Javascript Commands
Without Commentary
(for use as reference on exams and quizzes)

Embedding Javascript

```
<html>
   <head>
      ...
      <script type="text/javascript" src="path/to/script1.js"
          defer></script>
      <script type="text/javascript" src="path/to/script2.js"
          defer></script>
    ...
  </head>
<body>
  ...
  <script>
     // Javascript code goes here
  </script>
  ...
</body>
</html>
```

Variables, Types, Comments, and Semicolons

```
/* When we declare variables we don't give
   them a type. We only specify a name. */

let value, name, isHot, fun

value = 1         // type is Number
name = 'Crazy'    // type is String
isHot = true      // type is Boolean
```

value	1	name	'Crazy'
isHot	true	fun	undefined

Displaying Values

```
let name
name = 'Crazy'
console.log(name)       // write to browser console
alert(name)             // present in annoying pop-up box
document.write(name)    // place on page in current location
```

Output of all three statements (but in 3 different locations):

```
Crazy
```

Getting Input

```
1  let doMore, num
2  doMore = confirm("Continue?")
3  num = prompt("Enter fav number: ")
4  console.log(doMore)
5  console.log(typeof(doMore))
6  console.log(num)
7  console.log(typeof(num))
```

Of course the output depends on what the user does. Suppose the user clicks OK at the first prompt generated by line 2 and then types 10.2 followed by OK at the second prompt generated by line 3. Expected output (in the console):

```
true
boolean
10.2
string
```

Arithmetic

```
1  let a, b, c, d, e, f, g, h
2  a = "5.5"
3  b = 3
4  c = a + b
5  d = b + Number(a)
6  b++
7  e = b / 3
8  f = Math.floor(b / 3)
9  g = b % 2
10 h = Math.random()
```

a	”5.5”
b	4
c	”5.53”
d	8.5
e	1.333333
f	1
g	0
h	0.7697511504757932

If Statements

```
let x, y
x = "5.5"
y = 0
if (x == 5.5) {
   console.log("A is true")
}
else {
   console.log("A is false")
}
if (x === 5.5) {
   console.log("B is true")
}
else {
   console.log("B is false")
}
if (x) {
   console.log("C is true")
}
else {
   console.log("C is false")
}
if (y) {
   console.log("D is true")
}
else {
   console.log("D is false")
}
```

Expected output (in console.log):
A is true
B is false
C is true
D is false

Boolean Expressions & Relational Operators

```
let isCook, a, b, c, d, e, f, g, h
isCool = false
a = 7
b = -1
c = true
d = a < b
e = a < b || a > 0
f = a <= b && a != 0
g = !isCool && a == 7
h = 0 <= b <= 100  // not what you expect!
```

a	7
b	-1
c	true
d	false
e	true
f	false
g	true
h	true

Strings

```
let one, two, num
let a, b, c, d, e, f, g, h
one = "The quick brown fox"
two = ' jumps.'
num = two.length  // 7
console.log(one + two)
if (one > "ABC") {
   console.log("bigger")
}
a = one[4]
b = one.substr(4,7) // 7 chars starting at 4
c = one.substr(15)  // skip first 15
d = one.substr(-2)  // keep last 2
e = two.replace("jump", "leap")
f = two.replace(/\w+/, "xxx")
g = two.trim()  // remove leading/trailing whitespace
h = two.toUpperCase()
```

num	7
a	"q"
b	"quick b"
c	" fox"
d	"ox"
e	" leaps."
f	" xxx."
g	"jumps."
h	" JUMPS."

Expected output:

```
The quick brown fox jumps.
bigger
```

Regular Expressions

```
let str, a, b, c, d, e, f, g
str = "There are 53 people with SSN 123-12-1234."
a = str.search(/fun/)
b = str.search(/re/)
c = str.search(/[aeiou]/)
d = str.search(/\d/)
e = str.search(/\d{3}-\d{2}-\d{4}/)
f = str.search(/\s/)
g = str.search(/^SSN/)
```

a	-1
b	3
c	2
d	10
e	29
f	5
g	-1

Loops

```
for (let i = 0; i < 3; i++) {
   console.log(i)
}

let val = 40
while (val > 0) {
   console.log(val)
   val -= 10
}

let count = 100
do {
   console.log(count)
   count += 100
} while (count <= 400)
```

Expected output:

```
0
1
2
40
30
20
10
100
200
300
400
```

Arrays

```
let a, b, c, val, str
a = ["zebra", 12, true, 17.3]
a.push("cool")
a[0] = 70
console.log(a)
console.log(a.length)
console.log(a[3])

a.sort()
console.log(a)
val = a.pop()
console.log(val)

b = [20, 30, 40]
for (let elem of b) {
   console.log(elem)
}
str= b.join(",")
console.log(str)
str = "Hi, how are you?"
c = str.split(" ")
console.log(c)
```

