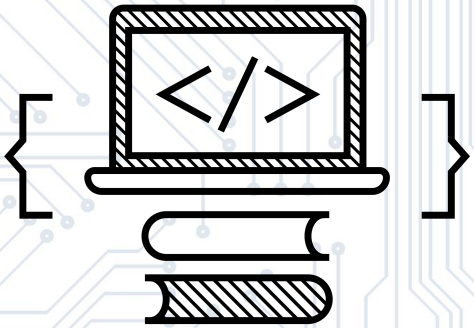
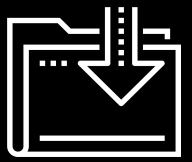


Object-Oriented Programming (OOP)

Web Development Boot Camp

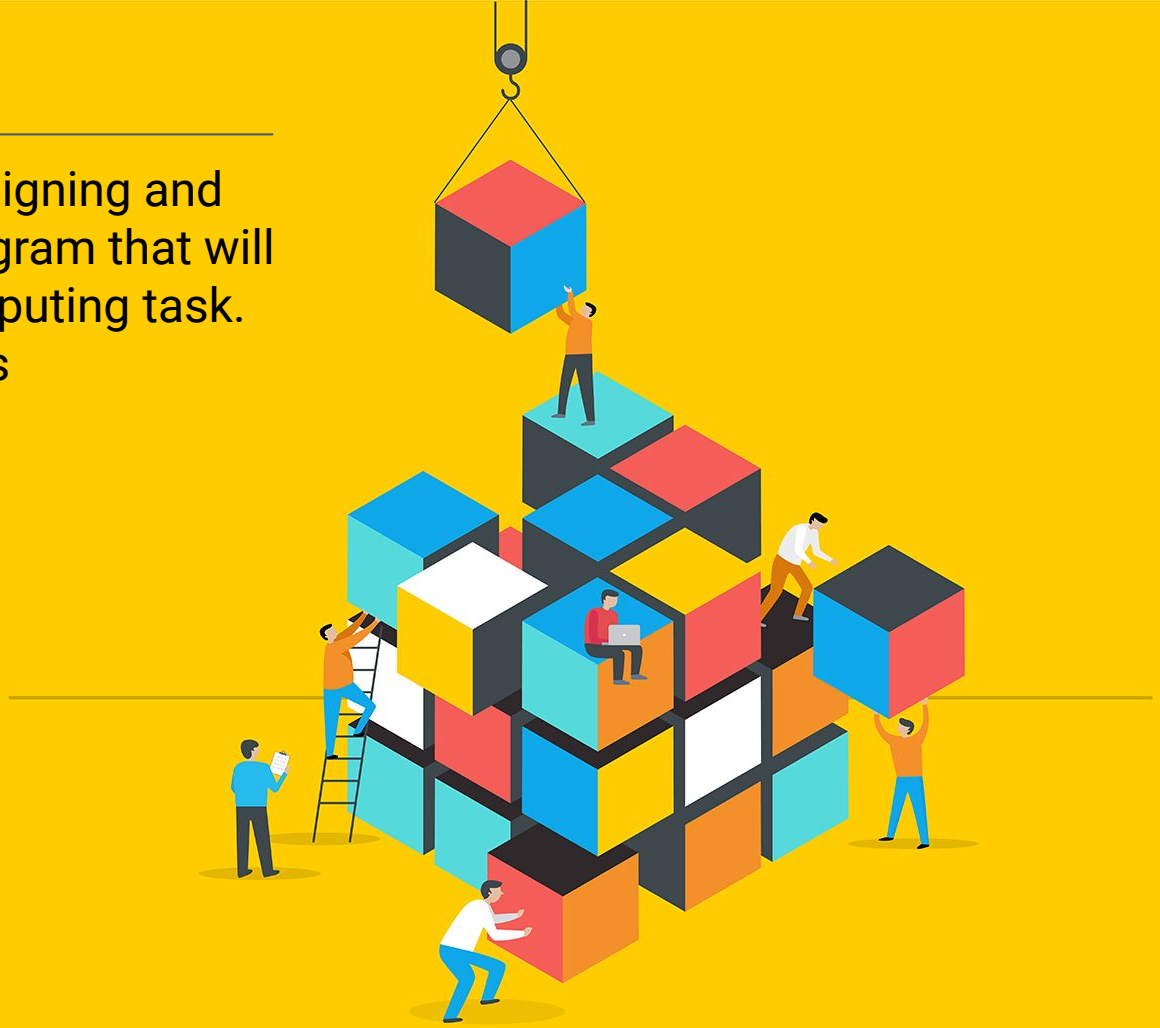
Unit 10



What is programming?

