Fundamentos de programación con JavaScript

Aprende los fundamentos de programación con uno de los lenguajes más demandados de la industria

John Serrano

Tabla de contenido

Fundamentos de programación con JavaScript

Aprende los fundamentos de programación con uno de los lenguajes de programación más demandados de la industria. Aprende desde cero hasta dominar JavaScript.

Este libro está a la venta en **leanpub.com**.

Publicado por **johnserrano.co**

Sobre el autor

John Serrano (Colombia, 1992) Desarrollador Web Full-Stack. Amante de JavaScript y entusiasta de las tecnologías web: Node.js, Docker, Firebase, React, etc. Con varios años de experiencia tanto en empresas privadas y como FreeLancer.

Tecnólogo en análisis y desarrollo de sistemas de la información, muy pronto un egresado de la Universidad Santo Tomás como **ingeniero en informática**. En constante proceso de aprendizaje sobre todas las nuevas tecnologías, es un amante del autoaprendizaje a través de Internet. Puedes encontrar sus artículos y tutoriales en su blog johnserrano.co.

Sobre este libro

Cuando comenzamos en el mundo de la programación no tenemos claro que es la lógica de programación simplemente porque no se comprendió bien en los estudios de la Universidad o porque comenzamos como una afición a programar, en este libro quiero llevarlo desde lo más básico de la programación hasta un nivel medio avanzado, aprenderás que es un algoritmo, como resolver problemas con algoritmos, adquirir la lógica de la programación hasta manejar un lenguaje de programación.

Porque JavaScript, bueno bien porque es uno de los lenguajes más demandado en la industria hoy en día, a pesar de que se creó para ser usado en los navegadores hoy en día también puede ser usado en el servidor con Node.js. Gracias a JavaScript se pueden crear Single Page Applications sin necesidad de recargar la página para navegar en el sitio web o aplicación. También se pueden crear aplicaciones basadas en componentes gracias a React, Polymer, aplicaciones híbridas con herramientas como Ionic, React Native para crear aplicaciones nativas para iOS y Android.

Lo primero que tenemos que aprender muy bien es aprender a resolver problemas de una forma lógica sin necesidad de aprender un lenguaje de programación o usar algunas herramientas para desarrollar como lo pueden ser librerías, frameworks, etc. Por esta razón comenzaremos resolviendo problemas planteando algoritmos y luego aprenderemos las bases de JavaScript puro o también conocido como **Vanilla JS**. Creo que la mejor forma de aprender a resolver un problema y adquirir ese pensamiento lógico son los diagramas de flujo, pero que es un algoritmo. Un algoritmo es un conjunto de instrucciones o reglas definidas y no-ambiguas, ordenadas y finitas que permite, típicamente, solucionar un problema.

Espero que disfrutes del ebook y te sirva para tu carrera profesional. Cualquier cosa que necesites me encuentras en mi blog:

- **John Serrano Blog**

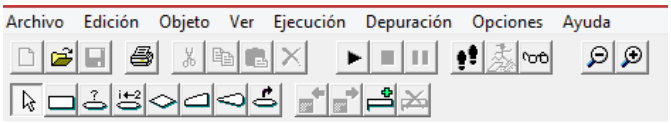
Y en las redes sociales:

- **Sígueme en Twitter @jandrey15**
- **Sígueme en Facebook**
- **Estoy en Instagram como @jandrey15**

Sin más, te dejo con el ebook.

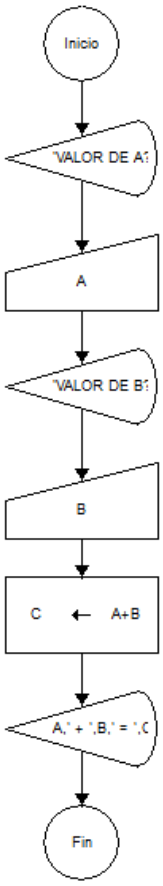
Diagramas de flujo

Para practicar con los diagramas de flujo pueden instalar un programa llamado **DFD** o usar su versión portable yo voy a usar ese para mostrar los ejemplos veamos cómo se ve el programa cuando lo abrimos por primera vez.