Expected output:

```
5
17.3
[12, 17.3, 70, "cool", true]
true
20
30
40
20,30,40
["Hi,", "how", "are", "you?"]
```

Spread Operator

```
// Spread operator on arrays in regular code
const nums = [12, 4, 32, 31, 19, 7]
console.log(nums)
console.log(...nums)

// Use spread operator to convert array to individual values
    for passing
// parameters to functions that require individual values.
let ans = Math.min(...nums)
console.log("Min is: " + ans)

// Spread operator on objects in regular code
const person = {
    name: 'Max',
    age: 29,
    greet() {
        console.log('Hi, I am ' + this.name)
    }
}

// Use spread operator to copy an object (not just its
    reference)
const copyOfPerson = {...person}
console.log(copyOfPerson)

// Use of spread operator in parameter list
const myMin = (...args) => {
   let min = args[0]
   for (let val of args) {
      if (val < min) {
         min = val
      }
   }
   return min
}

let ans2 = myMin(6, 20, 9, 34, 3, 22)
console.log("Ans: " + ans2)
```

Expected output (to console log):

```
[ 12, 4, 32, 31, 19, 7 ]
12 4 32 31 19 7
Min is: 4
{ name: 'Max', age: 29, greet: [Functio
Ans: 3
```

Functions

```
function showSum(a, b) {
   let c = a + b
   console.log(c)
}
function calcSum(a, b) {
   let c = a + b
   return c
}

// To use these functions
let x = 7.5
showSum(x, 1.0)
showSum(x * 2.0, x - 2.0)
x = calcSum(1.0, 2.2)
console.log(x)
```

Expected output:

```
8.5
20.5
3.2
```

Arrow Functions

```
// Old fashioned JS notation
function doubleIt0(val) {
   return 2 * val
}

// New-fangled JS notation
const doubleIt1 = (val) => {
   return 2 * val
}

// If the function takes only 1 argument then we can leave off ()'s.
// If the function only has a return statement we can leave off {}'s and return.
const doubleIt2 = val => 2 * val

console.log(doubleIt0(14.5))
console.log(doubleIt1(14.5))
console.log(doubleIt2(14.5))
```

Expected output (in console log):

```
29
29
29
```

Functions as Parameters

```
function doubleIt(x) {
   return 2 * x
}

let a, b, c, d, e, f, g
a = [25, 14, 75, 50, 22]
b = a.map(doubleIt)
console.log(b)
c = a.map(function doubleIt(x) {
            return 2 * x
         })
d = a.map(function (x) {
            return 2 * x
         })
e = a.map((x) => { return 2 * x })
f = a.map((x) => 2 * x )
g = a.map(x => 2 * x)
a.forEach((val, i) => console.log(i + " --> " +val))
```

Since we expect `b`, `c`, `d`, `e`, `f`, and `g` to end with the same contents we only display the array `b`. Expected output:

```
[50, 28, 150, 100, 44]
0 --> 25
1 --> 14
2 --> 75
3 --> 50
4 --> 22
```

Objects

```
let obj, obj2, str
obj = { name: "Fred", age: 10 }
obj2 = {
   name: "Alice",
   age: 12,
   display() {
      return this.name + " is " + this.age + " years old"
   }
}

console.log(obj.name)
console.log(obj['age'])
obj.name = "George"
console.log(obj.name)
str = obj2.display()
console.log(str)
```

Expected output:

```
Fred
10
George
Alice is 12 years old
```

Classes

```
class Animal {
   constructor(name, age) {
      this.name = name
      this.age = age
   }
   display() {
      console.log(this.name + " is " + this.age + " years old"
          )
   }
}
class Pet extends Animal {
   constructor(name, age, owner) {
      super(name,age)
      this.owner = owner
   }
   display() {
      console.log(this.name + " is owned by " + this.owner)
   }
}

let wild, pet
wild = new Animal("Pyro", 3)
wild.display()

pet = new Pet("Polly", 10, "B.Jones")
pet.display()
```

Expected output:

```
Pyro is 3 years old
Polly is owned by B.Jones
```

DOM Intro

```
<html>
   <head></head>
   <body>
      <h1>Greetings</h1>
      <div id="mydiv">
         <p>Welcome to this amazing page!</p>
         <p>I hope you like it.</p>
      </div>
      <p id="fun" class="cool neat">This is fun.</p>
      <script>
      // Here we use .innerHTML
         let elem
         elem = document.getElementById("fun")
         elem.style.backgroundColor = "#ccc"
         elem.style.border = "solid 1px black"
         elem.innerHTML = "New paragraph."
         console.log(elem.getAttribute("class"))

      // Here we use .insertAdjacentHTML
        let str = '<p>Great paragraph.</p>'
        elem = document.getElementById("mydiv")
         elem.insertAdjacentHTML('afterbegin',str)
      </script>
   </body>
</html>
```

