Encapsulating Useful Abstractions

- Defining new classes is a good way to modularize a program
- Once some useful objects are identified, the implementation details of the algorithm can be moved into a suitable class definition (*encapsulation*)

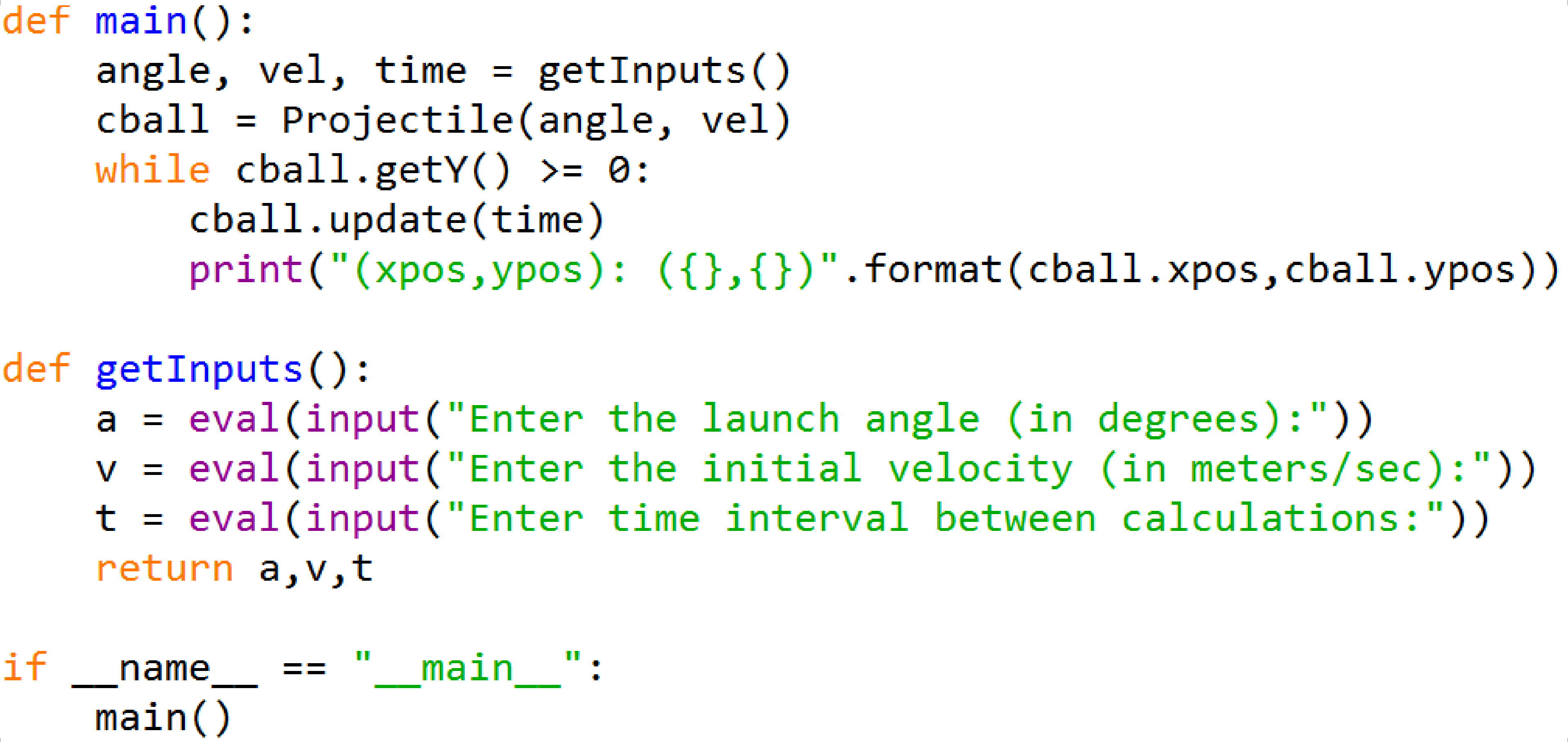
Encapsulating Useful Abstractions

- It allows us to update and improve classes independently without worrying about “breaking” other parts of the program

Object-Oriented Program

```
# cball3.py
from math import pi, sin, cos, radians
class Projectile:
    def __init__(self, angle, velocity):
        self.xpos = 0
        self.ypos = 0
        theta = radians(angle)  # temporary variable
        self.xvel = velocity * cos(theta)
        self.yvel = velocity * sin(theta)
    def update(self, time):
        self.xpos = self.xpos + time * self.xvel
        yvel1 = self.yvel - 9.8 * time  # temporary variable
        self.ypos = self.ypos + time * (self.yvel + yvel1) / 2.0
        self.yvel = yvel1
    def getY(self):
        return self.ypos
    def getX(self):
        return self.xpos
```