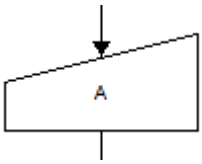
Como podemos ver en la parte superior tenemos unos símbolos y en el centro podemos ver dos círculos un que dice inicio y fin. Porque todo proceso o algoritmo tiene un inicio y un fin, veamos un ejemplo básico de cómo usar esta herramienta.

Nota: *Con el símbolo de play ejecutamos nuestro diagrama de flujo.*

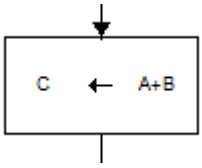


Un ejercicio básico para comprender como funciona el programa y que son todo este símbolo es una suma básica. Cada uno de estos símbolos es el flujo algoritmo que nosotros le damos para resolver un problema veamos que significa da uno de ellos.

Este símbolo nos indica que es de tipo salida el cual si hacemos doble click sobre el podemos agregar texto entre comillas.



Este símbolo es de tipo lectura y podemos indicarle una variable la cual va recibir el valor que pasemos nosotros manualmente cuando se ejecute el pequeño programa, una variable no es más que la representación de un valor con alguna letra o texto. Si hacemos doble click sobre el símbolo podemos crear esa variable.



Este símbolo es de tipo asignación, el cual nos permite hacer la suma de las variables que se crearon en este caso fue A y B y lo asignamos a una nueva variable llamada C donde se va guardar el resultado de sumar A y B. Prácticamente toda la lógica de nuestro pequeño programa está en esta operación porque el problema es sumar dos número y lo resolvemos con este símbolo.

Por último, tenemos otro símbolo de tipo salida donde colocamos texto con las variables ya creadas donde nos muestra el resultado de la suma. Como vemos no

es tan complicado usar el programa aún faltan algunos símbolos por ver, pero para comenzar esta bien con estos ya que son los básicos, a medida que avancemos iremos usando los demás símbolos.

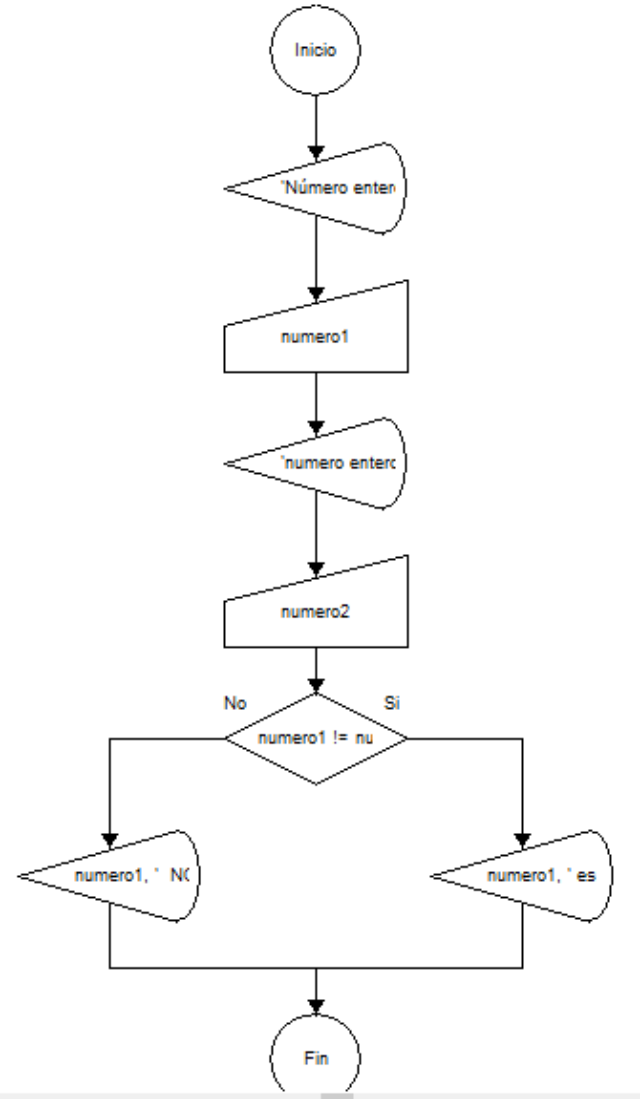
Ahora veamos un problema y cómo podemos resolverlo con nuestros diagramas de flujo.

