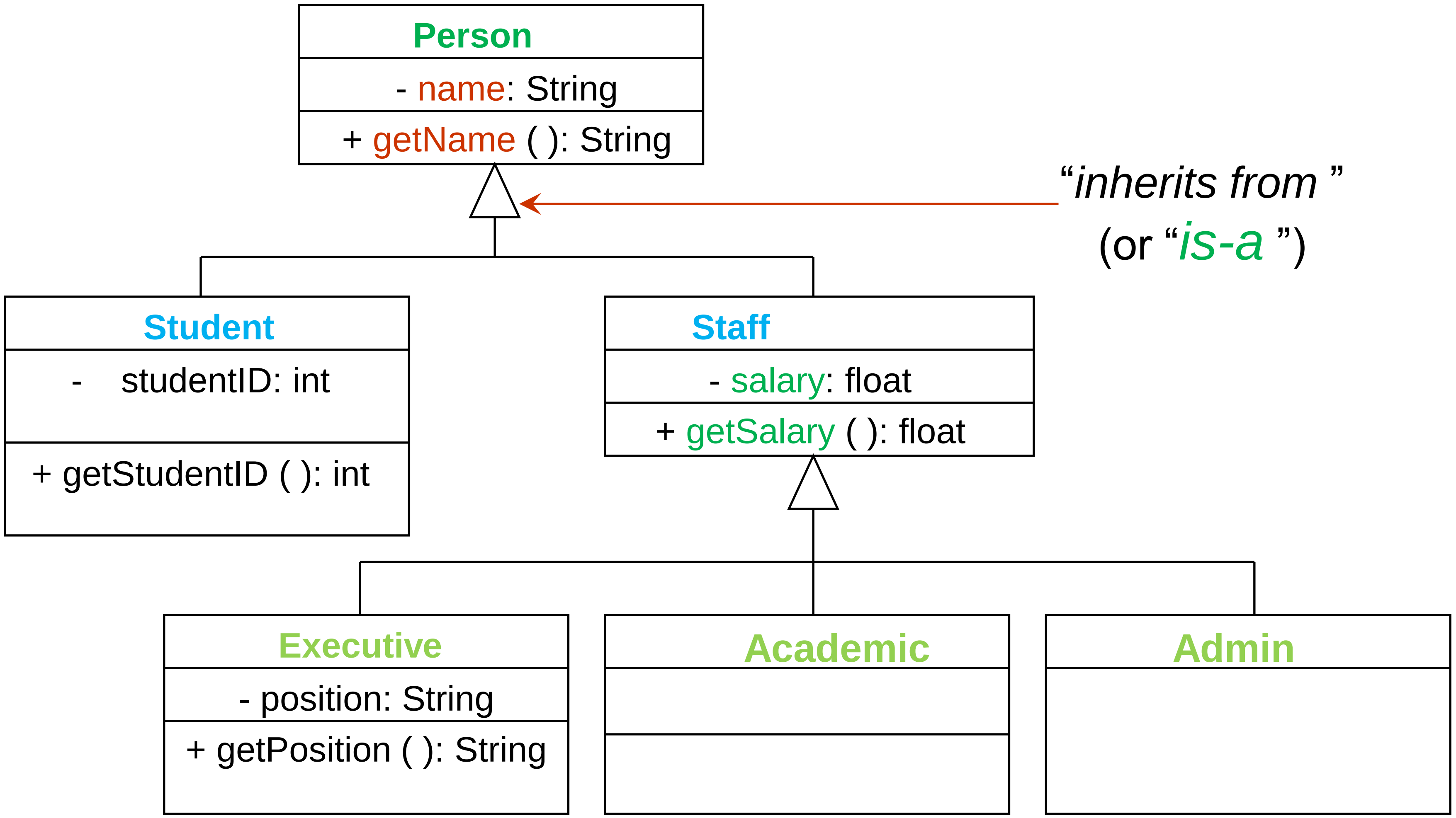
- Subclass can do 3 things with attributes/properties of parent class
  - ► Automatic inheritance
    - ‘Like father, like son’ – color of hair, genes, hobby, IQ, etc.
  - ► Addition
    - New features that do not exist in parent class
      - – my father can’t, but I can!
    - Question: may a child class remove some attributes from parent class?
      - – No, not suggested, it’s a violation of LSP (Liskov substation principle)
  - ► Overriding
    - Subclass can change/alter how parent class handles particular actions (methods)
      - – My father studies, so do I. However, I can change how I study!
      - – A very important concept of OOP to covered more later

Property inheritance in class hierarchy

- common properties (attributes & method)



