The modern mode, “use strict”, Variables

Task 11:

11. Write a script without using “use strict” and try to assign a value to an undeclared variable. Note the result.

Code:

```
C: > Users > Student.MAT-54.000 > task 11.html >
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            x=10;
            console.log(x);
        </script>
    </body>
</html>
```

Output:

```
top ▾ | Filter | Default levels ▾ | 1 Issue: 1
10                                          task_11.html:10
>
```

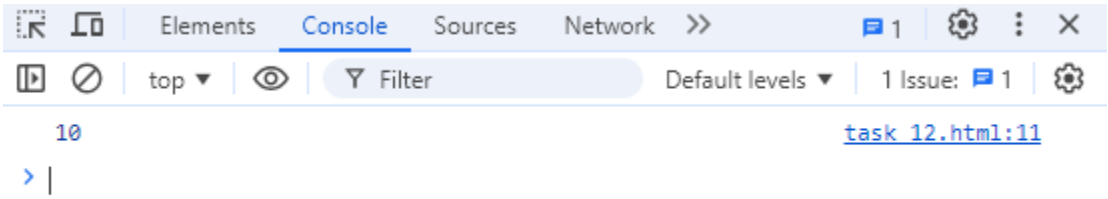
Task 12:

12 .Enable “use strict” mode and repeat the above action, noting the difference

Code:

```
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            `use script`;
            x=10;
            console.log(x);
        </script>
    </body>
</html>
```

Output:



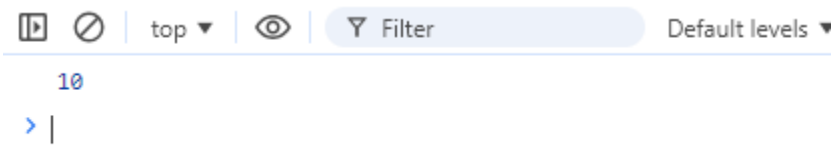
Task 13:

In “use strict” mode, try to delete a variable, function, or function parameter

Code:

```
C: > Users > Student.MAT-54.000 > task 12.html > html
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            `use script`;
            let x=10;
            delete x;
            console.log(x);
        </script>
    </body>
</html>
```

Output:



Task 14:

Assign a value to an undeclared variable without “use strict” and then with “use strict”.

Code:

```
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>

            a= 256;
            console.log(a);
            `use script`;
            x=10;
            console.log(x);
        </script>
    </body>
</html
```

Output:

```
top ▾   Filter   Default levels ▾   1 Issue
256                                          task 14.
10                                           task 14.
> |
```

Task 15:

Declare a variable with a reserved keyword in “use strict” mode.

Code:

```
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>

            `use script`;
         let let =10;
            console.log(let);
        </script>
    </body>
</html
```

Output:

```
Uncaught SyntaxError: let is disallowed as a lexically bound name    task%2014.html:11
>
```

Task 16:

Declare variables using let, const, and var. Discuss when each should be used.

Code:

```html
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            var x=106;
            var x=220;
            console.log(x);
            let name = "john";
            name="alex";
            console.log(name);
            const b=2345;
            console.log(b);

        </script>
    </body>
</html>
```

Output:

```
220                    task:
alex                   task:
2345                   task:
>
```

`let`:

Use when you need a variable to change over time and you want block-level scoping (e.g., in loops or conditionals).

Example: Loop counters, temporary values in conditions, etc.

`const`:

Use when the value should not change after initialization. This is the default choice for most variables that won't be reassigned.

Example: Constants, references to objects/arrays, etc.

`var`:

Use only in legacy code or when working with functions in very specific scenarios. Avoid using `var` in modern JavaScript to prevent scoping issues and bugs caused by hoisting.

Task 17:

Attempt to reassign a const variable and observe the result.

Code:

```
C: > Users > Student.MAT-54.000 > taask17
1   <html>
2       <head>
3           <title>
4               task
5           </title>
6       </head>
7       <body>
8           <script>
9               const b=2345;
10              b=5445;
11              console.log(b);
12          </script>
13      </body>
14  </html>
```

Output:

```
top ▼   Filter   Default levels ▼   1 Issue
▶ Uncaught TypeError: Assignment to constant variable.   taask17;.
    at taask17;.html:11:14
>
```

Task 18:

Declare a variable without initializing it and print its value.

Code:

```
C: > Users > Student.MAT-54.000 > task 18.htm
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            let name;
            console.log(name);
        </script>
    </body>
</html>
```

Output:

```
top ▾   Filter   Default levels ▾   1 Issu
undefined                                  task 18.
>
```

Task 19:

Assign a number, string, and boolean value to a variable and print its type using typeof.