Condicionales

Problema #1

IGUALES O DIFERENTES: Dados dos números enteros, calcular si son iguales o diferentes.

Leamos por un momento este problema piensa en la forma de resolverlo con los diagramas de flujo que ya vimos, dice dados dos números enteros, es decir vamos a necesitar el símbolo de lectura para crear dos variables y debemos usar un nuevo símbolo de tipo condición para resolver el problema.



Como podemos ver pedimos los valores o números, pero tenemos algo nuevo un símbolo de tipo condicional donde podemos usar un operador diferencial el cual lo usamos con estos símbolos != donde si numero1 es diferente de numero2 si eso es verdad pasa por el sí de lo contrario pasara por el no, lo único que hace un condicional es validar si una variable cumple con una condición y solo tiene 2 posibles resultados un sí o un no.

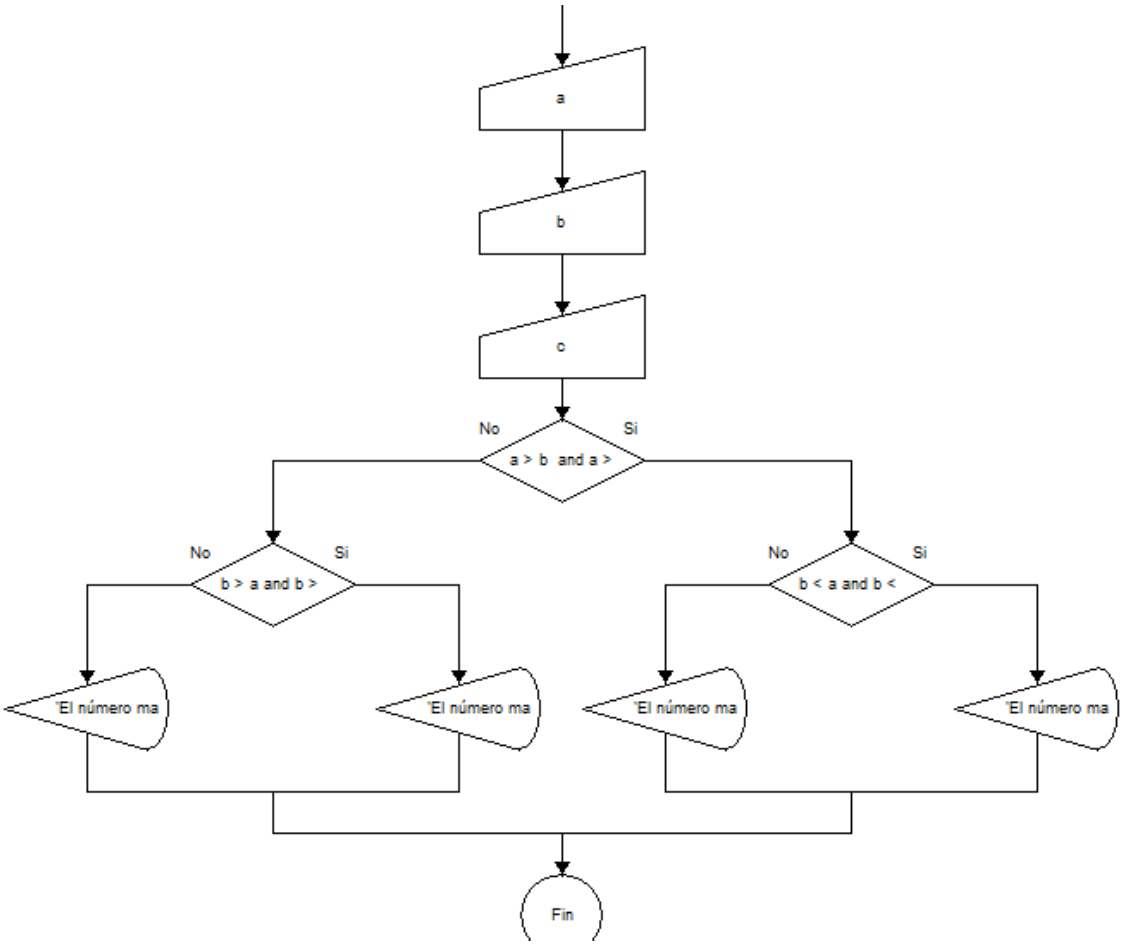
Nota: *más adelante hablaremos sobre los operados de comparación.*

