Introducción a lógica de programación

DEV.F::
DESARROLLAMOS(PERSONAS);

Lógica

- Método o razonamiento en el que las ideas se manifiestan o desarrollan de forma coherente y sin que haya contradicciones entre ellas.

¿Dónde se aplica la lógica en la programación?

- Al analizar problemas y plantear problemas mediante software.
- Al resolver bugs de código.
- Negociar con clientes y usuarios.
- Diagnosticar problemas.
- Brindar soluciones.
- Soft Skills transversal (útil en cualquier área).

¿Y por qué tengo que saber lógica?

el programador cuenta con un conjunto de herramientas y lenguajes para construir la solución

PROGRAMADOR

HERRAMIENTAS Y LENGUAJES

PROBLEMA

análisis del probema

diseño de la solución

construcción de la solución

proceso

SOLUCIÓN

¿Cómo entender un problema?

DEV.F::
DESARROLLAMOS(PERSONAS);

Fases del análisis de un problema

1. **Definición:** Conocer el problema.
2. **Análisis:** Identificar lo que se necesita (entradas, proceso, salidas).
3. **Diseño de un algoritmo:** Mediante lenguaje natural, diagrama de flujo, pseudocódigo.
4. **Código:** Transformación del algoritmo en código.
5. **Ejecución y validación:** Realizar pruebas de escritorio del problema.
6. **Pruebas de aceptación:** La persona que planteó el problema valida que la solución cumpla ante los distintos casos solicitados.

Características de un algoritmo

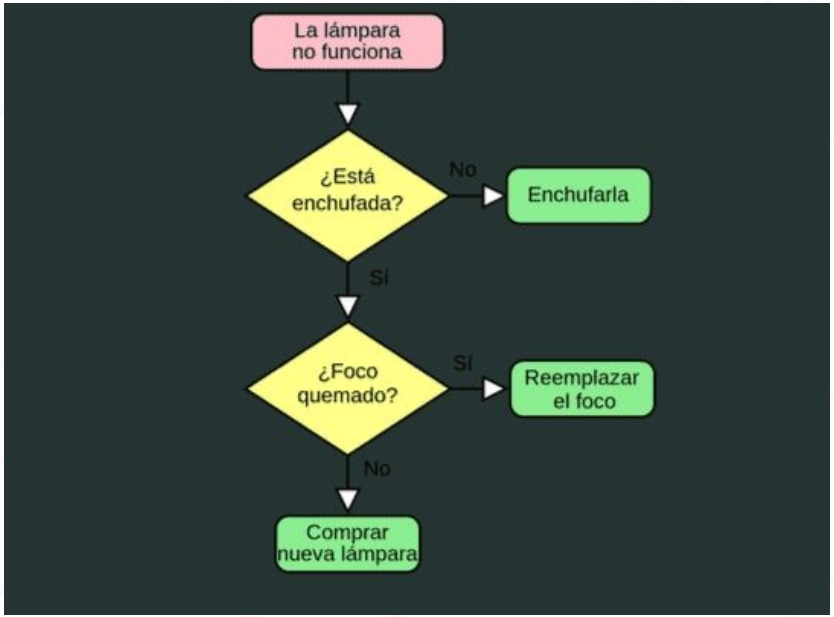
- **Exactitud:** Tiene que indicar un orden claro de la ejecución de cada paso, estos no pueden ser ambiguos.
- **Definido:** Si se realiza la ejecución de un mismo algoritmo en distintas instancias utilizando la misma entrada, debe resultar en la misma salida.
- **Completo:** En la solución se deben considerar todas las posibilidades del problema.
- **Finito:** Necesariamente un algoritmo debe tener un número finito de pasos.
- **Instrucciones entendibles:** Las instrucciones que lo describen deben ser claras y legibles.
- **General:** Debe poder abarcar problemas de un mismo tema soportando variantes del mismo.

Algoritmos

DEV.F::
DESARROLLAMOS(PERSONAS);

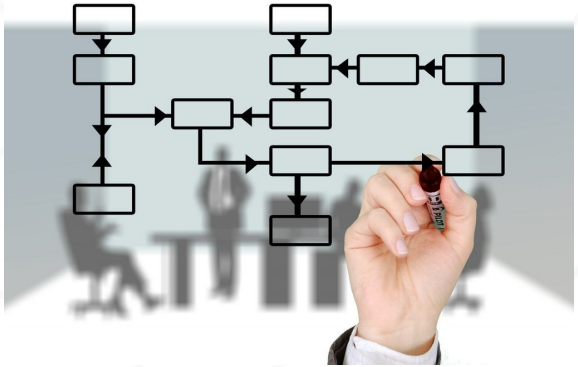
Algoritmo

- Un algoritmo es un conjunto de acciones que especifican la secuencia de operaciones realizadas, en orden, para resolver un problema.



Partes de un algoritmo

- **Entrada:** Se trata del conjunto de datos que el algoritmo necesita como insumo para procesar.
- **Proceso:** Son los pasos necesarios aplicados por el algoritmo a la entrada recibida para poder llegar a una salida o resolución del problema.
- **Salida:** Es el resultado producido por el algoritmo a partir del procesamiento de la entrada una vez terminada la ejecución del proceso.



