## Report:

### 1- Abstract VS interface .
### 2- Inheritance in function constructor .

---

## 1- Abstract VS interface :

Abstract is a concept in OOP that allows us to hide the implementation details and show only the functionality to the user. In other words, it ignores the irrelevant details and shows only the required one. Abstraction can be achieved with either abstract classes or interfaces (which you will learn more about in the next section).
In JavaScript, we can achieve abstraction using classes and methods. An abstract class can be used to define a common interface for its subclasses. The abstract class can have abstract and concrete methods. Subclasses that extend the abstract class are required to implement the abstract methods, providing the specific implementation.

Interface is a concept in OOP that defines the set of methods that a certain class should implement. It is a way to achieve runtime polymorphism. An interface can also be defined as a contract that specifies the rules for data communication between different components. The concept of interface is not present in JavaScript. However, TypeScript supports the concept of interfaces.
Because JavaScript doesn't have a built-in interface type, we can achieve it in two ways:

1- Using Abstract Class - as it is explained above.
2- Using Object Literal - it is a simple object that has a set of methods that must be implemented in another class.

## 2- Inheritance in function constructor :

In JavaScript, function constructors are used to create objects with shared properties and methods. Inheritance is achieved by linking objects through their prototypes. Each object created from a constructor has a prototype chain that allows it to inherit properties and methods from its constructor's prototype.