**Introduction to Object-Oriented Programming (OOP) in JavaScript:**

1. **What is Object-Oriented Programming (OOP)?**
   * Object-Oriented Programming is a programming paradigm based on the concept of "objects", which can contain data in the form of properties (also known as attributes or fields) and code in the form of methods (functions).
   * OOP allows for organising code in a way that models real-world entities, making it easier to manage, reuse, and scale.
2. **Core Concepts of OOP in JavaScript:**
   * **Objects**:
     + An object is a collection of related data and functionality. Objects are created using curly braces {} and can contain properties and methods.

**Example**:  
let car = {

brand: "Toyota",

model: "Corolla",

start: function() {

console.log("Car started");

}

};

* + **Classes**:
    - A class is like a blueprint for creating objects (instances). In JavaScript, classes were introduced in ES6 as syntactical sugar over JavaScript's existing prototype-based inheritance.

**Example**:  
  
class Car {

constructor(brand, model) {

this.brand = brand;

this.model = model;

}

start() {

console.log(`${this.brand} ${this.model} started`);

}

}

* + **Instances**:
    - An instance is a specific object created using a class. Each instance can have different property values, but it shares the same structure and methods defined by the class.

**Example**:  
let myCar = new Car("Toyota", "Corolla");

myCar.start(); // "Toyota Corolla started"

1. **Syntax and Usage**:
   * **Creating Objects**:
     + Objects can be created using object literals, constructors, or classes.

**Example** (using object literal):  
let person = {

name: "John",

age: 30,

greet: function() {

console.log("Hello, " + this.name);

}

};

person.greet(); // "Hello, John"