Vamos a ver otros problemas más y tratar de comprender como resolverlo, empleando algoritmos en un diagrama de flujo y llegar a la solución del problema.

Problema #2

MAYOR MEDIO Y MENOR: Dados 3 números diferentes, calcular el mayor, el medio y el menor.

Nuevamente comencemos a pensar cómo podríamos resolver este problema lo primero es que nos dicen dados 3 números diferentes ya con esto sabemos que necesitamos símbolos de lectura, calcular el mayor, el medio y el menor, ya con esto sí o sí debemos usar símbolo de condicional veamos la solución.

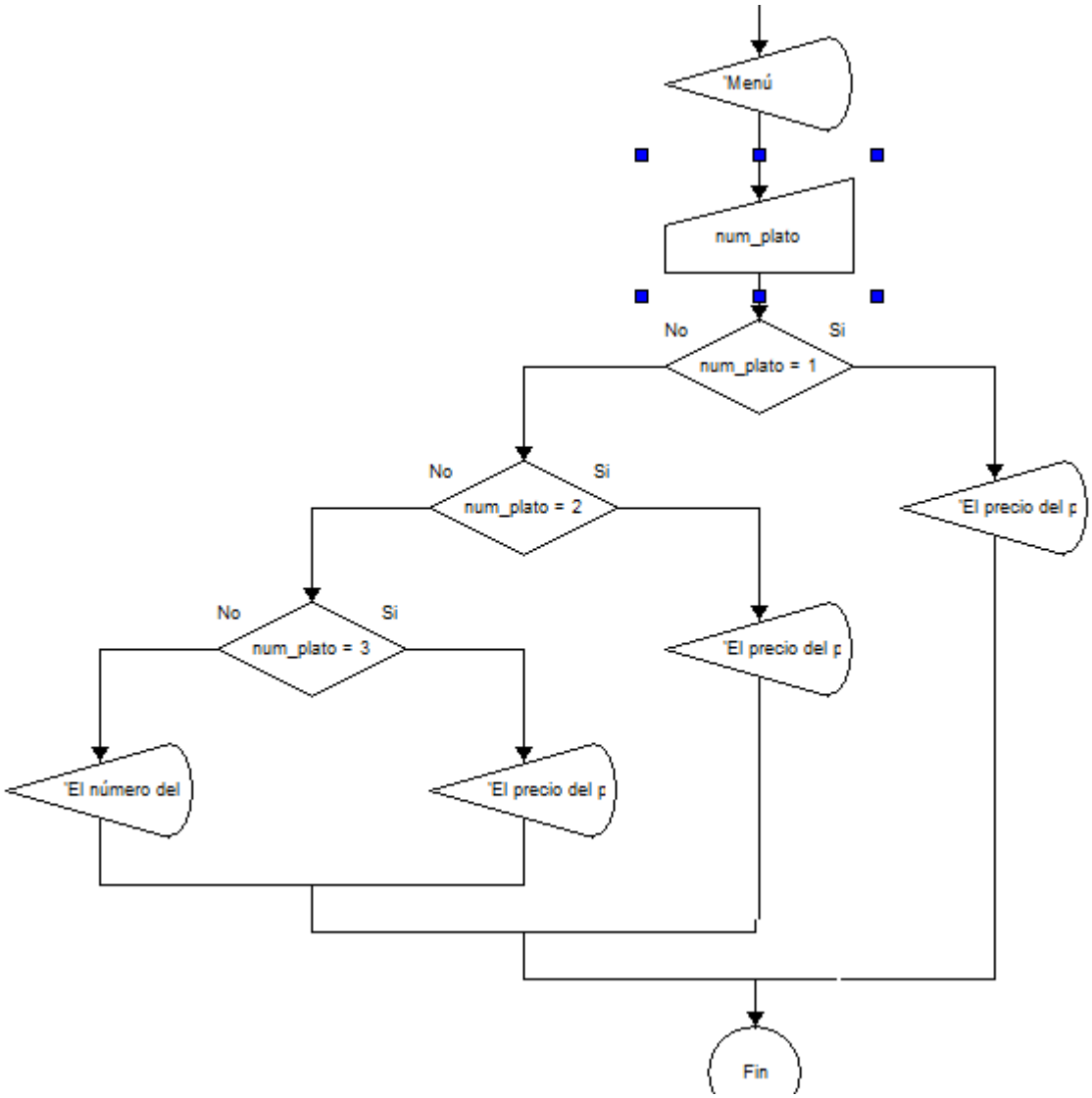


Como podemos ver este problema es un poco más complejo que el anterior porque tenemos que usar varios condicionales y usar operadores lógicos, no se preocupe más adelante veremos con mayor entendimiento que son los **operadores lógicos**. En el primer condicional preguntamos que si ***a > b and a > c*** con esto ya sabemos que el número mayor es el valor que está en la ***variable a*** sabiendo esto ahora preguntamos que si ***b < a and b < c*** si la respuesta es sí del condicional ya sabemos que el número menor es el valor que está en la variable ***b*** como resultado solo queda descartar que el medio en este caso sería el valor que está en la variable ***c***.

Esto mismo que se hizo por el condicional derecho hacemos por el condicional izquierdo, pero ya sabemos que ***a*** puede ser menor o medio.

Problema #3

Presentar un menú de 5 platos de un restaurante, el usuario al seleccionar un plato le debe salir el precio.



