IT1100 - Internet and Web Technologies

Lecture 04
JavaScript – Part 1

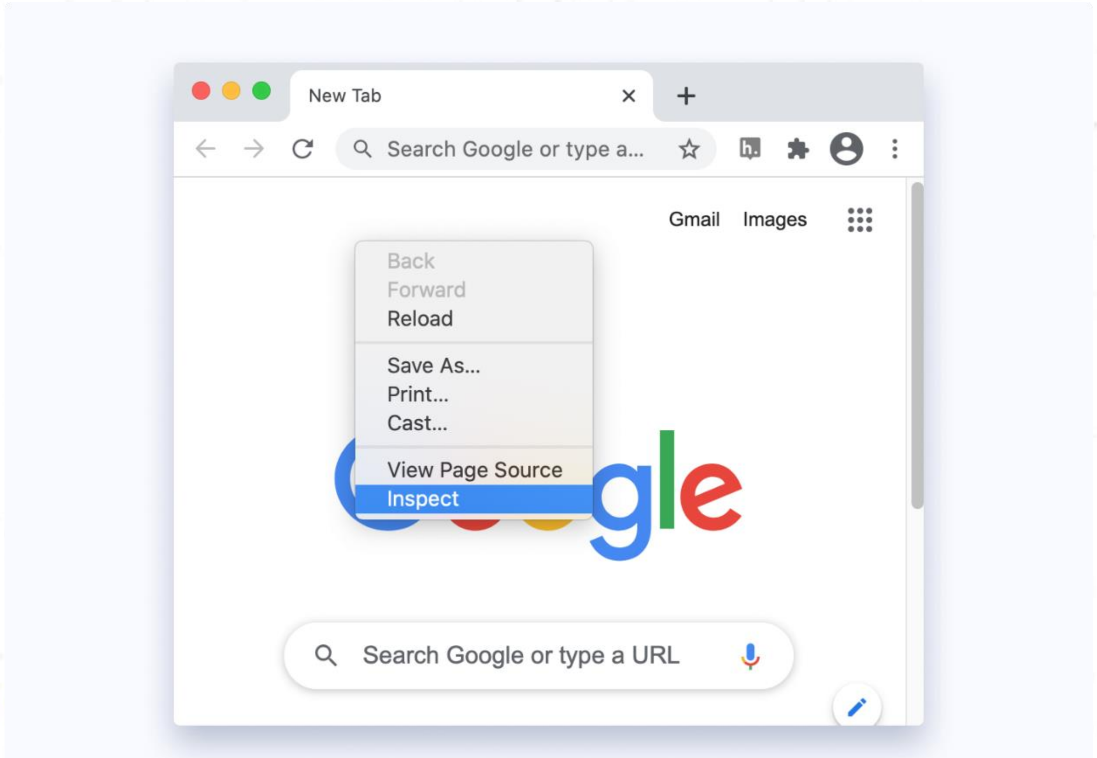
SLIIT
FACULTY OF COMPUTING

Content

- Introduction to the JavaScript
- Variables in JS
- Operators in JS
- Control structures in JS

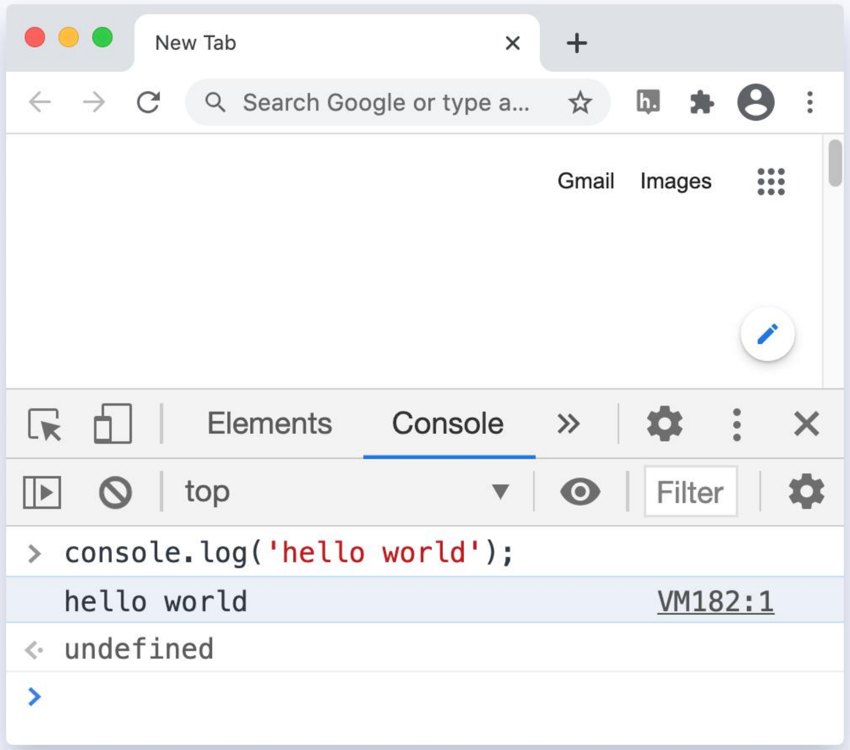
Using Console Tab of Web Browsers

- All the popular web browsers have built-in JavaScript engines. Hence, you can run JavaScript on a browser. To run JavaScript on a browser,
- Open your favorite browser (here we use Google Chrome).
- Open the developer tools by right clicking on an empty area and select Inspect. Shortcut: F12.



Using Console Tab of Web Browsers

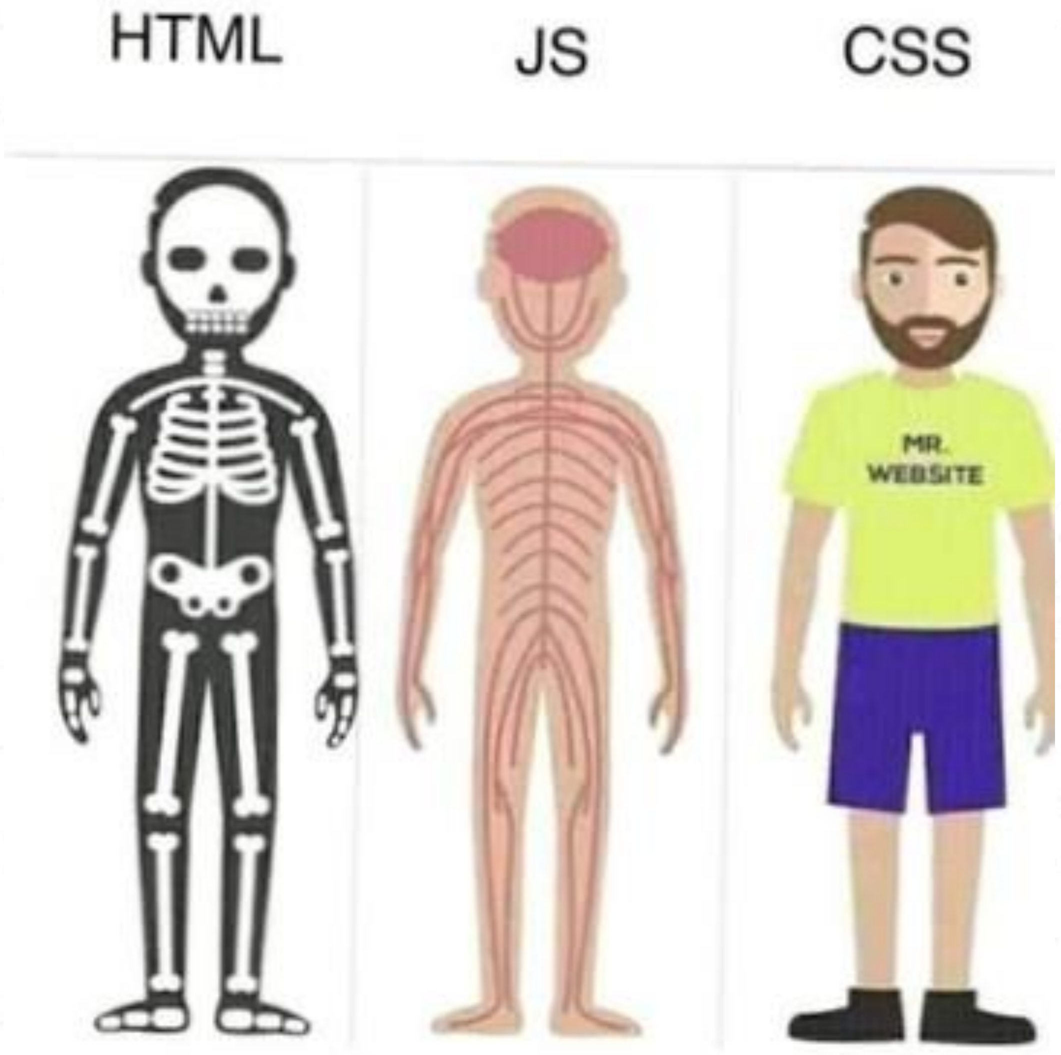
- On the developer tools, go to the console tab. Then, write JavaScript code and press enter to run the code.



```
> console.log('hello world');
  hello world                                VM182:1
<· undefined
>
```