The code modifies the screen to set the element whose id is “fun” to have a gray background and a border. Also, its contents are changed to hold “New paragraph.”. Output to the console is:

```
cool neat
```

```
<html>
   <head></head>
   <body>
      <h1>Greetings</h1>
      <p>Paragraph 1</p>
      <p class="even">Paragraph 2</p>
      <p>Paragraph 3</p>
      <p class="even">Paragraph 4</p>
      <p id="last">Paragraph 5 <span>The end</span></p>
      <script>
         let elem, arr, elem2
         elem = document.querySelector(".even")
         console.log(elem.innerHTML)

         arr = document.querySelectorAll("p")
         console.log(arr.length)

         // give all even paragraphs a grey background
         arr = document.querySelectorAll(".even")
         console.log(arr[1].innerHTML)
         for (elem of arr) {
            elem.style.backgroundColor = "#eee"
         }

         // Alternate way to make a change to all elements
         // that have been selected using querySelectorAll()
         [...document.querySelectorAll('.even')].forEach((
             element) => {
            element.style.color = 'green'
         })

         // search inside of an element
         elem = document.querySelector("#last")
         elem2 = elem.querySelector("span")
         console.log(elem2.innerHTML)

         // there is no span in first paragraph
         elem = document.querySelector("p")
         elem2 = elem.querySelector("span")
         console.log(elem2)
      </script>
   </body>
</html>
```

The even numbered paragraphs will be given a grey background and the console will show these values:

```
Paragraph 2
5
Paragraph 4
The end
null
```

setInterval() and clearInterval()

```
<html>
   <head></head>
   <body>
      <h1>Behold setInterval()</h1>
      <script>
         let colorChanger
         let green = 0

         function changeColor() {
            green = (green + 1) % 256
            document.querySelector("h1").style.color = "rgba
                (0, " + green + ", 0, 1.0)"
            if (green == 0) {
               clearInterval(colorChanger)
            }
         }

         colorChanger = setInterval(changeColor, 10)
      </script>
   </body>
</html>
```

Events and Listeners

```
<html>
   <body>
      Username: <input type="text" name="username"> <br>
      Password: <input type="password" name="password"
                       onchange="bewareOfChange()"> <br>
      <button>Done</button>
      <script>
         let textbox, button
         function bewareOfChange() {
            alert("A change was made!")
         }
         button = document.querySelector('button')
         button.onclick = () => alert("Done!")
         textbox = document.querySelector('input[name="
             username"]')
         textbox.style.color = "#bbb"
         textbox.value = "Enter username"
         textbox.addEventListener("focus", function() {
            textbox.style.color = "#000"
            textbox.value = ""
         })
      </script>
   </body>
</html>
```

Event Handler for Multiple/Future Objects

```
<html>
   <body>
      <ul id="grocerylist">
         <li><input type="text" value="Eggs" /></li>
         <li><input type="text" value="Bread" /></li>
         <li><input type="text" value="Milk" /></li>
         <li><input type="text" value="Butter" /></li>
         <li><input type="text" value="Cheese" /></li>
      </ul>

      <script>
         var list = document.getElementById("grocerylist")
         list.addEventListener("input", function (event) {
            if (event.target.matches('input[type="text"]')) {
               // call validation function and pass
               // event.target as parameter
            }
         })
      </script>
   </body>
</html>
```

Client-Side Form Validation

```
<html>
   <body>
      <form method="post" action="passed.html"
            onsubmit="return formIsValid()">
         Name: <input type="text" name="name">
         <span>Name must have at least 1 character</span><br>
         SSN: <input type="text" name="ssn">
         <span>SSN must be ###-##-####</span><br>
         <input type="submit" value="Done">
      </form>

      <script>
         for (let elem of document.querySelectorAll('span')) {
            elem.style.display="none"
         }
         function formIsValid() {
            let name, ssn, valid
            valid = true
            name = document.querySelector('input[name="name"]'
                )
            if (name.value.search(/^\s*$/) == 0) {
               name.nextElementSibling.style.display="inline"
               valid = false
            }
            else {
               name.nextElementSibling.style.display="none"
            }
            ssn = document.querySelector('input[name="ssn"]')
            if (ssn.value.search(/^\d{3}-\d{2}-\d{4}$/) != 0)
                {
               ssn.nextElementSibling.style.display="inline"
               valid = false
            }
            else {
               ssn.nextElementSibling.style.display="none"
            }
            return valid
         }
      </script>
   </body>
</html>
```