Memo	**IMED 2351**
Subject	JavaScript Conditionals, Loops

- **Part III**
  - Conditionals & Flow statements
    - If statement
    - Else
    - Else If
    - Switch
    - ? (ternary / conditional operator)
    - **Loops**
      - while()
      - do . . . while
      - for

<u>**Conditionals**</u>

<u>If statement</u>

- execute a specific set of code if the evaluation of a **conditional expression** is true
- contains 3 parts
    - If keyword
    - Conditional (with parenthesis)
    - Statement(s)
- If only 1 statement, then don't need {} brackets, but best to use anyway.
- The ; (semi-colon) must be at the end of any **statement** not brackets
- You can have more than one statement within an if

```
if ( conditional expression )
{
        statement(s);
}
```

<u>if . . . else</u>

- follows same logic as if, but uses the else as a back-up plan
- use the same bracketing rules
- can contain multiple statement within if or else

```
if ( conditional expression )
{
        statement(s);
} else {
        statement(s);
}
```

<u>if . . . elseif</u>

- provides a secondary comparison condition
- can use as many elseif as you want
- if one comparison fails, the next will check a different conditional
- can use the else by itself at the end if all other if's/elseif's are not true

```
if ( conditional expression )
{
        statement(s);
} elseif (conditional expression)
 {
        statement(s);
}
```

switch

- controls the program flow by executing a specific set of statements, depending on the value of an expression
- uses a **case** label
- if **case** value matches value of **switch** value then it will execute the statement beneath the **case**
- uses a **default** label
- always use **break** to indicate the end of a statement with the **case** block

```
switch (expression)
{
        case value1:
                statement 1;
                break;
        case value2:
                statement 2;
                break;
        default:
                statement 3;
                break;
}
```

? (ternary)

- similar to if statement
- except everything is evaluated on one line
- returns a value derived from one of two expressions separated by a colon
- uses a test expression
- used when the developer wants to quickly (shortcut) evaluate something without writing out lots of code to do so

```
test expression ? if true : if false;
```

Loops

while()

- repeats a statement or series of statements as long as a given conditional expression evaluates to true
- when the conditional = false the while loop ends

Syntax
 	
while (expression)
{
    statement(s);
}
	
	<script>
	    var y = 1;
	    while (y <= 10)
	    {
	        document.write(“y is ” + y);
	        y++;
	    }
	</script>

do while()

- executes a statement at least once then repeats as long as the conditional = true
- need to place inside of this to prevent an infinite loop

Syntax
 	
do
{
    statement(s);
} while (expression);
	
	<script>
	    var x = 1;
	    do
	    {
	        document.write(“x is “ + x);
	        x++;
	    } while (x <= 10);
	</script>

for

- also loops through code
- repeating a statement(s) as long as a given condition = true
- just like while, except the for contains code to eventually stop it
- good to prevent infinite loops

Syntax

```
for (initialization; condition; increment)
{
    statement(s);
}
```

```
<script>
    for(var z = 1; z <= 10; z++)
    {
        document.write(“z is “ + z);
    }
</script>
```