

531489412 Argentina Programa Capitulo 2 Programacion Imperativa

Paradigmas de Programación (Universidad Nacional de Tucumán)



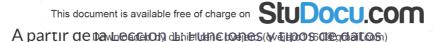
Apéndice

- Referencia rápida del lenguaje JavaScript
 - Declaración de Funciones
 - Operadores matemáticos
 - Operadores lógicos
 - Comparaciones
 - Alternativa Condicional
 - Variables
 - Repetición indexada
- Biblioteca simplificada
 - o longitud(unString)
 - o convertirEnMayuscula(unString)
 - comienzaCon(unString, otroString)
 - o imprimir(unString)
 - o tirarDado()
 - listasIguales
 - longitud(unaLista)
 - agregar(unaLista, unElemento)
 - o remover(unaLista)
 - posicion(unaLista, unElemento)
- Bibliografía complementaria

Referencia rápida del lenguaje JavaScript

El lenguaje JavaScript es utilizado ampliamente para construir software en todo el mundo, siendo una de las principales tecnologías de la Web. En Mumuki sólo usamos una muy pequeña parte del mismo, que listamos a continuación:

Declaración de Funciones



Las funciones en JavaScript se declaran mediante la *palabra clave* function , y su cuerpo va entre llaves { y } :

```
function nombreDeLaFuncion(parametro1, parametro2, parametro3)
{
   return ...;
}
```

Toda función debe tener al menos un retorno, que se expresa mediante return.

Operadores matemáticos

A partir de la Lección 1: Funciones y tipos de datos

```
4 + 5
10 - 5
8 * 9
10 / 5
```

Operadores lógicos

A partir de la Lección 1: Funciones y tipos de datos

```
true && false
true || false
! false
```

Comparaciones

A partir de la Lección 1: Funciones y tipos de datos

```
// para cualquier tipo de dato
"hola" === "hola"
"hola" !== "chau"

// para números
4 >= 5
4 > 5
4 <= 5
4 < 5</pre>
```

Alternativa ("Ondicional

/ Intermativa Demarcional

A partir de la Lección 1: Funciones y tipos de datos

Los if s en JavaScript encierran la condición entre paréntesis y su cuerpo entre llaves:

```
if (hayPersonasEnEspera()) {
    llamarSiguientePersona();
}
```

Además, los if s pueden opcionalmente tener un else:

```
if (hayPersonasEnEspera()) {
    llamarSiguientePersona();
} else {
    esperarSiguientePersona();
}
```

Por último, podemos combinar varios if s para tomar decisiones ante múltiples condiciones:

```
if (hayPersonasEnEspera()) {
    llamarSiguientePersona();
} else if (elPuestoDebeSeguirAbierto()) {
    esperarSiguientePersona();
} else {
    cerrarPuesto();
}
```

Variables

A partir de la Lección 3: Variables y procedimientos

Las variables nos permiten *recordar* valores y se declaran mediante la palabra reservada let y se les da un valor inicial usando = :

```
let pesosEnMiBilletera = 100;
let diasQueFaltanParaElVerano = 10;
```

La mismas se asignan mediante = :

```
pesosE diasQu This document is available free of charge on StuDocu.com

Downloaded by daniel rene ovejero (ovejero1160@gmail.com)
```

En ocasiones las asignaremos usando el valor anterior:

```
pesosEnMiBilletera = pesosEnMiBilletera * 2;
diasQueFaltanParaElVerano = diasQueFaltanParaElVerano - 1;
```

La asignación anterior se puede compactar combinando el signo = y la operación:

```
pesosEnMiBilletera *= 2;
diasQueFaltanParaElVerano -= 1;
```