Programming

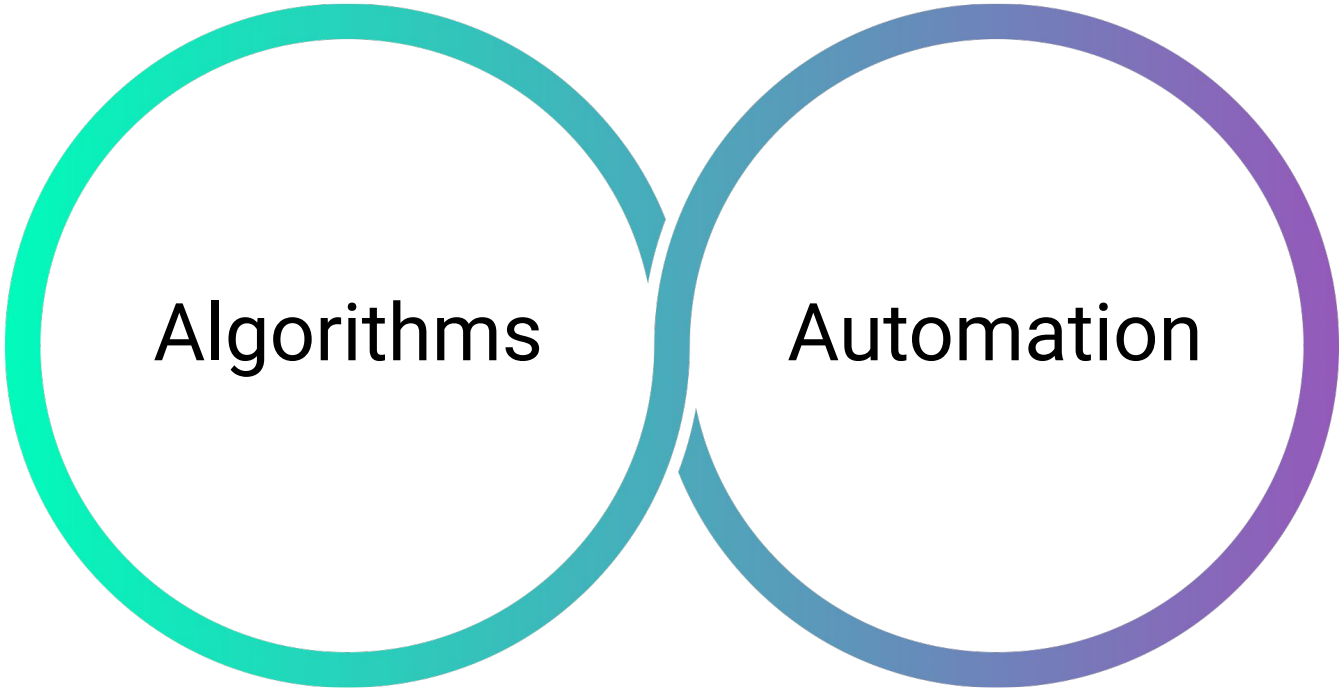
Programming refers to designing and building an executable program that will accomplish a specific computing task. Essentially, programming is problem-solving.



What problems do we solve?

Algorithms and Automation

Programming enables us to solve almost any task or problem on a computer, usually in one of two primary categories: algorithms or automation.



What is DRY?

Don’t Repeat Yourself (DRY)

DRY, or **Don’t Repeat Yourself**, is a fundamental programming principle. Duplicate code wastes time and memory and can confuse readers or contributors to your project.

What is an object?

Objects

Objects in JavaScript are unordered collections of related data built on a key-value structure in which values can be any data type, including functions.

```
const person = {
 name: ['Bob', 'Smith'],
 age: 32,
 gender: 'male',
 interests: ['music', 'skiing'],
 bio() {
   alert(
     `${this.name[0]} ${this.name[1]} is ${this.age} years old.
      He likes ${this.interests[0]} and ${this.interests[1]}.`
   );
 },
 greeting() {
   alert(`Hi! I'm ${this.name[0]}.`);
 },
};
```

Why are objects important in JavaScript?