Code:

```
: > Users > Student.MAT-54.000 > task19.html > htr
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            let name="john";
            let num=728;
            let san=true;
           console.log(typeof(name));
           console.log(typeof(num));
           console.log(typeof(san));
        </script>
    </body>
</html>
```

Output:

```
string                                  task19.html:12
number                                  task19.html:13
boolean                                 task19.html:14
>
```

Task 20:

Rename a variable and observe the outcome.

Code:

```
C: > Users > Student.MAT-54.000 > task20.html >
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            let name="john";
            let newname;
            newname=name;
            console.log(newname);
        </script>
    </body>
</html>
```

Output:

```
top ▾   Filter   Default levels ▾   1 Issue: 1
john                                    task20.html:12
>
```

Data types, Basic operators, maths

Task 21:

Create variables of different data types (e.g., string, number, boolean, null, undefined, object).

Code:

```
> Users > Student.MAT-54.000 > task 21.htm
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            let name="madhu";
            let num=821;
            let san=false;
            let litter=null;
            let fre=76.977;
           console.log(name);
           console.log(num);
           console.log(san);
           console.log(litter);
           console.log(fre);
        </script>
    </body>
</html>
```

Output:

```
top ▾   Filter   Default levels ▾   1 Issue: 1
madhu                                   task_21.html:14
821                                     task_21.html:15
false                                   task_21.html:16
null                                    task_21.html:17
76.977                                  task_21.html:18
>
```

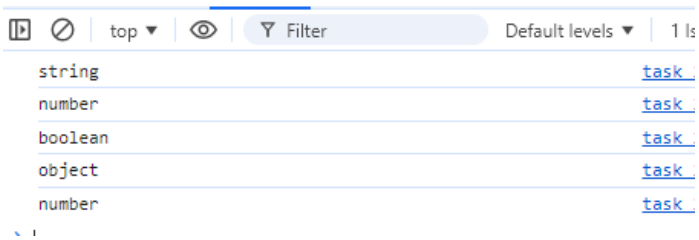
Task 22:

Use the typeof operator to determine the type of various variables.

Code:

```html
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            let name="madhu";
            let num=821;
            let san=false;
            let litter=null;
            let fre=76.977;
           console.log(typeof(name));
           console.log(typeof(num));
           console.log(typeof(san));
           console.log(typeof(litter));
           console.log(typeof(fre));
        </script>
    </body>
</html>
```

Output:



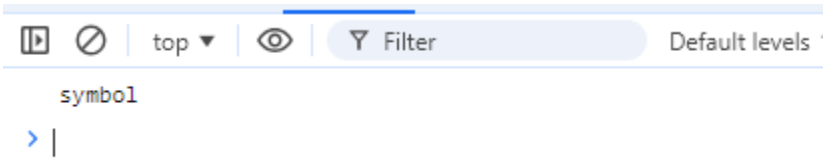
Task 23:

Declare a symbol and print its type.

Code:

```html
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            let name = Symbol('120');
           console.log(typeof(name));
        </script>
    </body>
</html>
```

Output:



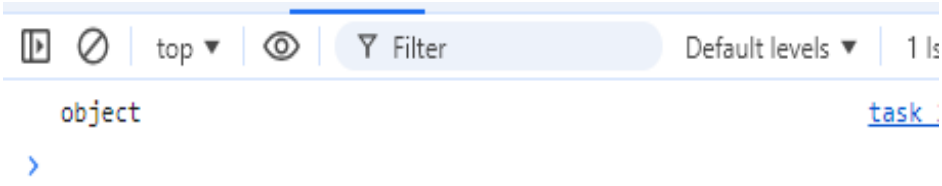
Task 24:

Assign the value null to a variable and check its type using typeof.

Code:

```
<html>
    <head>
        <title>
            task
        </title>
    </head>
    <body>
        <script>
            let name = null;
           console.log(typeof(name));
        </script>
    </body>
</html>
```

Output:



Task 25:

Differentiate between declaring a variable using var and let in terms of scope.

Code:

C: > Users > Student.MAT-54.000 > 24 task.html > ...

```html
<html>
    <head>
        <title>Task</title>
    </head>
    <body>
        <script>
            function test(){
                if(true){
                    var name = "john";
                    let newname = "alex";
                    console.log(name);
                }
                console.log(newname);
            }
            test();
        </script>
    </body>
</html>
```

Output:

top ▾ | Filter | Default levels ▾ | 1 Issue:

```
john                                                    24 task.ht
▶Uncaught ReferenceError: newname is not defined        24 task.ht
    at test (24 task.html:13:29)
    at 24 task.html:15:13
>
```