Si pensamos como plantear este problema en un algoritmo usando los diagramas de flujo podemos ver que nos dice un menú de 5 platos, lo primero es mostrar los platos y un indicador de cada plato asignamos ese valor en una variable ***num_plato*** y luego con los condicionales validamos si el número del plato seleccionado corresponde con los platos de nuestro menú hacemos esto por cada

plato y mostramos un mensaje con el precio correspondiente, en el caso de que no existe ese número le indicamos simplemente que ese plato no existe.

A continuación, dejare unos problemas de tipo condicional para que practiques.

Más problemas de tipo condicionales:

1. IGUALES O DIFERENTES: Dados dos números enteros, calcular si son iguales o diferentes.

2. MENOR DE DOS NUMEROS: Dados dos números enteros, calcular el menor de ellos.

3. MENOR DE TRES NUMEROS: Dados 3 números diferentes, calcular el menor de ellos.

4. MEDIO DE TRES NUMEROS: Dados 3 números diferentes, calcular el de la mitad (no es el mayor ni el menor)

5. MAYOR MEDIO Y MENOR: Dados 3 números diferentes, calcular el mayor, el medio y el menor.

6. PAR O IMPAR: Leer un número entero y calcular si es par o impar

7. POSITIVO O NEGATIVO: Leer un número y calcular si es positivo o negativo

8. MULTIPLO DE X: Dados 2 números calcular si el primero es múltiplo del segundo.

9. ECUACION CUADRATICA: Dados los coeficientes y el termino independiente de la ecuación cuadrática (ax^2+bx+c=0), calcular el imprimir el valor de las raíces reales. X1 y X2.
Si el valor de a=0 debe mostrar un mensaje indicando "División por cero". Si B^2 - 4AC<0 indicar que la raíz da negativo.