Because Everything in JavaScript Is an Object!

Well, except for primitive data types. Everything else is an object—essentially a list of key-value pairs.

Data types that are objects:

- Arrays
- Dates
- Math
- Functions
- And more!

Primitive data types (NOT objects):

- Null
- Undefined
- Strings
- Numbers
- Symbols
- Booleans

How do we create objects?

Creating Objects

We can use **object literals**, which define and create an object in one statement.

```
const car = { name: 'honda', model: 'civic', year: 2008, color: 'black' };
```

We can use the **`new`** keyword, which defines and creates a single object from a **`constructor`** or **`class`**. (Kind of like a blueprint for your objects.)

```
class Car {
 constructor(name, model, year, color) {
   this.name = name;
   this.model = model;
   this.year = year;
   this.color = color;
 }
}
const Honda = new Car()
```

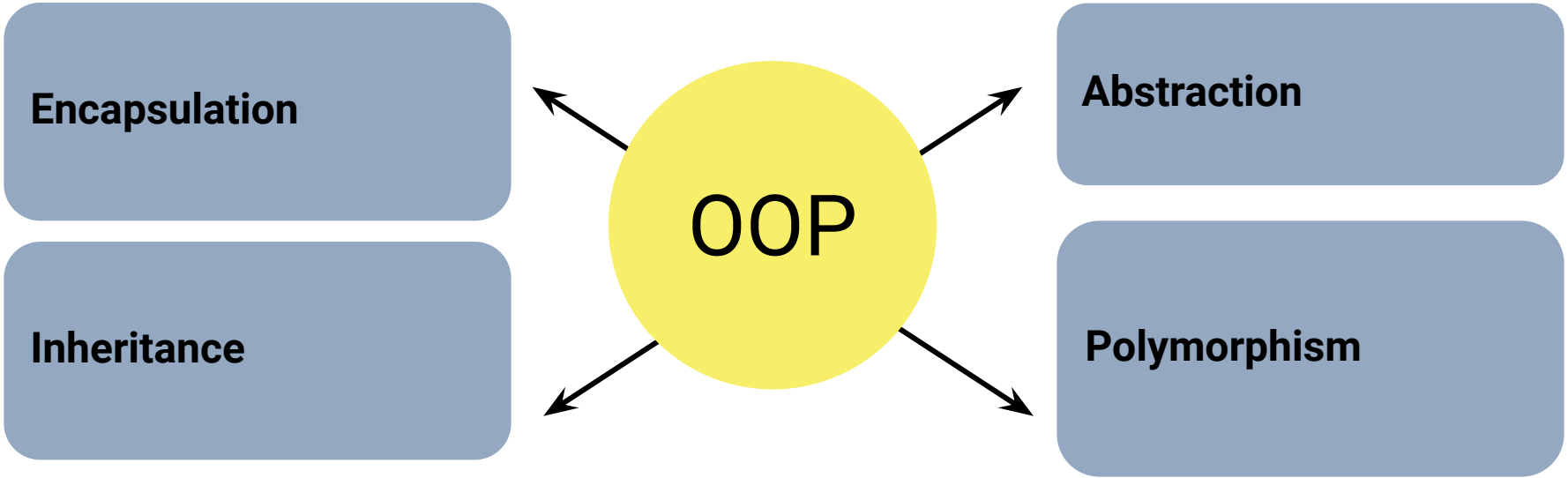
What is object-oriented programming?

Object-Oriented Programming (OOP)

- OOP is a programming paradigm, or pattern, centered around objects.
- In OOP, we solve problems by employing collections of objects that work together.
- Objects can communicate with each other makes them particularly well-suited to address large, complex problems.

Object-Oriented Programming (OOP)