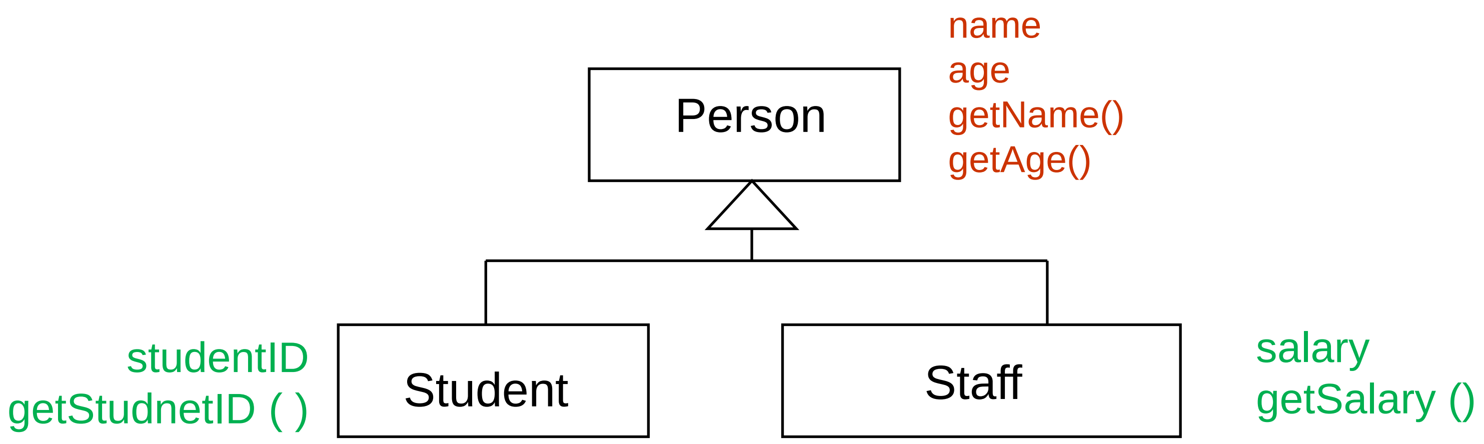
Inheritance: UML modeling



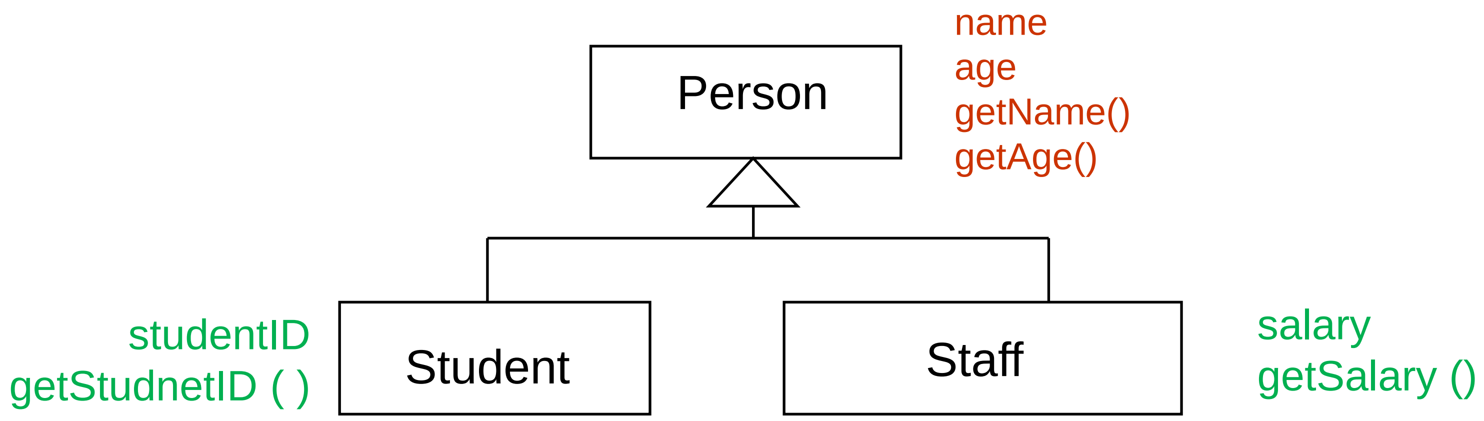
Implementation of inheritance in Java

Inheritance in Java

- Given the class diagram below, let’s see how we may code it in Java.



Inheritance in Java



```
class Person  {
      String name; int age;
      public String getName() { return name; }
      public int getAge() { return age; }
}

class Student extends Person {
      int studentID; public int getID() { return studentID; }
}

class Staff extends Person  {
   float salary; public float getSalary() { return salary; }
}
```

Inheritance

- What is the primary reason for using inheritance when programming?

A. To make a program more complicated
B. To duplicate code between classes
C. To reuse pre-existing code
D. To hide implementation details of a class
E. To ensure pre conditions of methods are met.