**Classes and Objects**

Objects and classes are used to wrap related [functions](https://www.programiz.com/cpp-programming/function) and data in one place in C++.

Suppose we need to store the length, breadth, and height of a rectangular room and calculate its area and volume.

To handle this task, we can create three [variables](https://www.programiz.com/cpp-programming/variables-literals#variables), say, length, breadth, and height, along with the functions calculate\_area() and calculate\_volume().

However, in C++, rather than creating separate variables and functions, we can also wrap the related data and functions in a single place (by creating **objects**).

This programming paradigm is known as [object-oriented programming](https://www.programiz.com/cpp-programming/oop).

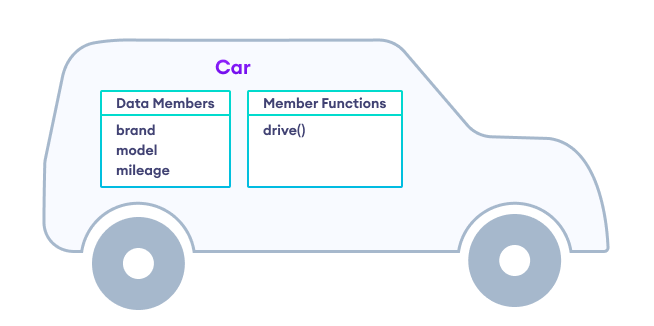
**Object Oriented Programming**

Being an object-oriented programming language, C++ uses objects to model real-world problems

Unlike procedural programming, where functions are written to perform operations on data, OOP involves creating objects that contain both data and functions.

An object has two characteristics: attributes and behavior. For example, a car can be an object. And, it has

* **attributes - brand, model, size, mileage, etc.**
* **behavior - driving, acceleration, parking, etc.**



**A class named Car**

**Class**

A class is a blueprint for the object.

We can think of a class as the technical design (prototype) of a car. It contains all the details about the brand, model, mileage, etc. We can then build different cars based on these descriptions. Here, each distinct car is an object.

class Car {

public:

// class data

string brand, model;

int mileage = 0;

// class function

void drive(int distance) {

mileage += distance;

}

};

In the above code, we have used the class keyword to create a class named Car. Here,

* brand and model are class attributes used to store data
* drive() is a class function used to perform some operation

The public keyword represents an access modifier.

**C++ Objects**

An object is an instance of a class.

For example, the Car class defines the model, brand, and mileage. Now, based on the definition, we can create objects like

Car suv;

Car sedan;

Car van;

Here, suv, sedan, and van are objects of the Car class. Hence, the basic syntax for creating objects is:

Class\_Name object\_name;

**Example 1: Class and Objects in C++**

#include <iostream>

using namespace std;

class Car {

public:

// class data

string brand, model;

int mileage = 0;

// class function to drive the car

void drive(int distance) {

mileage += distance;

}

// class function to print variables

void show\_data() {

cout << "Brand: " << brand << endl;

cout << "Model: " << model << endl;

cout << "Distance driven: " << mileage << " miles" << endl;

}

};

int main() {

// create an object of Car class

Car my\_car;

// initialize variables of my\_car

my\_car.brand = "Honda";

my\_car.model = "Accord";

my\_car.drive(50);

// display object variables

my\_car.show\_data();

return 0;

}

**Output**

Brand: Honda

Model: Accord

Distance driven: 50 miles

In this program, we have created a class Car with data members and a member function. Also, we have created an object my\_car of the Car class.

Notice that we have used the dot operator . with the my\_car object in order to access the class members.

my\_car.brand = "Honda";

my\_car.model = "Accord";

my\_car.drive(50);

my\_car.show\_data();