MENU DE OPCIONES:

10. Presentar un menú de 5 platos de un restaurante, el usuario al seleccionar un plato le debe salir el precio.

11. Presentar un menú de 5 destinos (ciudades) diferentes, el usuario al seleccionar el destino conocerá la distancia en kilómetros y el precio del pasaje por avión.

12. Dados el peso y estatura de una persona, calcular el IMC (índice de masa corporal). De acuerdo con el resultado del IMC mostrar un mensaje que diga si está: DELGADO, NORMAL, SOBREPESO u OBESO.

Ciclos

Siguiendo con los problemas vamos a ver un tipo de problemas que se resuelven con ciclos o diagramación repetitivos en este caso siguiendo con los diagramas de flujo, antes de ir a los problemas definamos que es un ciclo.

Un bucle o ciclo, en programación, es una secuencia que ejecuta repetidas veces un trozo de código, hasta que la condición asignada a dicho bucle deja de cumplirse. Los tres bucles más utilizados en programación son el bucle while, el bucle for y el bucle do-while. Fuente: **Wikipedia**

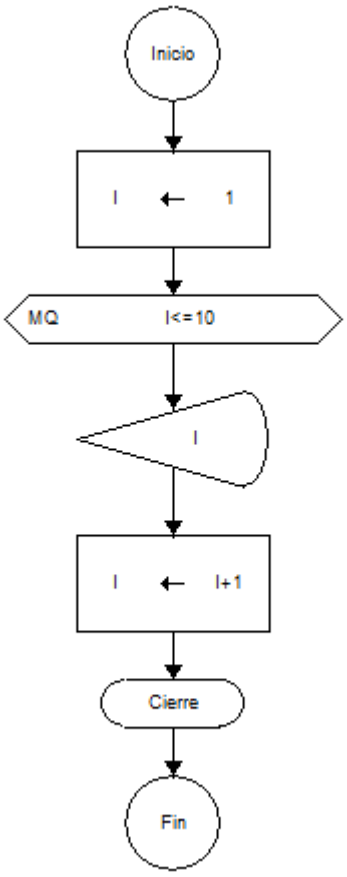
Teniendo claro que es un ciclo veamos cómo resolver problemas implementando cualquiera de estos tipos de bucles en los diagramas de flujo.

Problema #1

Imprimir números del 1 al 10.

Para resolver este problema debemos usar uno de los 2 nuevos símbolos que nos provee este programa el bucle ***mientras que*** y el buque ***para*** o ***for***.

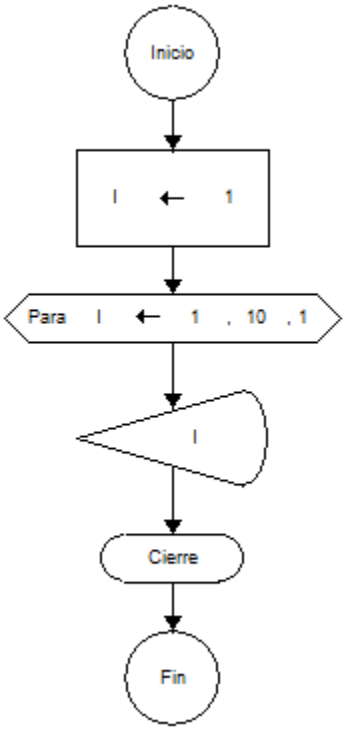
Como podemos ver comenzamos con símbolo de tipo asignación donde ***I*** es igual a ***1*** lo nuevo es el ciclo mientras que donde le decimos que si ***I <= 10*** quiero que pases y me imprimas ***I*** y seguido de eso uso otro símbolo de tipo asignación para aumentar a ***I*** en ***1*** con una suma donde ***I*** es igual a ***I + 1***, mientras se cumpla la condición de ***I <= 10*** va pasar y mostrarme cuánto vale ***I***.



