**Experiment – 08**

**Aim:** Write a typescript program to work with classes.

## CLASSES IN TYPESCRIPT:

In object-oriented programming languages like Java, classes are the fundamental entities which are used to create **reusable** components. It is a group of objects which have common properties. In terms of OOPs, a class is a **template** or **blueprint** for creating objects. It is a logical entity.

##### **A class definition can contain the following properties:**

**Fields:** It is a variable declared in a class.

**Methods:** It represents an action for the object.

**Constructors:** It is responsible for initializing the object in memory.

**Nested class and interface:** It means a class can contain another class.

TypeScript is an Object-Oriented JavaScript language, so it supports object-oriented programming features like classes, interfaces, polymorphism, data-binding, etc. JavaScript **ES5** or **earlier version** did not support classes. TypeScript support this feature from **ES6** and **later version**. TypeScript has **built-in** support for using classes because it is based on ES6 version of JavaSript. Today, many developers use class-based object-oriented programming languages and compile them into JavaScript, which works across all major browsers and platforms.

**Syntax to declare a class:**

class <class\_name>

{

field;

method;

}

The TypeScript compiler converts class into JavaScript code.

## Creating an object of class:

A class creates an object by using the **new** keyword followed by the **class name**. The new keyword allocates memory for object creation at runtime. All objects get memory in heap memory area. We can create an object as below.

## Syntax:

let object\_name = new class\_name(parameter)

1.new keyword: it is used for instantiating the object in memory.

2.The right side of the expression invokes the constructor, which can pass values.

**Object Initialization:**

Object initialization means storing of data into the object. There are three ways to initialize an object. These are:

1. By reference variable
2. By method
3. By constructor

**student.ts**

class Student

{

studcode:number

studname:string

grade:string

constructor(code:number,name:string,grade:string)

{

this.studcode=code

this.studname=name

this.grade=grade

}

display():void

{

console.log("Name: ",this.studname)

console.log("Code: ",this.studcode)

console.log("Grade: ",this.grade)

}

}

let obj1=new Student(532, 'Manohar', 'A')

let obj2=new Student(509, 'Nagendra', 'A+')

let obj3=new Student(520, 'Jaswanth', 'A')

obj1.display()

obj2.display()

obj3.display()

**student.js**

var Student = /\*\* @class \*/ (function () {

function Student(code, name, grade) {

this.studcode = code;

this.studname = name;

this.grade = grade;

}

Student.prototype.display = function () {

console.log("Name: ", this.studname);

console.log("Code: ", this.studcode);

console.log("Grade: ", this.grade);

};

return Student;

}());

var obj1 = new Student(532, 'Manohar', 'A');

var obj2 = new Student(509, 'Nagendra', 'A+');

var obj3 = new Student(520, 'Jaswanth', 'A');

obj1.display();

obj2.display();

obj3.display();

**output:**

PS D:\MEAN\_532\typescript> tsc student.ts

PS D:\MEAN\_532\typescript> node student.js

Name: Manohar

Code: 532

Grade: A

Name: Nagendra

Code: 509

Grade: A+

Name: Jaswanth

Code: 520

Grade: A