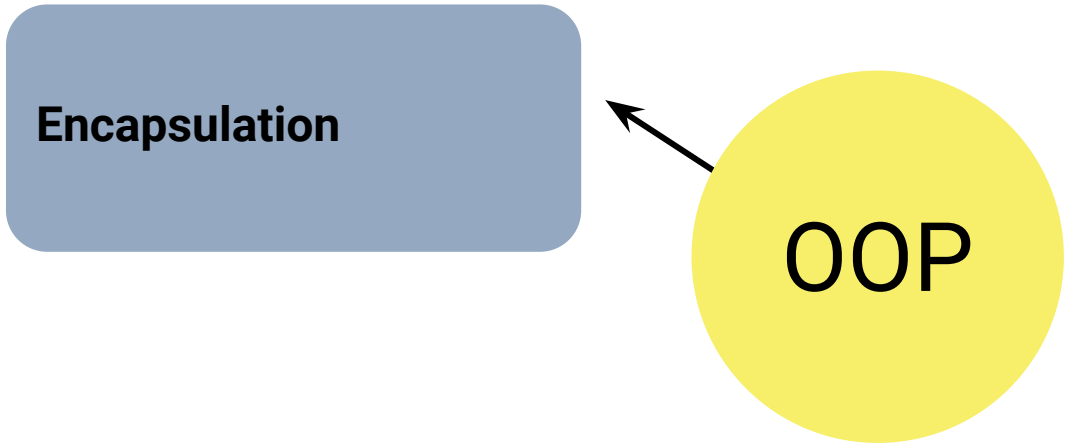
OOP Offers the following benefits:



Object-Oriented Programming (OOP)

Encapsulation:

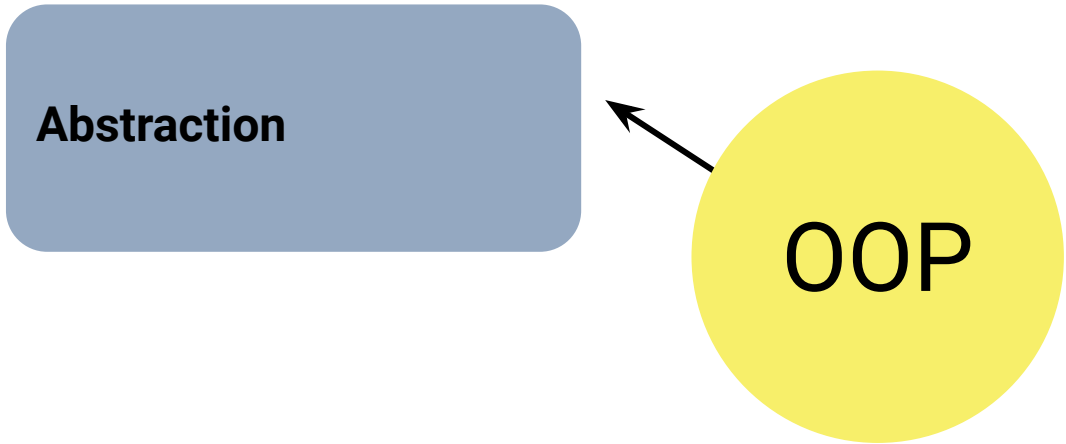
- Object data (values, state) is stored in a neat and organized fashion
- Hides values and state from other objects and provides functions to get/set the values.



Object-Oriented Programming (OOP)

Abstraction:

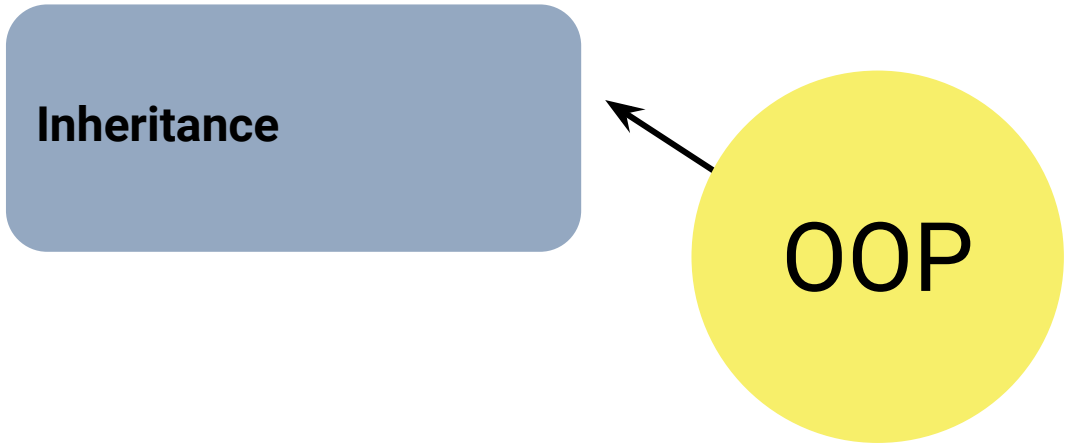
- Provide a simple interface for other objects hiding complex logic.
- Helps decompose the problem by separating part of the application into its own space.
- Objects can be used to model reality. (e.g. if our app has data about cars then we might create a car object with properties like color, make, model, etc.)



