```
//============ Constructor Function ==========

  // create a function .. and function can be here like class

    // one key point is: when object is create it (this) refer object inside the function
    function User(){
          console.log(this); // User{} same  => me
    }
    // create object of this function
    let me = new User(); // create new object;
    console.log(me); // User{};

//**** when object is created we can set value by (this) inside the function it will effect both of them  */

    function Set(){
        console.log(this); // empty object => Set{};

        this.name = "Mehedi"; // set value into object
        this.Age = 20; // set value into Object

        console.log(this); // Set => Set {name: "Mehedi", Age: 20}
    }

    // Create New Object of
    let set = new Set();
    console.log(set); // Set => Set {name: "Mehedi", Age: 20};


//***** We can pass vale creating time in Constactor *******/

    function Default(Nam){
        this.name = Nam;
    }
    // create new object
    let passName = new Default("Mehedi");
    console.log(passName); // Default {name: "Mehedi"}
    // same function but create many objects
    let you = new Default("Hasan");
    console.log(you); // Default {name: "Hasan"}

    // two objects are created with deferent Name
   // Default {name: "Mehedi"};
   // Default {name: "Hasan"};


//****** We can pass many parameter with different type *******/

    function Many(Name, Interst, Year){
        this.name = Name;
        this.intra = Interst;
        this.year = Year;
    }
    // Create few Object and Output
    let mehedi = new Many("Mehedi", ["book", "Talking", "Sleeping"], 2021);
```

```
    let mehedi = new Many("Mehedi", ["book", "Talking", "Sleeping"], 2021);
    let Hasan = new Many("Hasan", ["Travel"], 2024);
    console.log(mehedi); // Many {name: "Mehedi", intra: Array(3), year: 2021}
    console.log(Hasan); // Many {name: "Hasan", intra: Array(1), year: 2024}

    // we can add property after creating object like
    mehedi.age = 20;
    console.log(mehedi); // Many {name: "Mehedi", intra: Array(3), year: 2021, age: 20}



// ============= Factory function ===========

    function User1(Name, Age){
        this.name = Name;
        this.age = Age;
    };

    function newUser1(Name, Age){
        let person = {
            name: Name,
            age: Age,
        }

        return person;
    }

    let Me1 = new User1("Mehedi", 30);
    let You1 = newUser1("Hasan", 20);
    console.log(Me1); // User1 {name: "Mehedi", age: 30}
    console.log(You1); // {name: "Hasan", age: 20}

// ============ Prototype of Constructor ==========

    function Parent(Name, Age){
        this.name = Name;
        this.age = Age;
        /*
            // we can create fuction it will copy for all object
            this.Ouput = fuction(){
                console.log("My name" + this.name + " Age"+ this.age);
            }
        */
    }
    // but copy all the object is wasted the memory we can use prototype to ascess the date object
    Parent.prototype.print = function(){
      console.log("My name " + this.name + " Age "+ this.age);
    }
    let Me2 = new Parent("Mehedi", 20);
    let You2 = new Parent("Hasan", 30);
    Me2.print(); // My name Mehedi Age 20
```