Object-Oriented Program

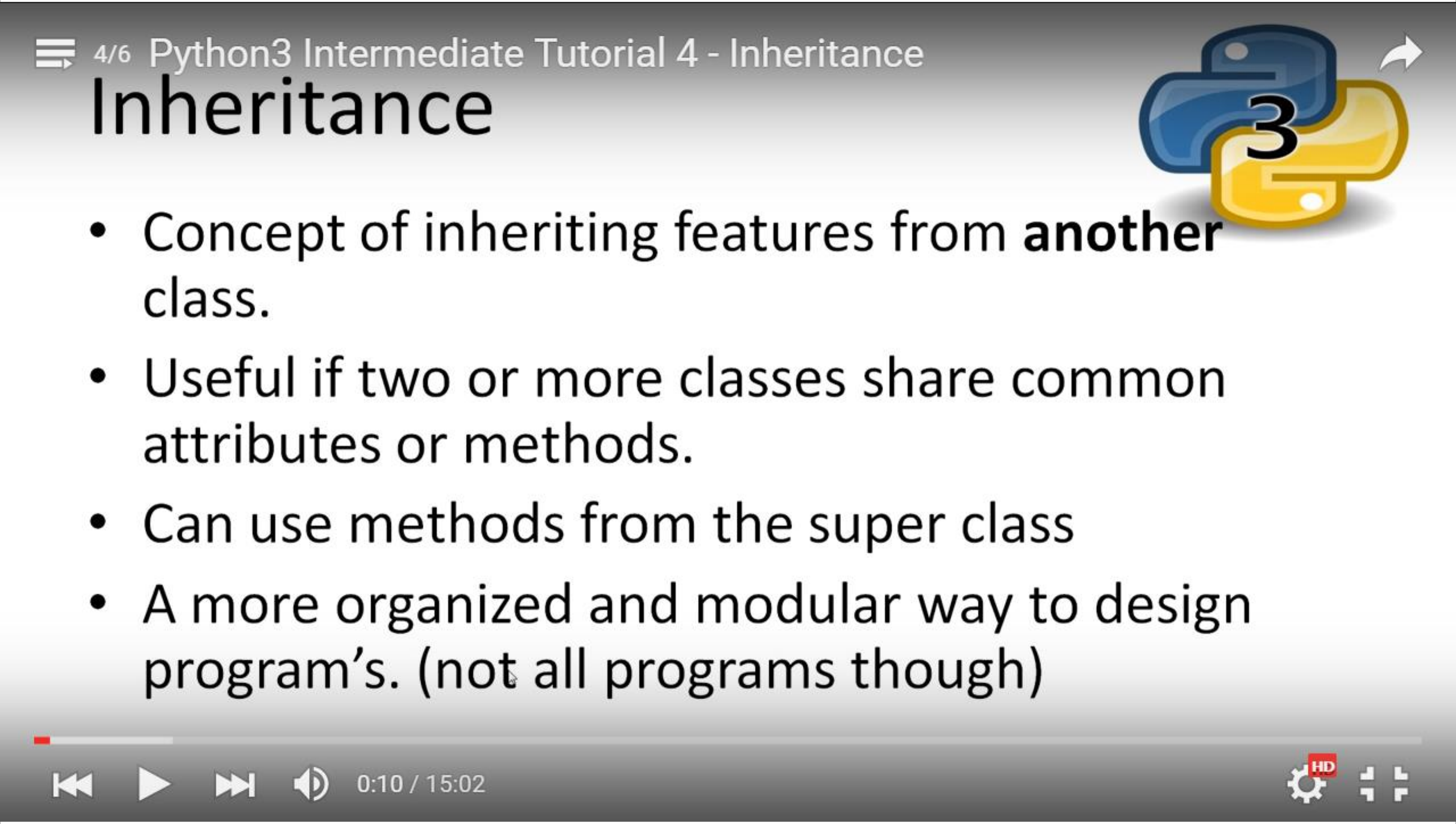


```
def main():
    angle, vel, time = getInputs()
    cball = Projectile(angle, vel)
    while cball.getY() >= 0:
        cball.update(time)
        print("(xpos,ypos): ({},{})".format(cball.xpos,cball.ypos))

def getInputs():
    a = eval(input("Enter the launch angle (in degrees):"))
    v = eval(input("Enter the initial velocity (in meters/sec):"))
    t = eval(input("Enter time interval between calculations:"))
    return a,v,t

if __name__ == "__main__":
    main()
```

Class Inheritance



Inheritance

- Concept of inheriting features from **another** class.
- Useful if two or more classes share common attributes or methods.
- Can use methods from the super class
- A more organized and modular way to design program's. (not all programs though)

PEP 1 Purpose and Guidelines

- Python's development is conducted largely through the **Python Enhancement Proposal** (PEP) process
- The PEP process is the primary mechanism for proposing major new features, for collecting community input on an issue, and for documenting the design decisions that have gone into Python
- PEP 0 Index of Python Enhancement Proposals (PEPs)

PEP 257 Docstring Conventions

- "This PEP documents the semantics and conventions associated with Python docstrings"
- "The aim of this PEP is to standardize the high-level structure of docstrings: what they should contain, and how to say it"
- "The PEP contains conventions, not laws or syntax"
- "If you violate these conventions, the worst you'll get is some dirty looks"

PEP 8 Style Guide for Python Code

- This document gives coding conventions for the Python code comprising the standard library in the main Python distribution

- Examples:
  – Use 4 spaces per indentation level.
  – Tabs or Spaces?
  – Whitespace in Expressions and Statements
  – Comments
  – Naming Conventions