Nota: *Los ciclos tiene un inicio y un fin este caso terminan con un símbolo llamado cierre.*

Podemos hacer el mismo ejemplo con el ciclo ***para*** veamos cómo se ve.

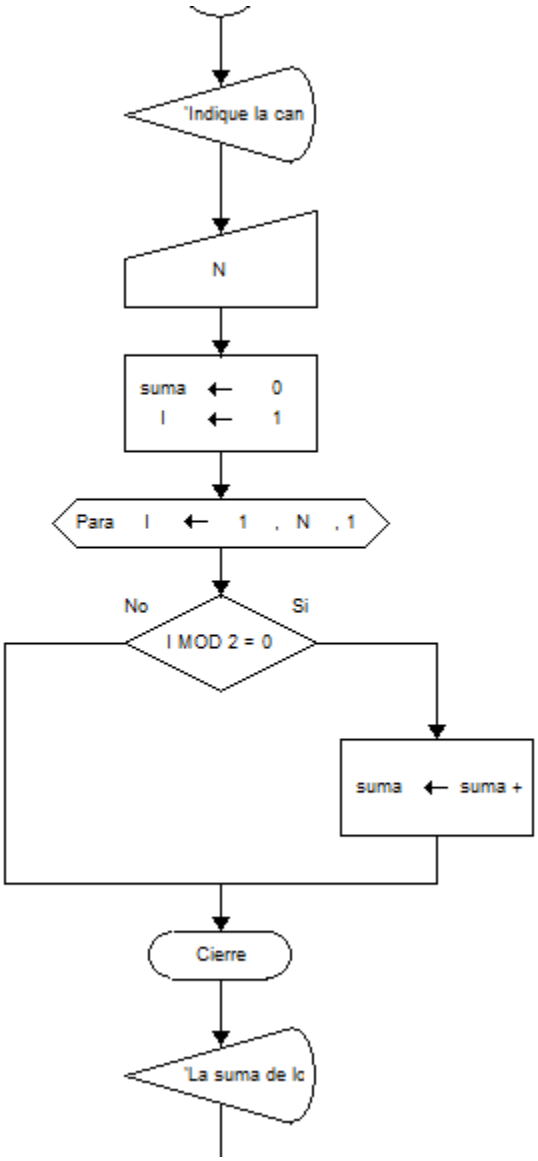
Con el ciclo para es un poco más sencillo le indicamos la variable ***I*** luego que comienza en ***1*** y va hasta ***10*** y que aumente ***I*** en ***1*** cada vez que haga un ciclo y con el símbolo de salida muestro el valor de ***I*** y listo eso es todo ejecutamos.



Problema #2

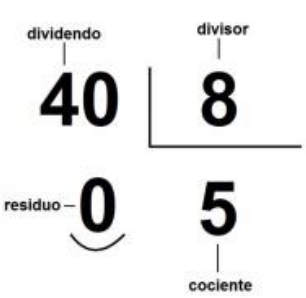
Leer N valores y calcular e imprimir la Sumatoria de los números pares.

Si leemos muy bien el problema nos dice que ***leer N*** valores lo cual no sabemos cuántos valores vamos a leer, entonces lo que hay que hacer es preguntar cuántos valores quieres leer entrar en un ciclo con esa cantidad de valores y sumas los números pares, veamos cómo queda la solución al problema en diagrama de flujos.



