

Name: Ayush Agrawal

Date: 09-03-2022

## Assignment-1

1. Constants: Declare a constant & confirm its value cannot be changed.

TypeScript Code:

```
Q1 > TS index.ts > ...
1  const pi = 3.14;
2  document.write(`<h3>${pi}</h3>`);
3
4  pi = 22/7;
5  document.write(`<h3>pi value cannot be changed</h3>`);
6
```

Output:

```
Q1 > TS index.ts > ...
1  const pi = 3.14;
2  document.write(`<h3>${pi}</h3>`);
3
4  const pi: any
5  Cannot assign to 'pi' because it is a constant. ts(2588)
6  View Problem Quick Fix... (Ctrl+.)
7
8  pi = 22/7;
9  document.write(`<h3>pi value cannot be changed</h3>`);
10
```

← → ↻ ⓘ 127.0.0.1:5500/Q1/index.html

**Q1) Output:**

**pi = 3.14**

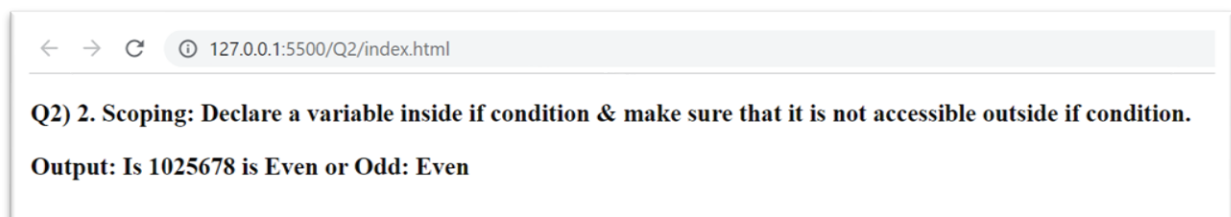
**pi value cannot be changed**

**2. Scoping: Declare a variable inside if condition & make sure that it is not accessible outside if condition.**

**TypeScript Code:**

```
Q2 > index.ts > ...
1
2 function OddEven(num){
3
4     if(num%2==0){
5         return "Even";
6     }
7     else{
8         return "Odd";
9     }
10 }
11 document.write(`<h3>Output: Is 1025678 is Even or Odd: ${OddEven(1025678)}</h3>`)
```

**Output:**



Q2) 2. Scoping: Declare a variable inside if condition & make sure that it is not accessible outside if condition.

Output: Is 1025678 is Even or Odd: Even

**3. Enhanced object properties: Create an 'Order' object having data members 'id', 'title', 'price'. Add the methods printOrder() & getPrice(). Now, copy the order object using Object.assign().**

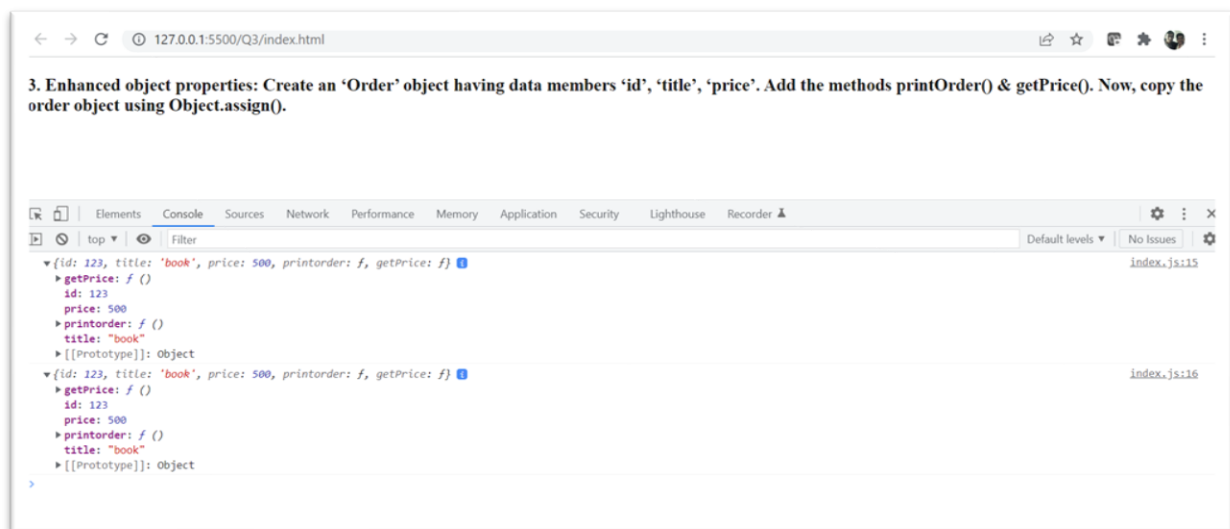
**TypeScript Code:**

```

Q3 > TS index.ts > ...
1  export {}
2  let order = {
3      id: 123,
4      title: "book",
5      price: 500,
6      printorder: function() {
7          return (this.id);
8      },
9      getPrice: function() {
10         return (this.price);
11     }
12 };
13
14 let order2 = (<any>Object).assign({}, order);
15
16 console.log(order);
17 console.log(order2);
18

