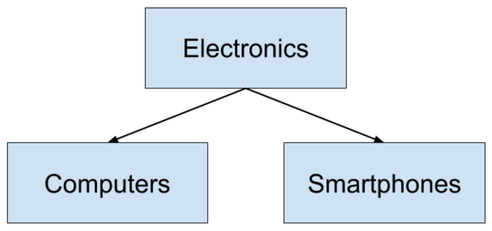
https://youtu.be/qu4dDc-xARM

Inheritence

In our everyday life, we tend to divide things into groups, based on their shared characteristics. Here are some groups that you have probably used yourself: electronics, tools, vehicles, or plants.

Sometimes these groups have hierarchies. For example, computers and smartphones are both types of electronics, but computers and smartphones are also groups in and of themselves. You can imagine a tree with "electronics" at the top, and "computers" and "smartphones" each as children of the "electronics" node.



Object-oriented programming uses the same principles! For instance, imagine a `Vehicle` class:

```
class Vehicle {
public:
  int wheels = 0;
  string color = "blue";

  void Print() const
  {
    std::cout << "This " << color << " vehicle has " << wheels << " wheels!\n";
  }
};
```

We can derive other classes from `Vehicle`, such as `Car` or `Bicycle`. One advantage is that this saves us from having to re-define all of the common member variables - in this case, `wheels` and `color` - in each derived class.

Another benefit is that derived classes, for example `Car` and `Bicycle`, can have distinct member variables, such as `sunroof` or `kickstand`. Different derived classes will have different member variables:

```
class Car : public Vehicle {
public:
  bool sunroof = false;
};

class Bicycle : public Vehicle {
public:
  bool kickstand = true;
};
```

Instructions

1. Add a new member variable to `class Vehicle`.
2. Output that new member in `main()`.
3. Derive a new class from `Vehicle`, alongside `Car` and `Bicycle`.
4. Instantiate an object of that new class.
5. Print the object.