Iniciamos con un símbolo de salida mostrando un mensaje el cual dice que indique la cantidad de número que quiere sumar, tenemos un símbolo de lectura donde recogemos esa cantidad de número en la variable ***N***, asignamos dos variables iniciales la cual es ***suma*** e ***I*** que comienza ***suma*** en ***0*** e ***I*** en ***1***, entramos al ciclo y dentro del ciclo tenemos un condicional con una expresión nueva la cual es ***MOD*** y lo que hace ***MOD*** es obtener el residuo de una división en este caso la división de ***I / 2*** y preguntamos que si el residuo de esa operación es 0 entonces el número es par y usamos otro símbolo de asignación donde acumulamos o sumamos la variable ***suma + I*** y al final mostramos la suma de todos los números pares.

Nota: En matemáticas, un número par es un número entero que es divisible entre dos. También vamos a encontrar en diferentes lenguajes de programación que MOD es igual a %, es decir en vez de MOD usamos %.



Un ejemplo de una división con residuo.

Voy a dejar una lista de problema tipo ciclo donde pueden usar el ciclo ***mientras-que*** o el ciclo ***par*** y como vimos puede que tengan que usar condicionales.

Más problemas de tipo ciclos:

1. Imprimir números del 1 al 10
2. Sumatoria de 10 números
3. Dados base y exponente, calcular e imprimir la potencia.
4. Factorial de N
5. Imprimir la Serie de fibonacci
6. Leer N valores y calcular e imprimir la Sumatoria de los números pares
7. Leer N valores y calcular e imprimir el Promedio de estos.
8. Leer N valores y calcular e imprimir el Promedio de los números impares
9. Leer N números y calcular e imprimir el Mayor de ellos.
10. Leer un Número y determinar si es PRIMO o No.
11. Leer N números y calcular la Sumatoria de los múltiplos de 3
12. Leer N valores y calcular el Promedio de los múltiplos de 5
13. Dados calificación y créditos de las materias cursadas por un estudiante universitario calcular el promedio ponderado del semestre.
14. Dados nombre del artículo, precio y cantidad N productos, calcular e imprimir: Valor parcial por artículo, Subtotal, IVA y Total de la venta.
15. Dados nombre, edad y sexo de N personas, calcular e imprimir: total hombres, promedio edad mujeres, nombre del mayor y edad.

Mi primer hola mundo con JavaScript

Lo primero que vamos a hacer es hacer un hola mundo, pero antes de hacer eso definamos que es JavaScript de donde nació este lenguaje de programación, algunas características y algo de su historia.

¿Qué es JavaScript?

JavaScript (JS) es un lenguaje de programación ligero e interpretado, orientado a objetos con funciones de primera clase. Aunque es más conocido como el lenguaje de scripting para páginas web, muchos entornos no relacionados con el navegador también lo usan, tales como node.js, Apache CouchDB y Adobe Acrobat. Es un lenguaje script multiparadigma, basado en prototipos, dinámico, soporta estilos orientados a objetos, imperativos y declarativos.

Desarrollado por Netscape que se utiliza en millones de páginas web y aplicaciones de servidor en todo el mundo. JavaScript de Netscape es un superconjunto del lenguaje de scripts estándar de la edición de ECMA-262 3 (ECMAScript) que presenta sólo leves diferencias respecto a la norma publicada.

El estándar de JavaScript es ECMAScript. Desde el 2012, todos los navegadores modernos soportan completamente ECMAScript 5.1. Los navegadores más antiguos soportan por lo menos ECMAScript 3. El 17 de Julio de 2015, ECMA International publicó la sexta versión de ECMAScript, la cual es oficialmente llamada ECMAScript 2015, y fue inicialmente nombrada ECMAScript 6 o ES6. Desde entonces, los estándares ECMAScript están en ciclos de lanzamiento anuales. Esta documentación se refiere a la última versión del borrador, que actualmente es ECMAScript 2019.