SLIIT FACULTY OF COMPUTING

Why JavaScript?



Methods of using JavaScript

Methods of using JavaScript

1. Internal Script
   - Scripts in the <body> section
   - Scripts in the <head> section

1. External Script files

Internal script

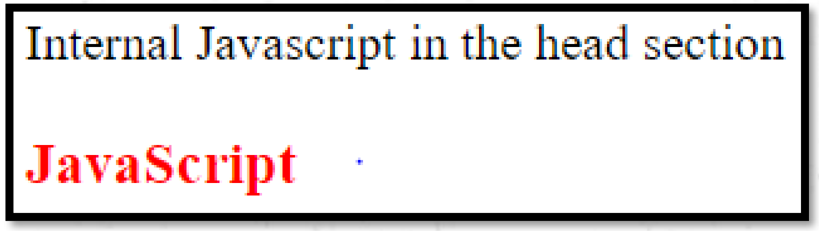
- JavaScript is embedded into the HTML document using the **script** element

```
<script >

        //JS code

</script>
```

Internal script Example in the <head> section

```
<html>
<head>
<title>Internal JavaScript</title>
<script>
      document.write("Internal JavaScript in the head section");
</script>
</head>

<body>
       <h3 style="color:red;"> JavaScript </h3>
</body>
</html>
```

output



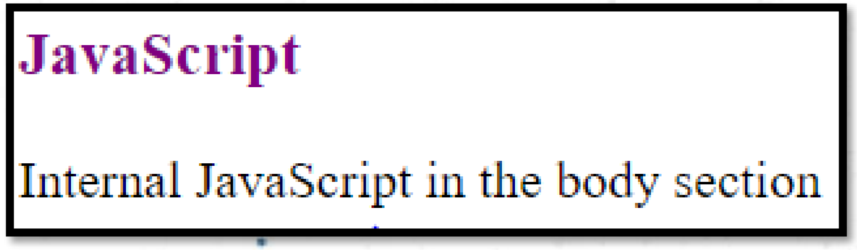
```
Internal Javascript in the head section

JavaScript
```

SLIIT
FACULTY OF COMPUTING

Internal script Example in the <body> section

```
<html>
<head>
<title>Internal JavaScript</title>
</head>
<body>
  <h3 style="color:purple;"> JavaScript </h3>
  <script>
    document.write("Internal JavaScript in the body section");
  </script>
</body>
</html>
```

output


JavaScript

Internal JavaScript in the body section

SLIIT
FACULTY OF COMPUTING

External script

- The external JS file should use the extension as .js

- External file is linked to the web page in head using the **script** element

- The script element uses the **src** attribute to specify the URL of the source JS file

  src=*"<Location>/<FileName>.js"*

External script

External script file linking

```
<head>
        <script src="…/ClientScripts/MainJS.js"> </script>
</head>
```

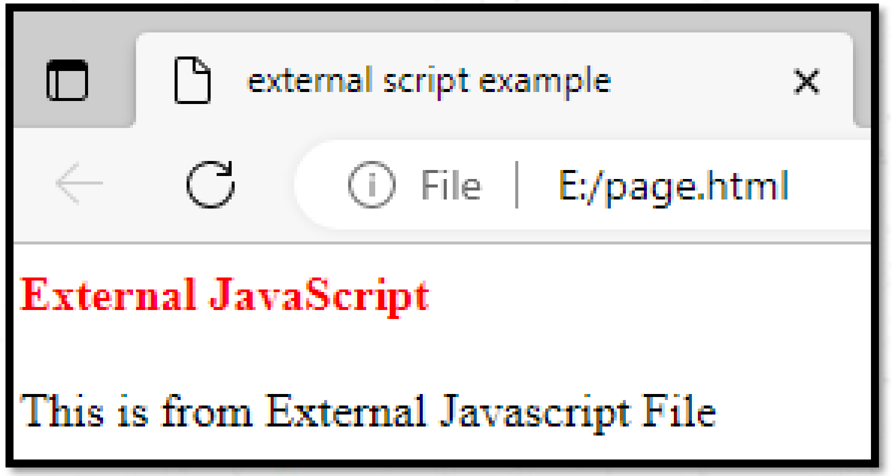
- The same JS file can be linked to multiple pages

External JavaScript Example

Page.html

```
<html>
<head>
        <title>
                external script example
        </title>
</head>
<body>
        <h4 style="color:red">External JavaScript</h4>
        <script src="ex1.js"></script>
</body>
</html>
```

output



External JavaScript

This is from External Javascript File

ex1.js

```
document.write("This is from External Javascript File");
```

Variables in JS

Data types in JS

1. Numerical
   - Integers – 1, 2, 3, -56, -135, 3464
   - Floating point/Decimal – 34.46, -65.135
2. Strings
   - Single characters – “a”, “b”, “c”, “2”, “7”
   - Multiple characters – “abc”, “12/04/2012”, “34”
3. Boolean – true / false
4. Null
5. Undefined

Data types

Note:

- JavaScript does not make a distinction between integer values and floating point values.
- All numbers in JavaScript are represented as floating-point values.

