Day 3/Question_1

1) Various methods in Console function in JavaScript

<u>Console object</u>

In JavaScript, the console is an object which provides access to the browser debugging console. We can open a console in web browser by using: *Ctrl* + *Shift* + *K* for windows and *Command* + *Option* + *K* for Mac. The console object provides us with several different methods, like :

- console.log()
- console.error()
- console.warn()
- console.clear()
- console.time() and console.timeEnd()
- console.table()
- console.count()
- console.group() and console.groupEnd()
- custom console logs

Let's look at all these methods one by one.

1.console.log()

Mainly used to log(print) the output to the console. We can put any type inside the log(), be it a string, array, object, boolean etc.

e.g :

```
console.log('abc');
console.log(1);
console.log(true);
console.log(null);
console.log(undefined);
console.log([1, 2, 3, 4]); // array inside log
console.log({a:1, b:2, c:3}); // object inside log
```

2. console.error()

Used to log error message to the console. Useful in testing of code. By default the error message will be highlighted with red color.

e.g :

```
console.error('This is a simple error');
```

3. console.warn()

Used to log warning message to the console. By default the warning message will be highlighted with yellow color.

e.g:

```
console.warn('This is a warning.');
```

4. console.clear()

Used to clear the console. The console will be cleared, in case of Chrome a simple overlayed text will be printed like : ‘Console was cleared’ while in firefox no message is returned.

e.g. :

```
console.clear();
```

5. console.time() and console.timeEnd()

Whenever we want to know the amount of time spend by a block or a function, we can make use of the time() and timeEnd() methods provided by the javascript console object. They take a label which must be same, and the code inside can be anything(function, object, simple console).

e.g. :

```
console.time('sample');
 let fun1 =  function(){
    console.log('fun1 is running');
 }
 let fun2 = function(){
    console.log('fun2 is running..');
 }
 fun1(); // calling fun1();
 fun2(); // calling fun2();
console.timeEnd('sample');
```

In the above code sample, we can see that the label is ‘'sample' which is same for both the time() and the timeEnd() method. If we increase the amount of code inside the block defined by these methods, then the time will increase. It is also worth remembering that the time returned to the console will be in milliseconds and might be different each time we refresh the page.

6. console.table()

This method allows us to generate a table inside a console. The input must be an array or an object which will be shown as a table.

e.g.:

```
console.table({'a':1, 'b':2});
```

7. console.count()

This method is used to count the number that the function hit by this counting method.

e.g.

```
for(let i=0;i<5;i++){
   console.count(i);
}
```

8. console.group() and console.groupEnd()

group() and groupEnd() methods of the console object allows us to group contents in a separate block, which will be indented. Just like the time() and the timeEnd() they also accepts label, again of same value.

e.g. :

```
console.group('simple');
  console.warn('warning!');
  console.error('error here');
  console.log('vivi vini vici');
console.groupEnd('simple');
console.log('new section');
```

9. Custom Console Logs

User can add Styling to the console logs in order to make logs Custom . The Syntax for it is to add the css styling as a parameter to the logs which will replace %c in the logs as shown in the example below .

e.g. :

```
const spacing = '10px';
const styles =
        `padding: ${spacing}; background-color: white; color: green; font-style:
         italic; border: 1px solid black; font-size: 2em;`;
  console.log('%cHello Everyone !', styles);
```