# Chapter 6: Object Oriented Programming

Erudition Labs

Computer Science 101: Introduction to Java and Algorithms

May 26, 2019

# Contents

# 1 Object Oriented Programming (OOP)

Back in the day of the ancient Greek philosophers, there was a guy named Plato. His writings are pretty famous especially since he is credited for writing down the teachings of Socrates. One of the questions that Plato was asking (and really all the Greek Philosophers) was, “what is real¿‘. Plato had an interesting approach to this that I think matches up to how we use object oriented programming. For Plato, there was a creator who was basically a mathematician (the ancient Greeks loved math) and there exists some objective truth; some objective reality which he referred to as “the Ideas“ or “the Forms“. So for Plato, all that is real is “the Ideas“. You could think of it almost as some place in space that exists that just hosts all of reality. The “Ideas“ are the perfect mathematical blueprints of reality, and everything else is merely a copy. For example, a chair. You are probably sitting in a chair, but what makes it a chair as opposed to, say a desk? Well a chair has certain properties that belong to it, that a desk doesn‘t (perhaps like a back to lean against). A chair has four legs, a seat, and a back, each leg is at a certain angle, and perhaps has a certain height, etc. For Plato, the chair you are sitting on isn‘t real, but it is a copy of the “Idea“ of a chair. The “Idea“ of the chair is the blueprint and the chair you are sitting on is the manifestation of that idea.

Well, for Object oriented programming, we also have blueprints, we call them “classes“. The manifestations of those blueprints is what we call “objects“.

Object oriented programming (OOP for short) is probably the most widely used programming paradigm of our time due to its intuitiveness. OOP is simply a way or organizing your code so that you can take advantage of design principles that would allow for efficient code reuse and organization. Basically you write the blueprint and then create instances of that blueprint, each with it‘s own copies of the properties and functionalities.

# 2 Classes (Video Series Lecture 40 and 41)

Classes are the blueprints of our objects. It‘s in the class definition that we model the properties and functionalities of our objects. Let‘s just look at an example of a class.

```
class Square {
  private int length;

  Square(int l) {
    length = l;
  }

  public int area() {
    return length * length;
  }

  public int perimeter() {
    return 4 * length;
  }
}
```

# 3 Objects (Video Series Lecture 42 and 43)

# 4 More Keywords (Video Series Lecture 44 - 47)

## 4.1 static

# 5 Overloading(Video Series Lecture 48 and 49)

# 6 Strings(Video Series Lecture 50 and 51)