Variable declaration

The keyword **var and let** is used to declare a variable

- Examples
  - var age;
  - var smallNumber;
  - var initial
  - var name
  - var isPassed;
  - var num1, num2, num3;

Standars for the variable name

- You should not use any of the JavaScript **reserved keywords** as a variable name.
- No spaces
- Meaningful
- Use camel case

abstract	else	Instanceof	switch
boolean	enum	int	synchronized
break	export	interface	this
byte	extends	long	throw
case	false	native	throws
catch	final	new	transient
char	finally	null	true
class	float	package	try
const	for	private	typeof
continue	function	protected	var
debugger	goto	public	void
default	if	return	volatile
delete	implements	short	while
do	import	static	with
double	in	super	

SLIIT FACULTY OF COMPUTING

Variable Initialization

```
var age = 20;
var height = 5.5;
var initial = "K";
var name = "Kamal";
var isPassed = true;
```

Assign values to the variables

```
age = 20;
height = 5.5;
initial = "K";
name = "Kamal";
isPassed = true;
```

JavaScript Constant Variables

The **const** keyword was also introduced in the **ES6(ES2015)** version to create constants. For example,

```
const x = 5;
x = 10;   // Error! constant cannot be changed
console.log(x)
```

Also, you cannot declare a constant without initializing it. For example

```
const x; // Error! Missing initializer in const declaration
x = 5;
console.log(x)
```

Read and use the variable value

```
var age = 20; //Declare and initialize the variable
document.write(age);

age = 25; //Assign a new value to the variable
document.write("<br>Modified age = " + age);
```

SLIIT
FACULTY OF COMPUTING

JS is a weakly typed language

```
<html>
 <head><title>JavaScript Page</title> </head>
<body><script type="text/javascript">
                    document.write("4"/3);
                    document.write("<br>");
                    document.write("5" +5);
                    document.write("<br>");
                    document.write("5"- 3);
                    document.write("<br>");
                    document.write("5"*"5");
                    document.write("<br>");
                    document.write(4*3);
                    document.write("<br>");
                    document.write(5* "5");
 </script>
  <h1>Hello world</h1>
</body>
</html>
```

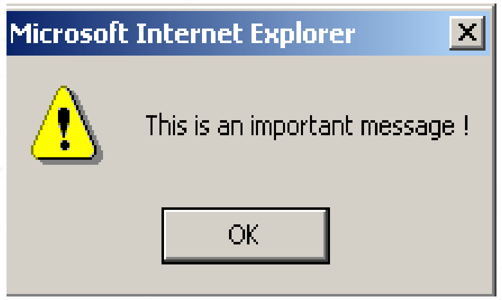
output

```
1.3333333333333333
55
2
25
12
25
```

Hello world

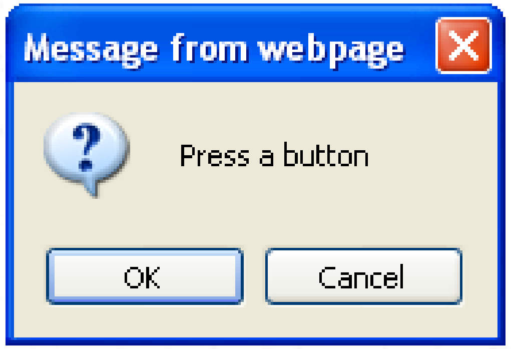
JavaScript popup boxes

- Alert box
- alert (“This is an important message !”);



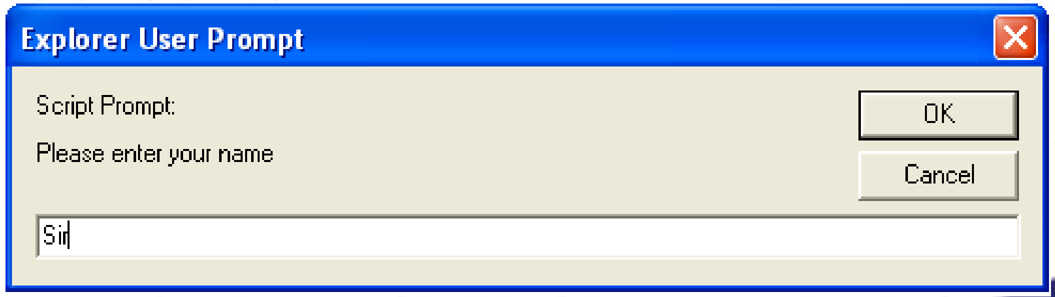
This is an important message !

- Confirm box
- var response=confirm("Press a button");



Press a button

- Prompt box
- var name=prompt("enter your name",“Sir");