```

**Output:**



**4. Arrow functions: Take an array of strings & convert it into another array of object which has two properties {string, string\_length}. For example:**

```
let names = ['Tom', 'Ivan', 'Jerry']
```

Output: [ {name: 'Tom', length: 3}, {name: 'Ivan', length: 4 }, {name: 'Jerry', length: 5} ]

TypeScript Code:

```
Q4 > TS index.ts > ...
1
2 export {}
3 var arrow = (names: string[]) => {
4     let out = [];
5     for(let color of names ){
6         var row : any= {
7
8         };
9         row.name = color;
10        row.length = color.length;
11        out.push(row);
12    }
13    return out;
14 };
15
16 let names = ["Hritik","Ruchika","Roshni"];
17 console.log(arrow(names));
```

Output:

← → ↻ ⓘ 127.0.0.1:5500/Q4/index.html 🔗 ☆ 🏠 ⚙️ 👤 ⋮

**4. Arrow functions: Take an array of strings & convert it into another array of object which has two properties {string, string\_length}. For example: let names = ['Tom', 'Ivan', 'Jerry'] Output: [ {name: 'Tom', length: 3}, {name: 'Ivan', length: 4 }, {name: 'Jerry', length: 5} ]**

**Output: Ayush Agrawal ,Ritik Dixit ,Adisri Sarode ,Madhavi Chavhan ,Pratik Nandurkar**

🔍 📄 | Elements Console Sources Network Performance Memory >> | ⚙️ ⋮ ✕

🔍 🛑 | top ▾ | 👁 | Filter | Default levels ▾ | No Issues | ⚙️

▼ Array(5) ⓘ index.js:21

- ▶ 0: {name: 'Ayush Agrawal ', length: 14}
- ▶ 1: {name: 'Ritik Dixit ', length: 12}
- ▶ 2: {name: 'Adisri Sarode ', length: 14}
- ▶ 3: {name: 'Madhavi Chavhan ', length: 16}
- ▶ 4: {name: 'Pratik Nandurkar ', length: 17}

length: 5

▶ [[Prototype]]: Array(0)

>

#### 5. Extended parameter handling:

a. Write a add() with default values.

b. Write a function userFriends() that takes 2 arguments username & array of user friends. The function should print username & his list of friends. (Use rest parameters)

c. Write a function printCapitalNames() that takes five names as argument & prints them in capital letters. Use spread operator in order to call printCapitalNames() function.

**TypeScript Code:**

Q5 > TS index.ts > username

```
1  export{}
2  let addvalue=function(a=5){
3      console.log(a);
4  };
5  addvalue();
6  addvalue(25);
7  var names=["A","B","C","D","E"];
8  let userFriends=function(username: any,...friends: any)
9  {
10     console.log(username)
11     for(let friend of friends){
12         console.log(friend);
13     }
14 }
15 var username="Ayush Agrawal";
16 userFriends(username,names);
17
18 let printcapitalname=function(...names:any){
19     for(let name of names){
20         console.log(name.toUpperCase());
21     }
22 }
23 printcapitalname(...names);
```

## Output:

127.0.0.1:5500/Q5/index.html

### 5. Extended parameter handling:

a. Write a add() with default values.

b. Write a function userFriends() that takes 2 arguments username & array of user friends. The function should print username & his list of friends. (Use rest parameters)

c. Write a function printCapitalNames() that takes five names as argument & prints them in capital letters. Use spread operator in order to call printCapitalNames() function.