Object-Oriented Programming (OOP)

Inheritance:

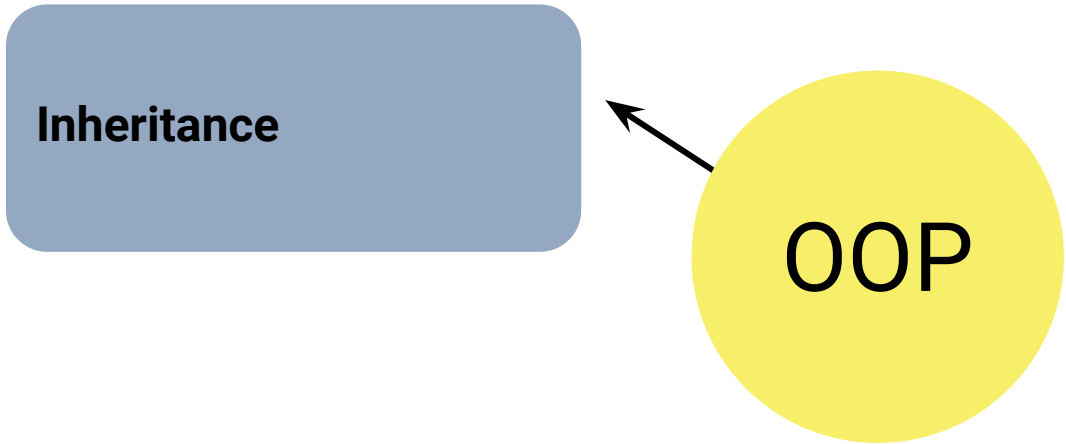
- New types of objects (classes) may be created from other types of objects (classes)
- For example, a Student class may inherit attributes and methods from a Person class.



Object-Oriented Programming (OOP)

Inheritance:

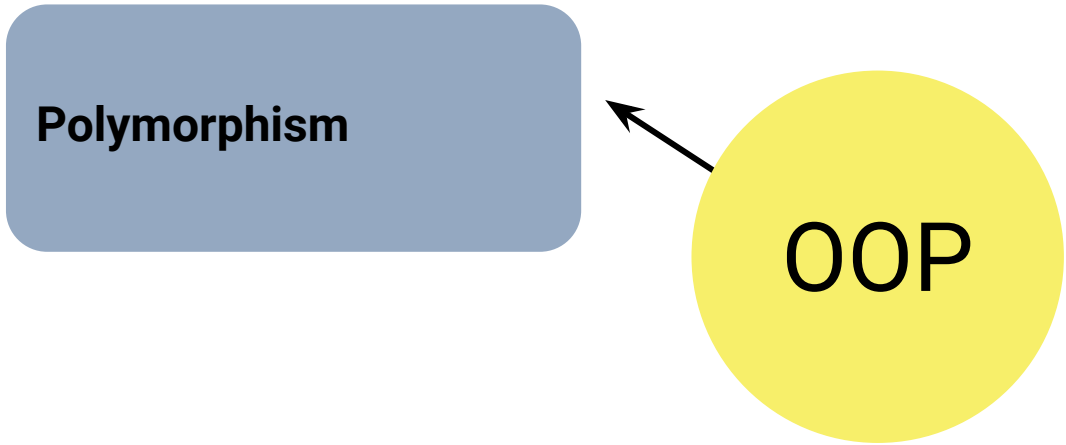
- New types of objects (classes) may be created from other types of objects (classes)
- For example, a Student class may inherit attributes and methods from a Person class.



Object-Oriented Programming (OOP)

Polymorphism:

- Multiple types of objects are able to implement the same methods.
- Methods and functions can behave differently based upon the type and number of arguments received.



How can we learn to use OOP?

How to Learn OOP

OOP is a broad concept that is best learned through real-life examples. We begin to see the value of OOP when we use objects to model real-world things in code and provide functionality that would otherwise be hard or impossible to achieve.

Try some of the following techniques to learn OOP:

- Read the docs and practice with the provided examples.
- Reverse-engineer finished code to see how it was created.
- Build something from scratch.
- Debug a broken app using Chrome DevTools.
- And most importantly, ask questions!