Script Prompt:
Please enter your name
Sir

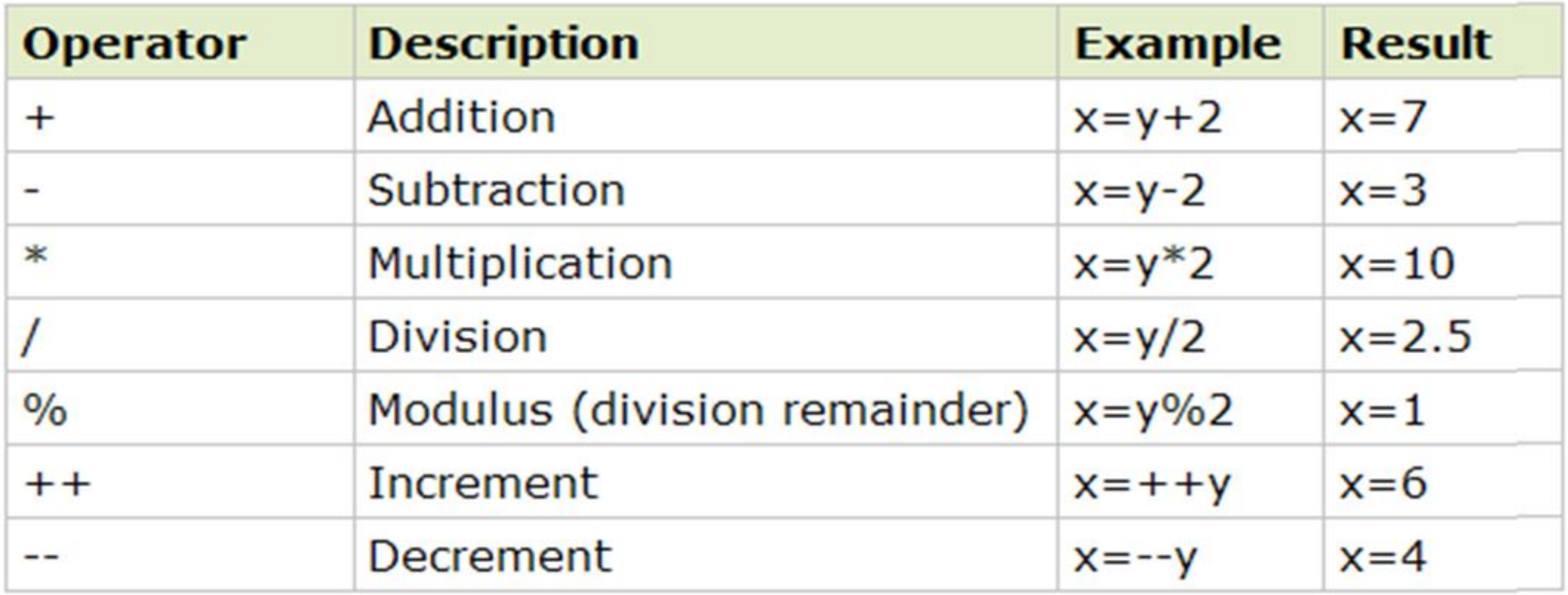
SLIIT FACULTY OF COMPUTING

Operators in JS

SLIIT
FACULTY OF COMPUTING

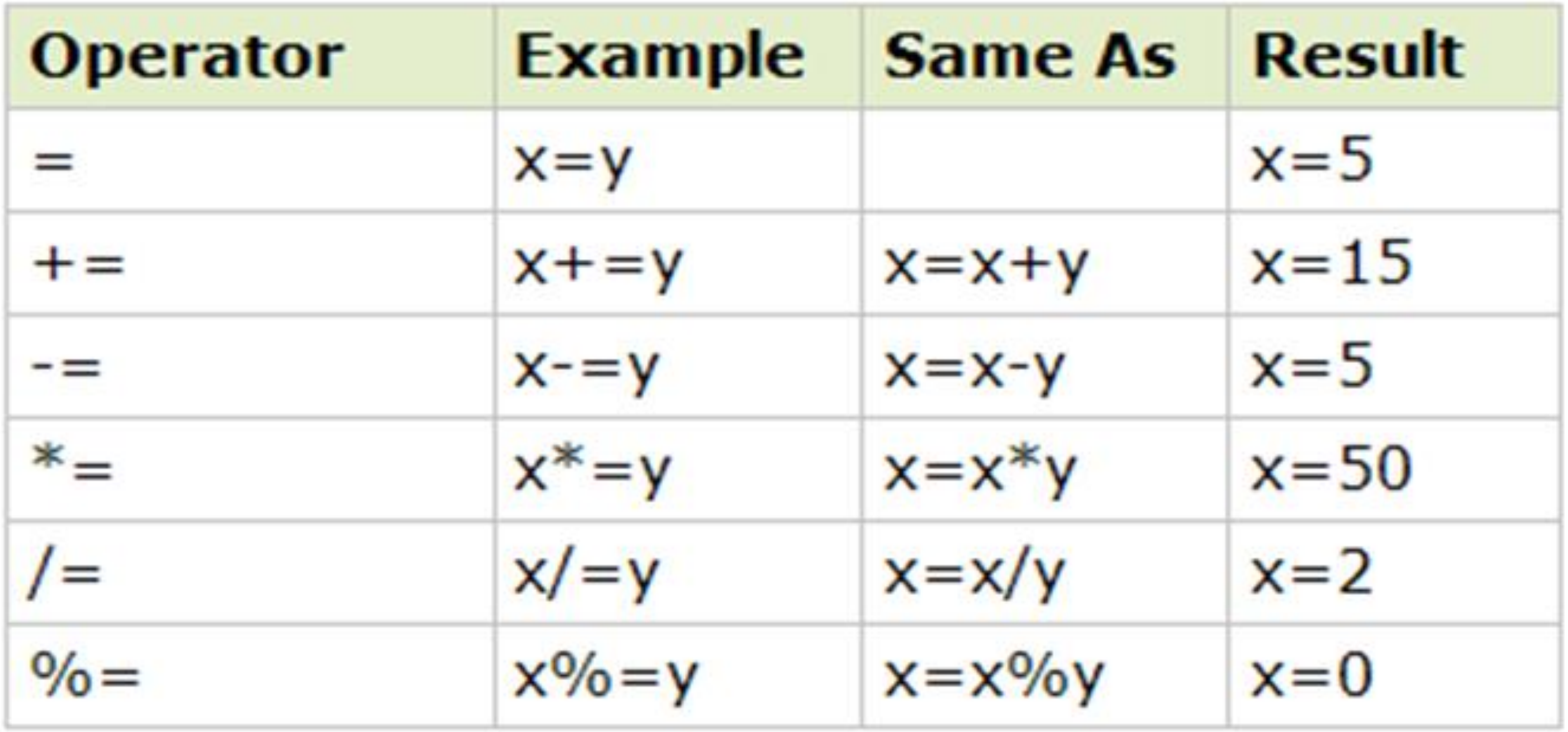
- Arithmetic Operators
- Assignment Operators
- Comparison Operators
- Logical Operators

Arithmetic Operators



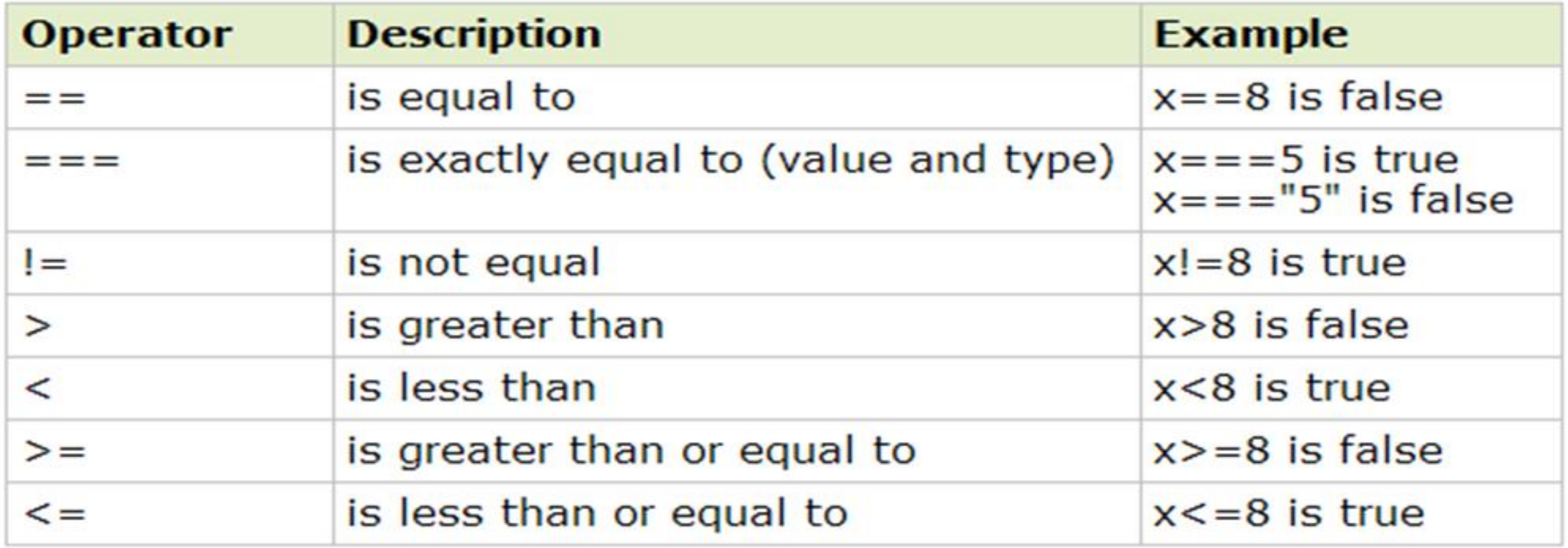
Operator	Description	Example	Result
+	Addition	x=y+2	x=7
-	Subtraction	x=y-2	x=3
*	Multiplication	x=y*2	x=10
/	Division	x=y/2	x=2.5
%	Modulus (division remainder)	x=y%2	x=1
++	Increment	x=++y	x=6
--	Decrement	x=--y	x=4

Assignment Operators



Operator	Example	Same As	Result
=	x=y		x=5
+=	x+=y	x=x+y	x=15
-=	x-=y	x=x-y	x=5
*=	x*=y	x=x*y	x=50
/=	x/=y	x=x/y	x=2
%=	x%=y	x=x%y	x=0

