Experiment No.: 03

Title: Demonstrate Interfaces, Classes and Objects in Typescript.

Objectives:

1. To study Interfaces, Classes and Objects in Typescript.

Theory:

Interface-

Interface is a structure that defines the contract in application. It defines the syntax for classes to follow. Classes that are derived from an interface must follow the structure provided by their interface. The TypeScript compiler does not convert interface to JavaScript. It uses interface for type checking. This is also known as "duck typing" or "structural subtyping".

An interface is defined with the keyword **interface** and it can include properties and method declarations using a function or an arrow function.

interface interface_name { }

Interface in TypeScript can be used to define a type and also to implement it in the class. TypeScript interface is also used to define a type of a function. This ensures the function signature. An interface can also define the type of an array where you can define the type of index as well as values.

Interface & Inheritance-

An interface can be extended by other interfaces. In other words, an interface can inherit from other interface. Typescript allows an interface to inherit from multiple interfaces. Use the **extends** keyword to implement inheritance among interfaces.

Single Interface Inheritance Syntax

Child_interface_name extends super_interface_name

Multiple Interface Inheritance Syntax

Child_interface_name extends super_interface1_name,
super_interface2_name, … , super_interfaceN_name

Class-

A class is a blueprint for creating objects. A class encapsulates data for the object. Typescript gives built in support for class. Use the class keyword to declare a class in TypeScript. The syntax is given below –

```
class class_name {
        //class scope        }
```

A class can include the following:

- Constructor
- Properties
- Methods

Abstract Classes-

Abstract classes are base classes from which other classes may be derived. They may not be instantiated directly. Unlike an interface, an abstract class may contain implementation details for its members. The abstract keyword is used to define abstract classes as well as abstract methods within an abstract class.

Object-

An object is an instance which contains set of key value pairs. The values can be scalar values or functions or even array of other objects. An object of the class can be created using the **new** keyword.

Key Concept: Interfaces, Classes, Objects