Repetición indexada

A partir de la Lección 7: Recorridos

Las listas pueden ser *recorridas*, visitando y haciendo algo con cada uno de sus elementos. Para ello contamos con la estructura de control for..of, que encierra su generador entre paréntesis ((y)) y su cuerpo entre llaves ({ y }):

```
Ф
let patrimoniosDeLaHumanidad = [
  {declarado: 1979, nombre: "Parque nacional Tikal", pais: "Gua
temala"},
  {declarado: 1983, nombre: "Santuario histórico de Machu Picch
u", pais: "Perú"}
  {declarado: 1986, nombre: "Parque nacional do Iguaçu", pais:
"Brasil"},
  {declarado: 1995, nombre: "Parque nacional de Rapa Nui", pais
: "Chile"},
  {declarado: 2003, nombre: "Quebrada de Humahuaca", pais: "Arg
entina"}
]
let cantidadPatrimoniosDeclaradosEnEsteSiglo = 0;
for (let patrimonio of patrimoniosDeLaHumanidad) {
  if (patrimonio.declarado >= 2000) {
    cantidadPatrimoniosDeclaradosEnEsteSiglo += 1;
  }
}
```

Biblioteca simplificada

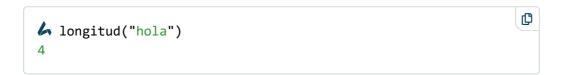
Dentro de Mumuki usamos una biblioteca de funciones inspirada en la Downloaded by daniel rene ovejero (ovejero1160@gmail.com)

sencilla y segura de usar. A continuación listamos las principales funciones que se pueden usar, indicando el equivalente *real* en JavaScript cuando corresponda.

longitud(unString)

A partir de la Lección 1: Funciones y tipos de datos Versión simplificada de length

Uso:



convertirEnMayuscula(unString)

A partir de la Lección 1: Funciones y tipos de datos Versión simplificada de toUpperCase

Convierte un unString en mayúsculas:

```
ConvertirEnMayuscula("hola")
"HOLA"
```

comienzaCon(unString, otroString)

A partir de la Lección 1: Funciones y tipos de datos Versión simplificada de startsWith

Dice si unString empieza con otroString:



Downloaded by daniel rene ovejero (ovejero1160@gmail.com)

imprimir(unString)

A partir de la Lección 3: Variables y procedimientos Versión simplificada de console.log

Imprime por pantalla unString:

```
imprimir("¡estoy imprimiendo!")
¡estoy imprimiendo!
```

tirarDado()

A partir de la Lección 3: Variables y procedimientos

Devuelve al azar un número entre 1 y 6:

```
Land tirarDado()

tirarDado()

tirarDado()

tirarDado()

tirarDado()
```

listasIguales(unaLista, otraLista)

A partir de la Lección 5: Listas

longitud(unaLista)

A partir de la Lección 5: Listas

• length de listas

Nos dice cuan largo es unaLista:

```
Longitud([true, false, false, true])

Downloaded by daniel rene ovejero (ovejero1160@gmail.com)
longitud([5, 6, 3])
```

agregar(unaLista, unElemento)

A partir de la Lección 5: Listas Versión simplificada de push

3

Inserta unElemento al final de unaLista. Este es un procedimiento que no devuelve nada pero modifica a unaLista:

```
Let cancionesFavoritas = ["La colina de la vida", "Zamba por
vos"]

// agrega el elemento "Seminare" a la lista cancionesFavoritas

agregar(cancionesFavoritas, "Seminare")

// ahora la lista tiene un elemento más:

cancionesFavoritas

["La colina de la vida", "Zamba por vos", "Seminare"]
```

remover(unaLista)

A partir de la Lección 5: Listas Versión simplificada de pop

Quita el último elemento de unaLista. Este es un procedimiento que no devuelve nada pero modifica a unaLista:

```
Let listaDeCompras = ["leche", "pan", "arroz", "aceite", "ye
rba"]
// removemos el último elemento
remove(listaDeCompras)
// la "yerba" ya no está en lista de compras
listaDeCompras
["leche", "pan", "arroz", "aceite"]
```

posicion(unaLista, unElemento)

A partir de la Lección 5: Listas Versión simplificada de index0f

Nos dice en qué posición se encuentra unElemento dentro de unaLista. Si el elemento no está en la lista, devuelve -1

O

```
posicion(premios, "dani")

posicion(premios, "juli")

posicion(premios, "feli")
-1
```

Bibliografía complementaria

- https://developer.mozilla.org/es/docs/Web/JavaScript
- https://es.javascript.info/

© 2015-2021 Mumuki
Información importante
Términos y Condiciones
Reglas del Espacio de Consultas







