Inheritance

- Say we want to implement class RightAngleTriangle:

Right-angled triangles have three vertices (corners), have a perimeter, an area, can move themselves by adding an offset to each corner, and can draw themselves.

Inheritance

Right-angled triangles have three vertices (corners), have a perimeter, an area, can move themselves by adding an offset to each corner, and can draw themselves.

Inheritance

Squares have four vertices (corners), have a perimeter, an area, can move themselves by adding an offset to each corner, and can draw themselves.

Right-angled triangles have three vertices (corners), have a perimeter, an area, can move themselves by adding an offset to each corner, and can draw themselves.

- Sounds very similar, right? • Implementation ... Options?

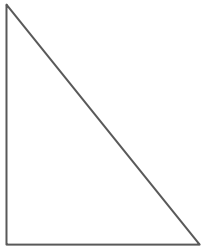
Our (semi-optimal) options

1. Copy-paste-modify Square => RightAngleTriangle

Let’s see in PyCharm

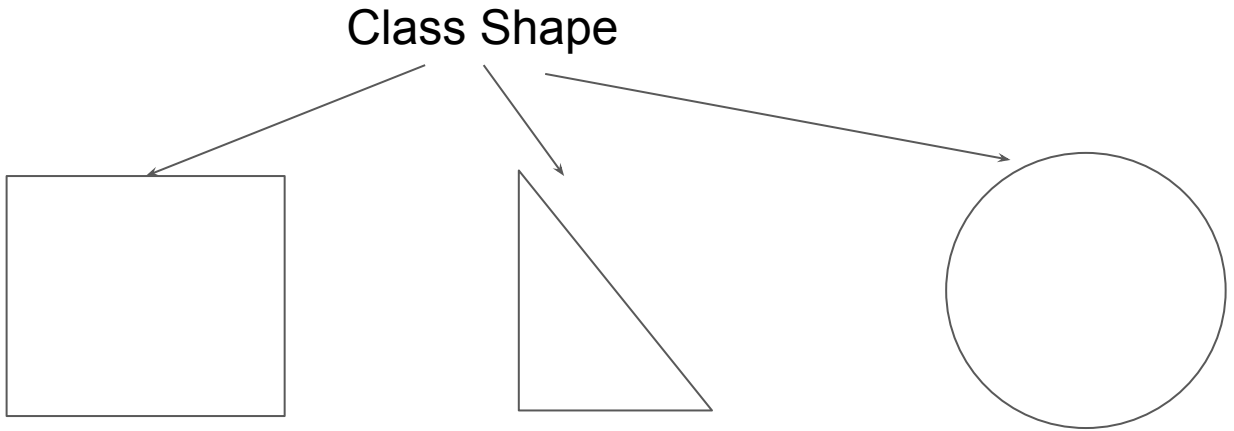
OOP features - overview

- Abstraction
  - A shape has a perimeter (triangle, square, etc.)
    - A square or triangle can inherit the perimeter from a shape
  - A shape has an area (triangle, square, etc.)
    - Can area be abstracted at the shape level?



Inheritance

We really need a general Shape with common features to both Square and RightAngleTriangle (and possibly others)



Abstract class Shape

Most features of Square are *identical* to RightAngleTriangle

- Corners/points, perimeter, area, move, draw
- Differences: class name, code to calculate the area

Key idea:Place common features into class Shape, with unimplemented _set_area as a place-holder...

- Declare Square and RightAngleTriangle as subclasses of Shape, inheriting the identical features by declaring Class Square(Shape): ...

Developing Shape, Square ...

Onto Pycharm

Inherit, override, or extend?

Subclasses use three approaches to recycling the code from their superclass, using the same name

1. Methods and attributes that are used as-is from the superclass are **inherited–** perimeter in Shape

2. Methods and attributes that replace what's in the superclass are **overriden–** example? -- Area in Shape

3. Methods and attributes that add to what is in the superclass are **extended–** example? -- findAngle in Trianlge shape

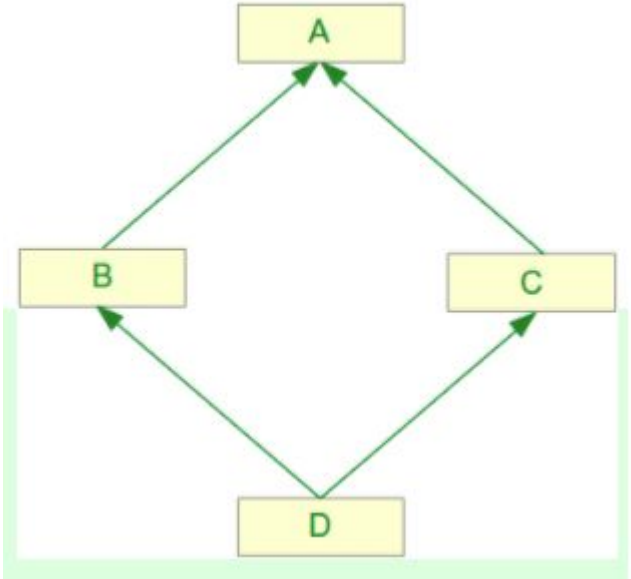
Quiz: Inheritance or Composition

- Fruit, Apple and Peel
- Computer and CPU
- Laptop and Computer
- Person Employee

- Composition and inheritance
  - A square has some vertices (points), so does triangle, etc.
  - A square is a shape, so is a triangle, etc.
- Relationship between has_a and is_a

- Fruit, Apple and Peel
- Computer and CPU
- Laptop and Computer
- Person Employee (A person is_a employee -- but throughout his lifetime??)

Diamond Relationship



Composition or Inheritance?

- Debatable
- As a general rule: Choose Composition over Inheritance
- Composition simpler to understand than inheritance
- In fact, there is a principle : Composition over inheritance (or composite reuse principle)
- https://en.wikipedia.org/wiki/Composition_over_inheritance