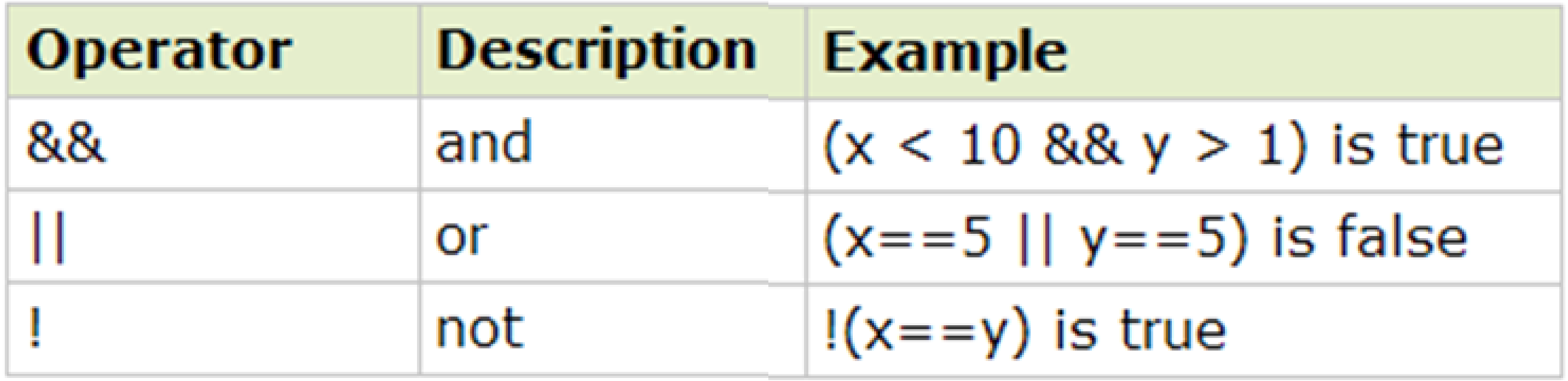
Comparison Operators



Operator	Description	Example
==	is equal to	x==8 is false
===	is exactly equal to (value and type)	x===5 is true x==="5" is false
!=	is not equal	x!=8 is true
>	is greater than	x>8 is false
<	is less than	x<8 is true
>=	is greater than or equal to	x>=8 is false
<=	is less than or equal to	x<=8 is true

http://www.w3schools.com/js/js_operators.asp

Logical Operators



Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
||	or	(x==5 || y==5) is false
!	not	!(x==y) is true

http://www.w3schools.com/js/js_operators.asp

SLIIT
FACULTY OF COMPUTING

Control Structures in JS

SLIIT
FACULTY OF COMPUTING

Selection / Branching

- Simple if-else
- If-else ladder
- Nested if-else
- Switch

Repetition / Iteration / Looping

- While loop
- For loop

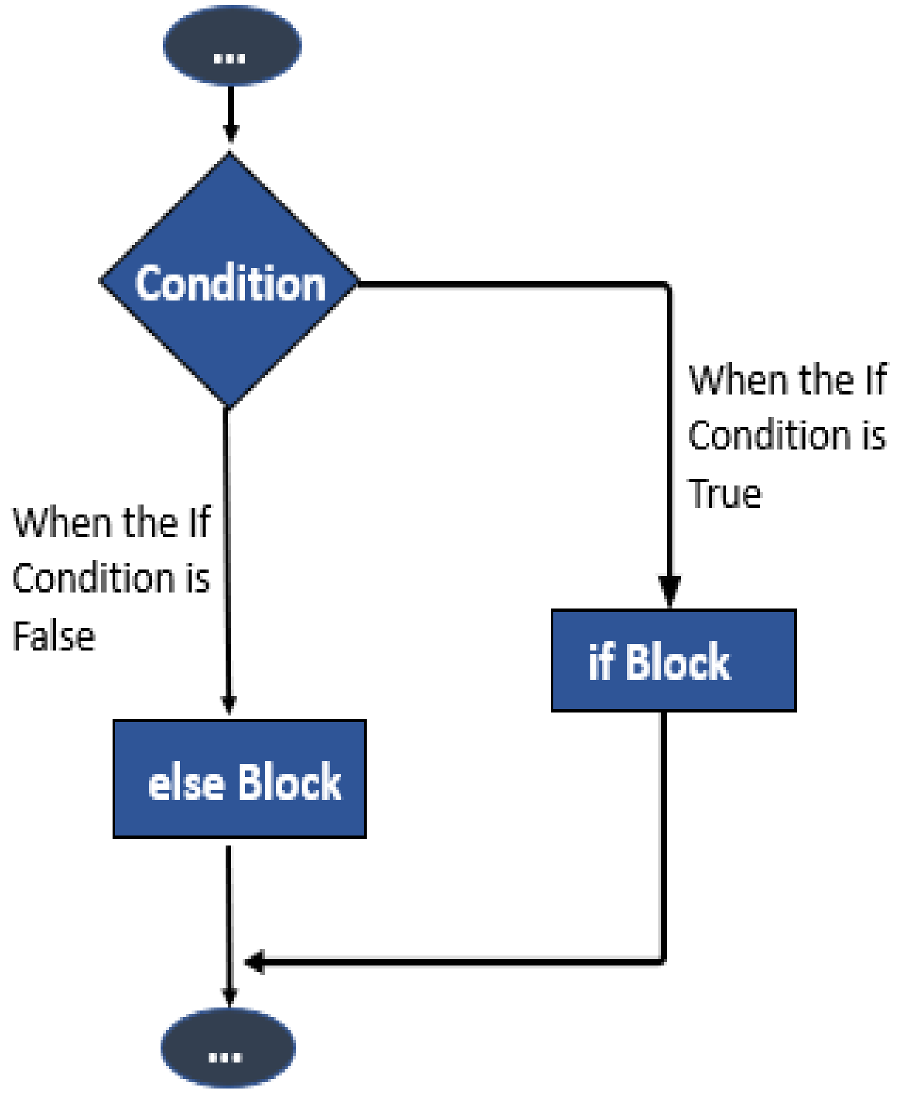
Simple if-else

- Used to divide the algorithm execution path into branches based on conditions
- Conditions produce Boolean results
- **If** the condition is true – we can do something
- **Else** we can do some other thing

Simple if-else

```
if (<Condition>)
{
        //Do something
}
else
{
        //Do some other thing
}
```

- **Else is optional**



Simple if-else example

- User enters the mark for Maths.
  - If the mark is greater than or equals 50 then display a message “Pass”
  - Else display a message “Fail”

Simple if-else example

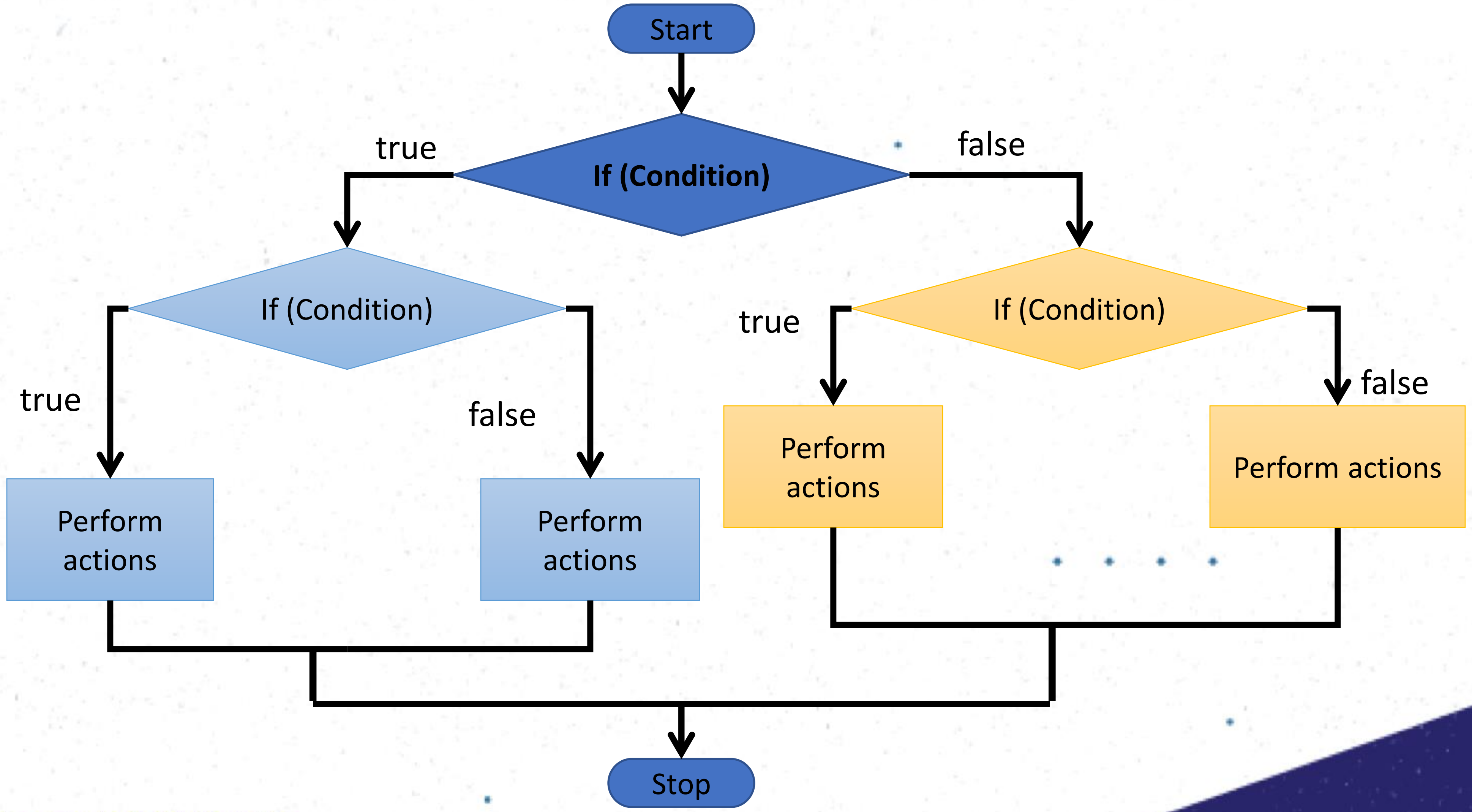
```
if(mark >= 50)
{
  document.write ("Pass");
}
else
{
        document.write ("Fail");
}
```

Simple if-else example

```
// check is the number is positive or negative/zero
 const number = prompt("Enter a number: ");

// check if number is greater than 0
 if (number > 0) {
         console.log("The number is positive");
 } // if number is not greater than 0

 else {
         console.log("The number is either a negative number or 0");
 }


console.log("The if...else statement is easy");
```

SLIIT
FACULTY OF COMPUTING

Nested if-else



Nested if-else

```
if(<Condition1>)
{
        if(<Condition2>)  {  //Actions  }
        else  {  //Actions  }
}
else
{
        if(<Condition3>)  {  //Actions  }
        else  {  //Actions   }
}
```

Nested if-else

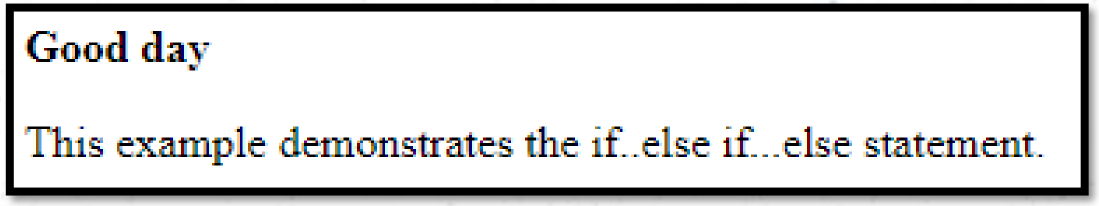
- Are these equivalent?

```
if ( age < 12 ) {
  entry = "free";
} else if ( age < 18 ) {
  entry = "£10";
} else {
  entry = "£20";
}
```

```
if ( age < 18 ) {
  entry = "£10" ;
} else if ( age < 12 ) {
  entry = "free";
} else {
  entry = "£20";
}
```

Nested if-else example

output



Good day

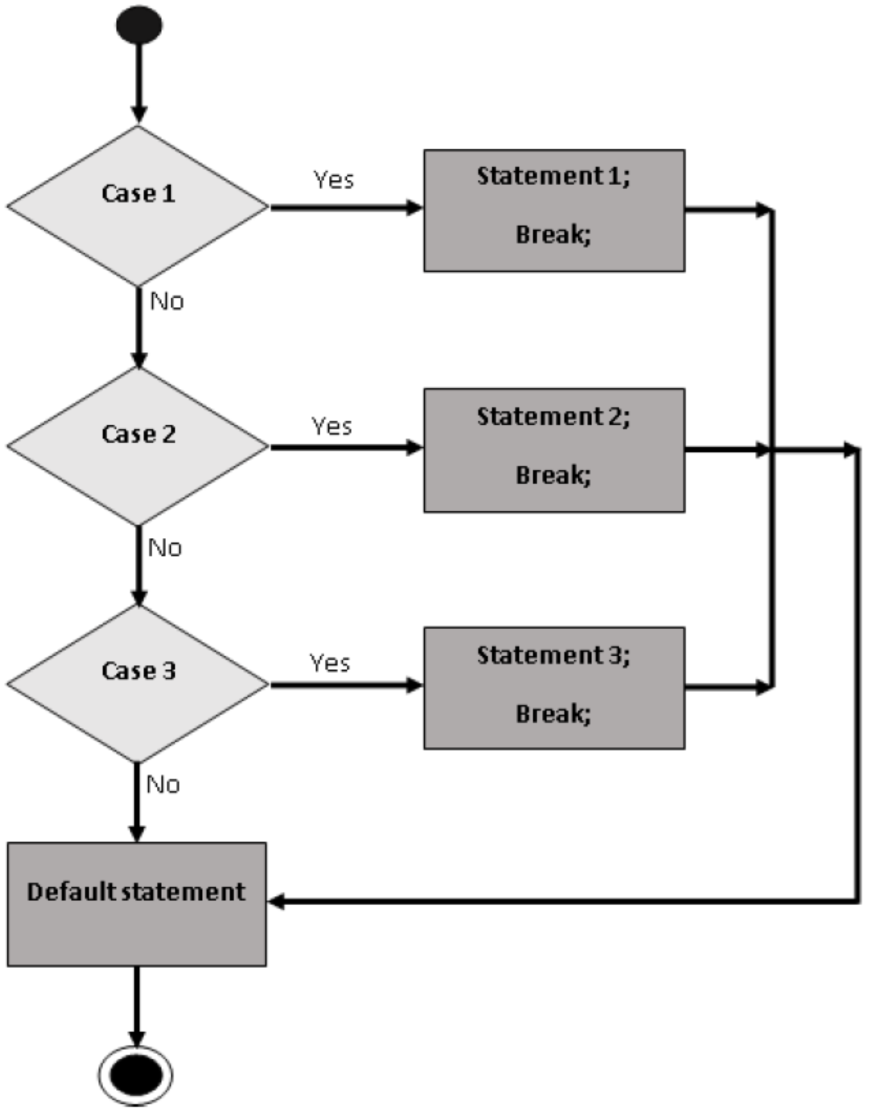
This example demonstrates the if..else if...else statement.

```
<html>
<body>
<script type="text/javascript">
var d = new Date();
var time = d.getHours();
if (time<10)
{
   document.write("<b>Good morning</b>");
}
else if (time>=10 && time<16)
        {
           document.write("<b>Good day</b>");
        }
        else
        {
           document.write("<b>Hello World!</b>");
        }
</script>
<p>
This example demonstrates the if..else if...else statement.
</p>
</body>
</html>
```

SLIIT FACULTY OF COMPUTING

Switch

```
switch(n)
{
case 1:
  execute code block 1
  break;
case 2:
  execute code block 2
  break;
default:
  code to be executed if n is different
  from case 1 and 2
}
```



Switch example

```
var grade=“B”;
switch (grade)
    {
    case "A":
         alert("Excellent");
         break;


    case "B":
         alert("Good");
         break;

    default:
         alert("Average");
         break;
}
```

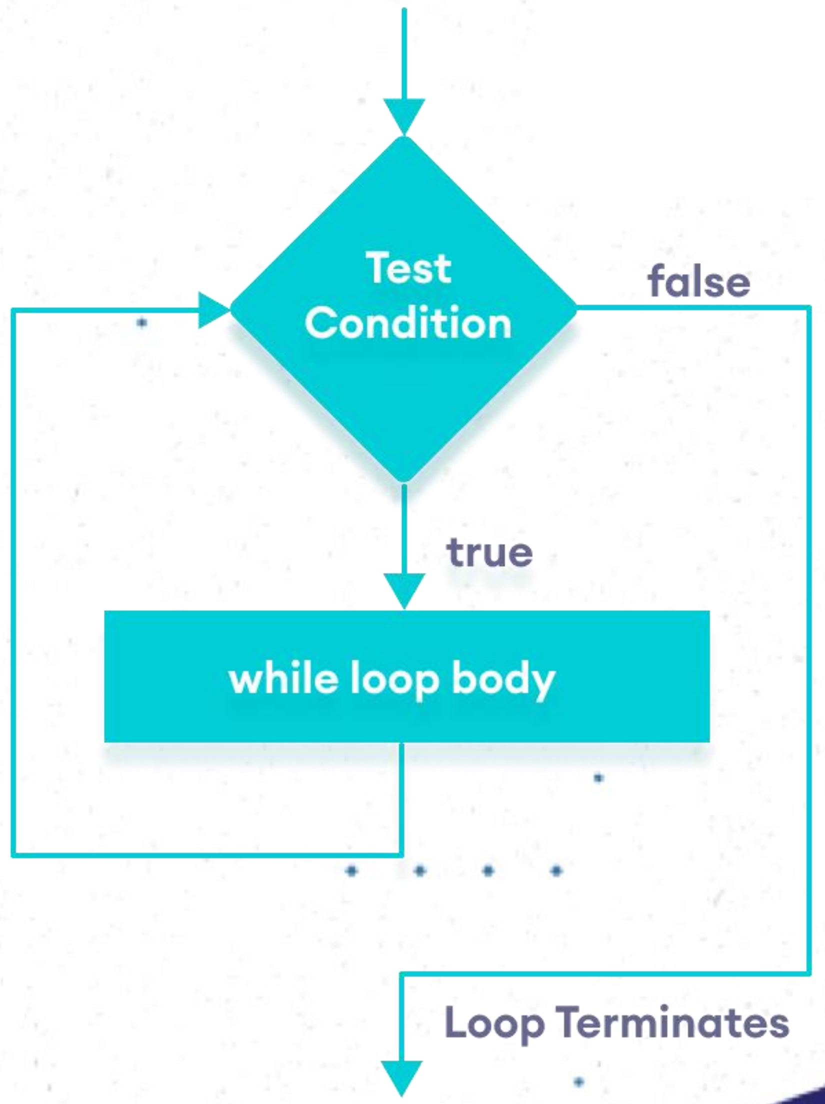
Switch example

```
<html>
<body>
<script type="text/javascript">
var d = new Date();
theDay=d.getDay();
switch (theDay)
{
case 5:
  document.write("<b>Finally Friday</b>");
  break;
case 6:
  document.write("<b>Super Saturday</b>");
  break;
case 0:
  document.write("<b>Sleepy Sunday</b>");
  break;
default:
  document.write("<b>I'm really looking forward to this weekend!</b>");
}
</script>
<p>This JavaScript will generate a different greeting based on what day it is. Note that
    Sunday=0, Monday=1, Tuesday=2, etc.</p>
</body>
</html>
```

SLIIT FACULTY OF COMPUTING

while loop

- The purpose of a **while** loop is to execute a statement or code block repeatedly as long as an **expression** is **true**.
- Once the expression becomes **false**, the loop terminates.

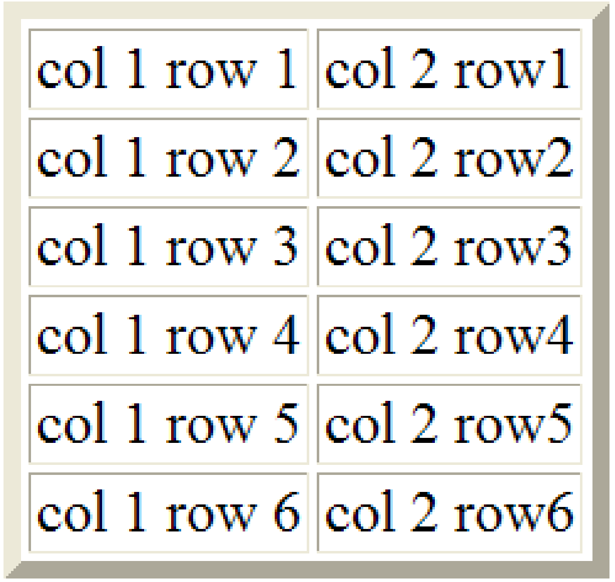


SLIIT
FACULTY OF COMPUTING

while loop example

```
<html>
<body>
 <table border="5">
  <script>
   var i=1;
     while (i<=6)
        {
             document.write("<tr>”);
             document.write("<td>col 1 row " + i + "</td>")
             document.write("<td>col 2 row " + i + “</td>");
             document.write("</tr>");
            i++;
            }
  </script>
 </table>
</body>
</html>
```

output

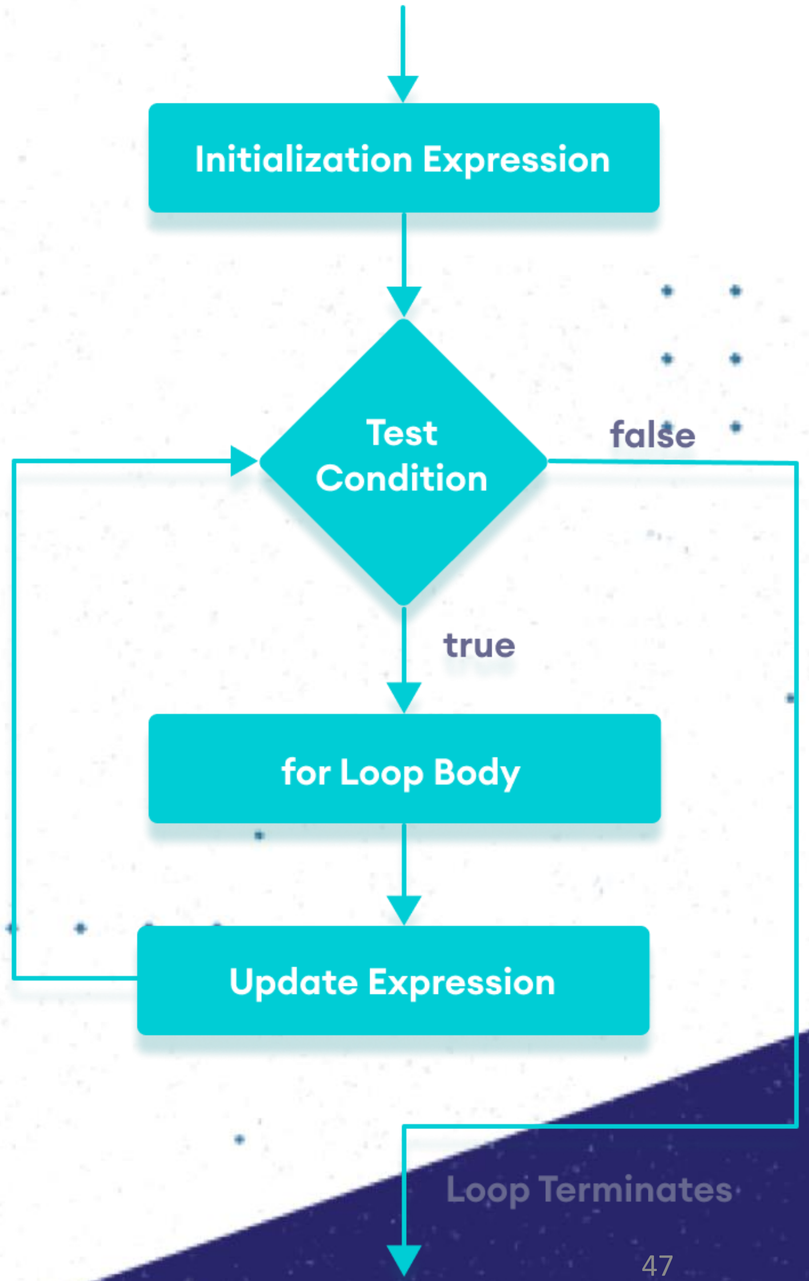


col 1 row 1	col 2 row1
col 1 row 2	col 2 row2
col 1 row 3	col 2 row3
col 1 row 4	col 2 row4
col 1 row 5	col 2 row5
col 1 row 6	col 2 row6

SLIIT
FACULTY OF COMPUTING

for loop

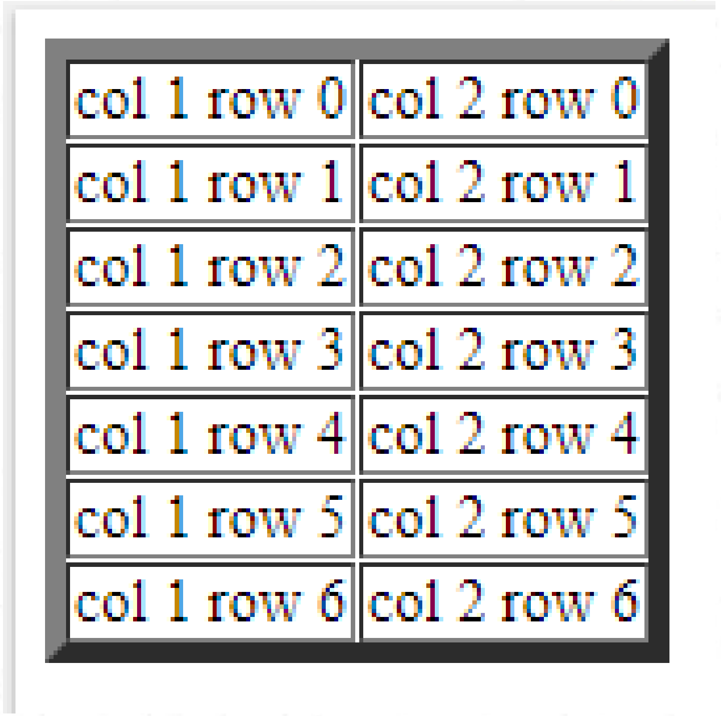
```
for (var=startvalue; var<=endvalue; var=var+increment)
{
        //code to be executed
}
```



for loop example

```
<html>
<body>
 <table border="5">
  <script>
          for (i=0;i<=6;i++)
        {
         document.write("<tr>");
         document.write("<td>col 1 row " + i + "</td>")
         document.write("<td>col 2 row " + i + "</td>");
         document.write("</tr>");
         //i++;
        }
  </script>
 </table>
</body>
</html>
```

output



col 1 row 0	col 2 row 0
col 1 row 1	col 2 row 1
col 1 row 2	col 2 row 2
col 1 row 3	col 2 row 3
col 1 row 4	col 2 row 4
col 1 row 5	col 2 row 5
col 1 row 6	col 2 row 6

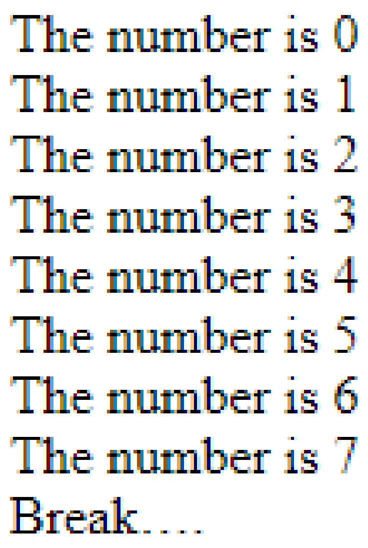
- The break Statement
  - The break statement will break the loop and continue executing the code that follows the loop (if any).

- The continue Statement
  - The continue statement will break the current loop and continue with the next iteration.

Break statement example

```
<html>

<body>

<script

for (i=0;i<=10;i++){

     if (i==8)

     {

                    break;

     }

  document.write("The number is " + i);

  document.write("<br />");

}

document.write("Break....");

</script>

</body>

</html>
```

output



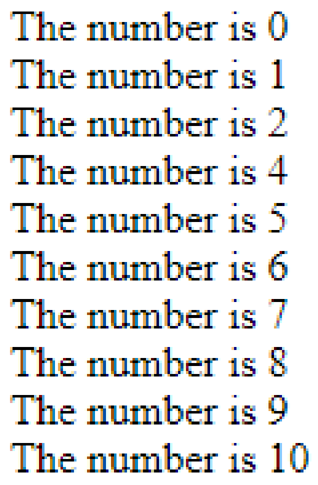
```
The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
The number is 7
Break....
```

SLIIT
FACULTY OF COMPUTING

Continue statement example

```
<!DOCTYPE html>
<html>
<body>
<script>
var i=0
for (i=0;i<=10;i++)
{
   if (i==3)
   {
      continue;
   }
 document.write("The number is " + i);
 document.write("<br />");
}
</script>
</body>
</html>
```

output



```
The number is 0
The number is 1
The number is 2
The number is 4
The number is 5
The number is 6
The number is 7
The number is 8
The number is 9
The number is 10
```

Summary

- Introduction to the JavaScript
- Variables in JS
- Operators in JS
- Control structures in JS