Future University in Egypt
Faculty of Computers and Information Technology
Department of Computer Science

Prepared by	
ID	**Name**
20184017	**Ahmed hussein ahmed**

Supervised by
Ta, Hesham Mohamed

1-Rem vs Em

EM: is relative to the parent element's font size, so if you wish to scale the element's size based on its parent's size, use EM.

REM: is relative to the root (HTML) font size, so if you wish to scale the element's size based on the root size, no matter what the parent size is, use REM. If you've used EM and are finding sizing issues due to lots of nested elements, REM will probably be the better choice.

2-css position

The `position` property specifies the type of positioning method used for an element.

There are five different position values:

- `static`
- `relative`
- `fixed`
- `absolute`
- `sticky`

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the `position` property is set first. They also work differently depending on the position value.

1-position: static;

HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

This element has position: static

2- position: relative;

An element with position: relative; is positioned relative to its normal position.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

This element has position: relative;

3- position: fixed;

An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

A fixed element does not leave a gap in the page where it would normally have been located.

Notice the fixed element in the lower-right corner of the page.

4- position: absolute;

An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

5- position: sticky;

An element with position: sticky; is positioned based on the user's scroll position.

A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed).

3- for vs while loop

1- for

The For Loop

The for statement creates a loop with 3 optional expressions:

```
for (expression 1; expression 2; expression 3) {
  // code block to be executed
}
```

Expression 1 is executed (one time) before the execution of the code block.
Expression 2 defines the condition for executing the code block.
Expression 3 is executed (every time) after the code block has been executed.

2-while loop

The while loop loops through a block of code as long as a specified condition is true.

Syntax

```
while (condition) {
  }
```

If you forget to increase the variable used in the condition, the loop will never end. This will crash your browser.

The Do While Loop

The do while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

Syntax
do {

}
while (*condition*);
Example
The example below uses a do while loop. The loop will always be executed at least once, even if the condition is false, because the code block is executed before the condition is tested:

4- object method

1- An object is a collection of key/value pairs or proptis When the value is a function, the property becomes a method. Typically, you use methods to describe the object behaviors.

2- JavaScript Methods
JavaScript methods are actions that can be performed on objects.
A JavaScript method is a property containing a function definition.

5- Arrow and Regular Functions

1- **Arrow function**

The behavior of this inside of an arrow function differs considerably from the regular function's this behavior. The arrow function doesn't define its own execution context.

No matter how or where being executed, this value inside of an arrow function always equals this value from the outer function. In other words, the arrow function resolves this lexically.

In the following example, myMethod() is an outer function of callback() arrow function:

2- **Regular function**

nside of a regular JavaScript function, `this` value (aka the execution context) is dynamic.

The dynamic context means that the value of `this` depends on *how* the function is invoked. In JavaScript, there are 4 ways you can invoke a regular function.

During a *simple invocation* the value of `this` equals to the global object (or `undefined` if the function runs in strict mode):

6- object vs instance oop

The basic concept of OOP is this: Class >> Object >> Instance. The class = the blue print. The Object is an actual thing that is built based on the 'blue print' (like the house). An instance is a virtual copy (but not a real copy) of the object.