

# JavaScript

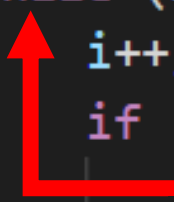
# Ejercicio

Escriba un programa que genere un número aleatorio y le pida al usuario que adivine cuál es el número. Si la suposición del usuario es mayor que el número aleatorio, el programa debería mostrar "Demasiado alto, intente nuevamente". Si la suposición del usuario es inferior al número aleatorio, el programa debería mostrar "Demasiado bajo, inténtelo de nuevo". El programa debe usar un ciclo que se repite hasta que el usuario adivine correctamente el número aleatorio.

# Sentencia continue

Termina la ejecución de las sentencias en la iteración actual del ciclo o etiqueta. Continúa la ejecución del ciclo con la próxima iteración ignorando sentencias posteriores en el interior del ciclo.

```
<script>
  var i = 0;
  var n = 0;
  while (i < 5) {
    i++;
    if (i == 3) {
      continue;
    }
    n += i;
    console.log(n);
  }
</script>
```



El siguiente ejemplo muestra un ciclo while que tiene una sentencia continue que se ejecuta cuando el valor de i es 3. Por lo tanto, n toma los valores 1, 3, 7 y 12.

Al hacer “continue” la siguiente instrucción será volver a preguntar por el valor de verdad para ejecutar el ciclo nuevamente desde el inicio. Si al ejecutar continue habían instrucciones pendientes en el interior del ciclo estas son ignoradas.

# Sentencia break

Al ejecutar la sentencia “break” termina/rompe el ciclo directamente relacionado, sin importar que la condición sea verdadera. La siguiente instrucción será fuera del ciclo.

```
<script>
  var i = 0;

  while (i < 10) {
    if (i == 3) {
      break;
    }
    i++;
  }

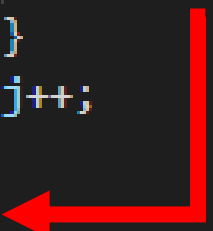
  console.log(i);
</script>
```

Al hacer “break” la siguiente instrucción continua después del cierre de llave del ciclo. Si al ejecutar break habían instrucciones pendientes en el interior del ciclo estas son ignoradas.


# Observación

Tanto la sentencia break y continue al ser ejecutadas van relacionadas al ciclo directo que las contiene.

```
var i = 0,  
    j = 0;  
while (i < 3) {  
    while (j < 3) {  
        if (j == 2) {  
            break;  
        }  
        j++;  
    }  
    i++;  
}
```



```
var i = 0,  
    j = 0;  
while (i < 3) {  
    while (j < 3) {  
        if (j == 2) {  
            continue;  
        }  
        j++;  
    }  
    i++;  
}
```



# Ciclo do/while

La sentencia do/while crea un bucle que ejecuta una sentencia especificada, hasta que la condición de comprobación se evalúa como falsa. La condición se evalúa después de ejecutar la sentencia, dando como resultado que la sentencia especificada se ejecute al menos una vez.

```
do {  
    /*Código aquí*/  
} while (/*Condición*/);
```

Observe que al termino del  
paréntesis debe ir un “;”



# Ejercicio

Imprima un listado del 0 al 10 utilizando do/while.

```
do {  
    /*Código aquí*/  
} while (/*Condición*/);
```

Observe que al termino del  
paréntesis debe ir un “;”



Recuerde que do/while  
primero ejecuta las  
sentencias y después  
pregunta si debe repetir.

# Ejercicio

Escribir un programa que pida números enteros indefinidamente hasta que se ingrese un 0 (cero) para terminar la ejecución.

Imprima la suma de todos los números ingresados.

Utilice un ciclo do/while para resolver.

**Desarrolle una variante de este ejercicio utilizando la sentencia `break`; para terminar la ejecución.**

```
do {  
    /*Código aquí*/  
} while (/*Condición*/);
```

Recuerde que do/while primero ejecuta las sentencias y después pregunta si debe repetir.

Observe que al termino del paréntesis debe ir un “;”



# Ejercicio

Los padres de una niña le prometieron darle 5 dólares cuando cumpliera 10 años de edad y duplicar el regalo en cada cumpleaños subsiguiente hasta que el regalo excediera los 1000 dólares. Escriba un programa para determinar qué edad tendrá la niña cuando se le dé el último regalo que exceda los 1000 dolares. Indique la cantidad total de dinero regalado.

Utilice un ciclo do/while para resolver.

# Ejercicio

Diseñe un programa que valide correctamente contraseñas.

Utilizando prompt pida 2 veces una contraseña, si una difiere de la otra se deberá volver a pedir ambas, hasta que sean iguales.

Utilice la función alert para indicar al usuario cuando se equivoque.

# Ejercicio

Escriba un programa que pida números hasta que se ingrese un 0 (cero).

Imprima el número mayor y menor ingresado.