The screenshot shows the Chrome DevTools console with the following output:

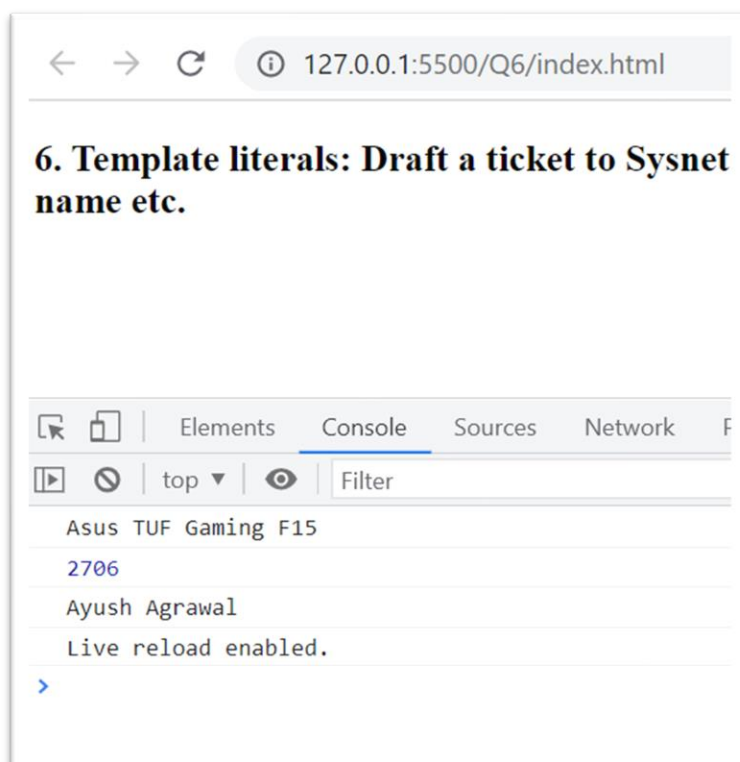
- Line 5: `Ayush Agrawal`
- Line 25: `Ayush Agrawal`
- Line 137: `["A", "B", "C", "D", "E"]`
- Line 137: `0: "A"`
- Line 137: `1: "B"`
- Line 137: `2: "C"`
- Line 137: `3: "D"`
- Line 137: `4: "E"`
- Line 137: `length: 5`
- Line 137: `[[Prototype]]: Array(0)`
- Line 132: `A`
- Line 132: `B`
- Line 132: `C`
- Line 132: `D`
- Line 132: `E`

**6.Template literals: Draft a ticket to Sysnet that describes problem with your laptop. Use 'template literals' to add value of laptop model, your desk no, your name etc.**

**TypeScript Code:**

```
Q6 > TS index.ts > ...
1   export{}
2   let model ="Asus TUF Gaming F15"
3   let desknumber=2706;
4   let names="Ayush Agrawal";
5   let laptop={
6       model,
7       desknumber,
8       names
9   };
10  console.log(laptop.model);
11  console.log(laptop.desknumber);
12  console.log(laptop.names);
```

**Output:**



## 7. De-structuring assignment:

- a. Suppose there is a javascript array with 4 elements. Print the value of 3rd element using array matching.
- b. Create an organization object having attributes name, address. Write a program to retrieve pin code of an address using object deep matching.

### TypeScript Code:

```
Q7 > Js index.js > ...
1  "use strict";
2
3  var info = ["Ayush Agrawal", "Male", 21, 5.7];
4  var name1 = info[0],
5      gender = info[1],
6      age = info[2],
7      height = info[3];
8  console.log(age);
9  var obj = {
10     student: "Ayush Agrawal",
11     address: { pin: 495677 },
12 };
13 var student = obj.student,
14     address = obj.address;
15 console.log(address.pin);
16
```

### Output:





8. Classes & Modules: Write a class Account with attributes id, name, balance. Add two sub classes SavingAccount & CurrentAccount having specific attribute interest & cash\_credit respectively. Create multiple saving & current account objects. Write a functionality to find out total balance in the bank.

TypeScript Code:

```
Q8 > TS index.ts > [E] currentAccount
1  export {}
2  class Account {
3      id: any;
4      name: any;
5      balance: any;
6      constructor(id, name, balance) {
7          this.id = id;
8          this.name = name;
9          this.balance = balance;
10     }
11
12     totalBalance() {
13         return this.balance;
14     }
15 }
16
17 class SavingAccount extends Account {
18     constructor(id, name, balance, interest) {
19         super(id, name, balance = balance + (balance*interest)/100);
20     }
21 }
22
23 class CurrentAccount extends Account {
24     constructor(id, name, balance, cash_credit) {
25         super(id, name, balance = balance+cash_credit);
26     }
27 }
28
29 let savingAccount = new SavingAccount(11717121, "Ayush Agrawal", 10000, 10);
30 let currentAccount = new CurrentAccount(11717122, "Ritik Dixit", 14000, 20);
31
32 console.log(savingAccount.totalBalance());
33 console.log(currentAccount.totalBalance());
```

Output:

