Introduction:

- Python doesn’t require you to use objects or classes.
- Complex programs are hard to keep organized.
- Object-oriented programming organizes and structures code.
- OOP groups together data and behavior into one place.
- OOP promotes modularization of programs.
- Isolates different parts of the program from each other.

OOP Terms:

Class	Blueprint for creating objects of same type.
Methods	Regular functions that are part of a class.
Attributes	Variables that hold data are part of class.
Object	A specific instance of a class.
Inheritance	Means by which a class can inherit capabilities from another.
Composition	Means of building complex objects out of other objects.

Creating a basic class:

```
class Book:
    def __init__(self, title, author, pages, price):
        // This is initializer function.
        self.title = title
        self.author = author
        self.pages = pages
        self.price = price
```

```
//Creating some instance methods.
def getprice(self):
        return self.price

//Creating some book instances.
B1 = Book(“War Peace”, “Leo Tolstoy”, 1225, 39.95)
B2 = Book(“Catcher”, “JD Salinger”, 234, 29.95)

//print the price of book1
print(b1.getprice())
```