



Seguir

Sé el primero de tus amigos
en seguir a Miríada X.



Follow @miriadax



Mi Página

Cursos

Universidades e instituciones

Conócenos

Soporte

Desarrollo de servicios en la nube con HTML5, Javascript y node.js

Inicio

Syllabus

Foro

Blog

Módulos

● **Modulo 0. Introducción al curso, al programa y al Sistema Operativo UNIX**

● **Modulo 1. Introducción a JavaScript de servidor y a node.js. Sentencias, Variables, Booleanos, Números, Strings y Funciones**

✓ Tema 0: Transparencias del módulo

✓ Tema 1. Introducción a Javascript. Tipos y valores

✓ Tema 1. Cuestionario opcional

✓ Tema 2. Programa, sentencia, variable y comentario

✓ Tema 2. Cuestionario opcional

✓ Tema 3. Expresiones con variables

✓ Tema 3. Cuestionario opcional

Modulo 1. Introducción a JavaScript de servidor y a node.js. Sentencias, Variables, Booleanos, Números, Strings y Funciones

Prueba realizada

Tu resultado en el test ha sido: 100%

Has superado el test.

Tus respuestas

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };  
}
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
return (x) ? function uno() { return 1; } : function cero() { return 0; } ;  
}
```

Cómo se evaluará las siguiente expresión: f_2 (7)()

☐ 0

☒ 1

✓ Tema 4. Introducción node.js

✓ Tema 5. Booleano, igualdad y otros operadores lógicos

✓ Tema 5. Cuestionario opcional

✓ Tema 6. Sentencia IF/ELSE

✓ Tema 7. Números

✓ Tema 7. Cuestionario obligatorio

✓ Tema 8. Strings e internacionalización (I18N)

✓ Tema 8. Cuestionario opcional

✓ Tema 9. Funciones

✓ Tema 9. Cuestionario obligatorio

✓ Tema 10. Funciones como objetos y cierres

➔ Tema 10. Cuestionario obligatorio

Ejercicio P2P Opcional

● **Modulo 2. Introducción a JavaScript de servidor y a node.js. Bucles, Clases predefinidas, Objetos, Propiedades y Métodos; Prototipos y Clases; Arrays; JSON; Funciones como Objetos y Cierres (Closures)**

● **Modulo 3. Modulos node.js; Expresiones Regulares; Eventos, Entorno de Ejecución y Concurrencia en node.js; Ficheros y Flujos**

● **Modulo 4. Introducción a HTTP y a los Servidores Web; Introducción a**

☐ undefined

☐ function cero()

☐ function uno()

☐ error_de_ejecución

 **Correct**

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
return (x) ? function uno() { return 1; } : function cero() { return 0; };  
}
```

Cómo se evaluará las siguiente expresión: f_2 (0)()

☒ 0

☐ 1

☐ undefined

☐ function cero()

☐ function uno()

☐ error_de_ejecución

 **Correct**

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
return (x) ? function uno() { return 1; } : function cero() { return 0; };  
}
```

express y al Middleware Static; Introducción a REST; Aplicaciones express.js y Composición de Middlewares; Formularios GET y POST; Parámetros Ocultos

- **Modulo 5. Gestión de versiones de proyectos con git y GITHUB; Proyecto, Espacio de Trabajo y Versiones (Commit); Arboles y Ramas de un proyecto; Repositorios Remoto y colaboración a través de GITHUB**
- **Modulo 6. Proyecto Quiz I: Patrón Modelo-Vista-Controlador (MVC); generación del proyecto con express-generator; Primera Página y Primera Pregunta; Despliegue en la nube (Heroku)**
- **Modulo 7. Proyecto Quiz II: La Base de Datos (DB), Tablas, sequelize.js y SQLite; Despliegue en Heroku utilizando Postgres; Presentación de Listas de Quizes y Autoload**
- **Modulo 8. Proyecto Quiz III: Gestión de Listas de Quizes, Creación, Edición y Borrado**
- **Modulo 9. Proyecto Quiz IV: Creación y Moderación de Comentarios a Quizes; Relaciones entre Tablas de la Base de Datos; Sesiones, Autenticación y Autorización; HTTP Seguro (HTTPS)**

}

Cómo se evaluará las siguiente expresión: f_1 (0)

- ☒ 0
- ☐ 1
- ☐ undefined
- ☐ function cero()
- ☐ function uno()
- ☐ error_de_ejecución



Correct

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };  
}
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
return (x) ? function uno() { return 1; } : function cero() { return 0; };  
}
```

Cómo se evaluará las siguiente expresión: f_1 (0)()

- ☒ 0
- ☐ 1
- ☐ undefined
- ☐ function cero()
- ☐ function uno()
- ☐ error_de_ejecución



Correct

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };  
}
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
  return (x) ? function uno() { return 1; } : function cero() { return 0; } ;  
}
```

Cómo se evaluará las siguiente expresión: f_1 (7)

- ☐ 0
- ☐ 1
- ☐ undefined
- ☐ function cero()
- ☒ function uno()
- ☐ error_de_ejecución



 **Correct**

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
  function cero () { return 0; };  
  function uno () { return 1; };  
}
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
  return (x) ? function uno() { return 1; } : function cero() { return 0; } ;  
}
```

Cómo se evaluará las siguiente expresión: f_2 ()()

- ☒ 0
- ☐ 1
- ☐ undefined
- ☐ function cero()
- ☐ function uno()
- ☐ error_de_ejecución



 **Correct**

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
return (x) ? function uno() { return 1; } : function cero() { return 0; };  
}
```

Cómo se evaluará las siguiente expresión: f_2 ()

- ☐ 0
- ☐ 1
- ☐ undefined
- ☒ function cero()
- ☐ function uno()
- ☐ error_de_ejecución



 **Correct**

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
return (x) ? function uno() { return 1; } : function cero() { return 0; };  
}
```

Cómo se evaluará las siguiente expresión: f_2 (7)

- ☐ 0
- ☐ 1

- ☐ undefined
- ☐ function cero()
- ☒ function uno()
- ☐ error_de_ejecución



 **Correct**

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
return (x) ? function uno() { return 1; } : function cero() { return 0; };  
}
```

Cómo se evaluará las siguiente expresión: f_1 (7)()

-
- ☐ 0
 - ☒ 1
 - ☐ undefined
 - ☐ function cero()
 - ☐ function uno()
 - ☐ error_de_ejecución



 **Correct**

Si tenemos las siguientes definiciones de funciones:

```
function f_1 (x) {  
function cero () { return 0; };  
function uno () { return 1; };
```

```
if (x) { return uno; } else { return cero; }  
};
```

```
function f_2 (x) {  
return (x) ? function uno() { return 1; } : function cero() { return 0; };  
}
```

}

Cómo se evaluará las siguiente expresión: f_1 ()

- ☐ 0
- ☐ 1
- ☐ undefined
- ☒ function cero()
- ☐ function uno()
- ☐ error_de_ejecución



 **Correct**

Enhorabuena. Terminaste este módulo.

 **anterior**

Siguiente 

Telefonica

Telefónica Educación Digital

 Santander

uni>ersia

2012-2015 Miríada X

| [Aviso legal](#)

| [Política de cookies](#)

| [Política de privacidad](#)