# Experiment No: 5

**Aim:** To study access modifiers using typescript.

The TypeScript access modifiers are of three types. These are:

- Public
- Private
- Protected

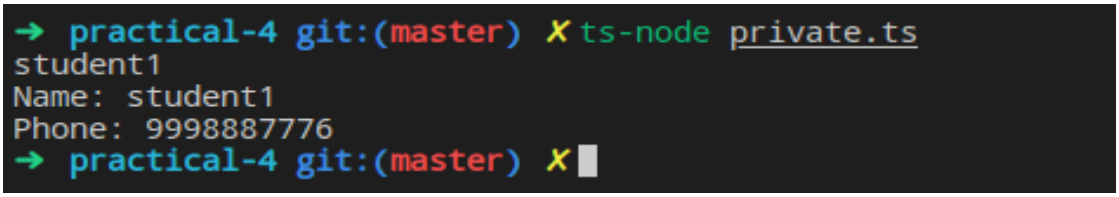
**Public:**

In TypeScript by default, all the members (properties and methods) of a class are public. So, there is no need to prefix members with this keyword.

## Code:

```
class Student{
 public name: string;
 public phone: number;
 constructor(name: string, phone: number){
 this.name = name;
 this.phone = phone;
 }
}

const std_obj = new Student("student1",9998887776);
std_obj.name = "1student";
console.log(`Name: ${std_obj.name}`);
console.log(`Phone: ${std_obj.phone}`);
```

**Output:**



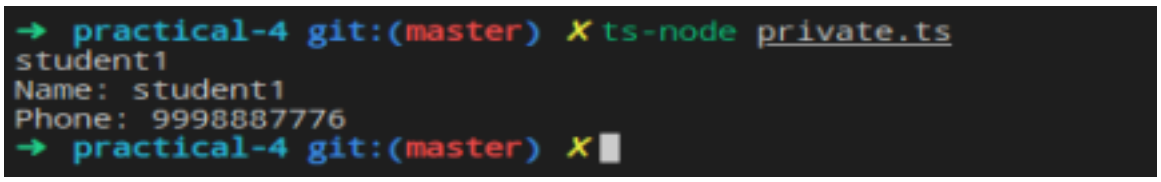
**Private:**

The private access modifier cannot be accessible outside of its containing class. It ensures that the class members are visible only to that class in which it is contained.

**Code:**

```
class StudentPrivate{
 private name: string;
 private phone: number;

 constructor(name: string, phone: number){
 this.name = name;
 this.phone = phone;
 }

 getName():string{
 return this.name;
 };

 setName(name:string):void{
 this.name = name;
 }

 public display(): void{
 console.log(`Name: ${this.name}\nPhone: ${this.phone}`);  }
}
const std_private = new StudentPrivate("student1", 9998887776);
console.log(std_private.getName());
std_private.display();
```

**Output:**




**Protected**

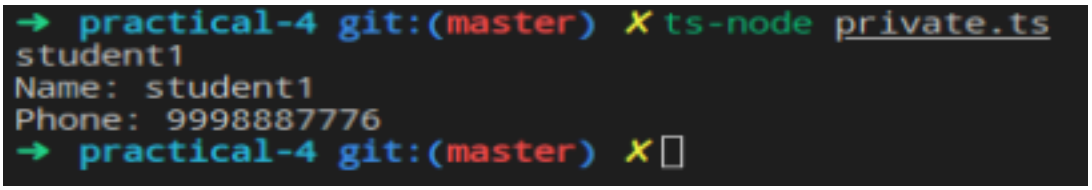
A Protected access modifier can be accessed only within the class and its subclass. We cannot access it from the outside of a class in which it is contained.

**Code:**

```
class Person{
 protected aadharNo:number;
 public name: string;
 constructor(aadharNo: number, name: string){
```

```
 this.aadharNo = aadharNo;
 this.name = name;
 }
 display():void{
 console.log("Inside Person")
 }
}

class Employee extends Person{
 private salary;
 constructor(aadharNo:number ,name: string){
 super(aadharNo,name);
 this.salary = 1000;
 super.display();
 }
 display(): void {
 console.log("Inside Employee");
 }
 print():void{
 console.log("Aadhar No:",this.aadharNo);
 }
}

const employee = new Employee(444455556666,"student1");
employee.print();
```

**Output:**




**Conclusion**: Access modifiers such as public, private, and protected ensure data encapsulation and security in TypeScript. These modifiers control access to class properties and methods, preventing unintended modifications. Understanding access control mechanisms helps in writing robust and maintainable