Partes de un algoritmo

Entrada: Son los d**atos que se le dan al algoritmo**

```
a = 1;
b = 2;
```

Proceso: Son las **operaciones que se hacen con los datos**

```
Suma = a + b;
```

Salida: Es el **resultado final** que se obtiene de las operaciones,

en este caso será 3

```
document.write(Suma)
console.log(Suma)
```

Tipos de salidas algoritmo

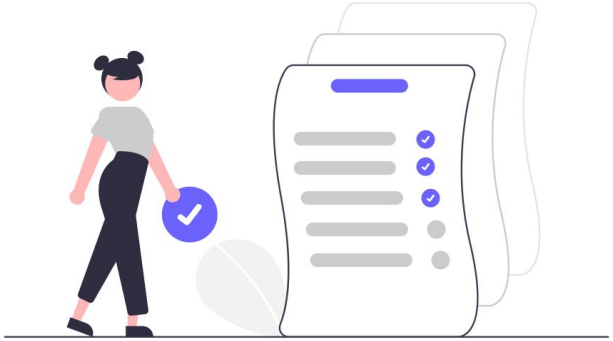
Todos los algoritmos tiene un fin, pero el resultado final de ese algoritmo puede ser de tres tipos:

- Algo que recibimos de retorno.
- Algo que se muestra en pantalla.
- O simplemente acción.

Representaciones de los algoritmos

DEV.F::
DESARROLLAMOS(PERSONAS);

Representaciones de un algoritmo



- Lenguaje natural.
- Diagrama de flujo.
- Pseudocódigo.
- Código.
- Prueba de escritorio.

Lenguaje natural

DEV.F::
DESARROLLAMOS(PERSONAS);

Lenguaje Natural

Es la lengua que usan los individuos para interactuar a través de alguna forma de comunicación sea escrita, oral o no verbal.

Ejemplo:

“Necesito un programa en JS que sume 2 numeros”.

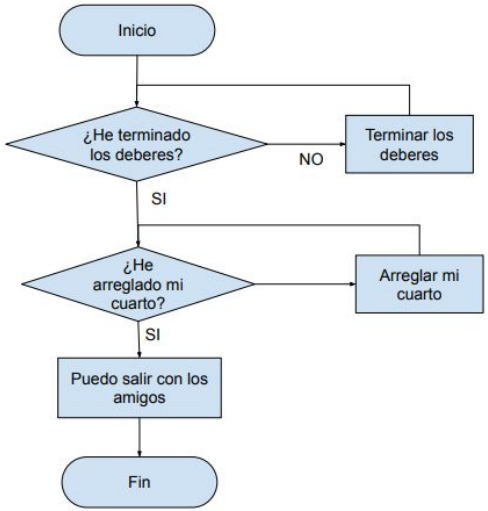
Diagramas de Flujo

DEV.F::
DESARROLLAMOS(PERSONAS);

¿Qué es un diagrama de flujo?

Un diagrama de flujo es un esquema que describe un proceso, sistema o algoritmo.

Se usan ampliamente para documentar, planificar, mejorar y comunicar procesos complejos en una representación clara y fáciles de comprender.



Simbología base

Docs

Conector

Flecha

Más simbología

Terminal/Terminador

Proceso

Documento

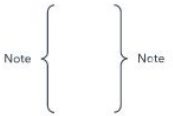
Decisión

Datos o entrada/salida

Datos almacenados

Flecha de flujo

Comentario o anotación



Proceso predefinido

Referencia/conector dentro de la página

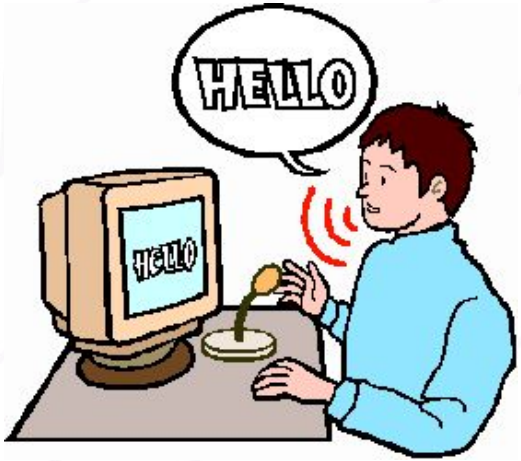
Pseudocódigo

DEV.F:
DESARROLLAMOS(PERSONAS);

¿Qué es un pseudocódigo?

El **pseudocódigo** es una forma de expresar los distintos pasos que va a realizar un programa, de la forma más parecida a un lenguaje de programación.

Su principal función es la de representar por pasos la solución a un problema combinando palabras entendibles por las personas que usualmente se usan en programación.



DEV.F

Convenciones de pseudocódigo

- “INICIO”
- “Leer”.
- “Si... entonces...”
- “Si no ... entonces...”
- “Mientras...”
- “Si y sólo si --- entonces”
- “Imprimir”
- “FIN”

Ejemplo de pseudocódigo

INICIO

Solicita Tipo Figura

Guarda tipo en var1

Solicita Magnitud a calcular

Guarda magnitud en var2

Solicita dimensiones (L, l, a, b... etc...)

SI Área y Cuadrado ENTONCES formula = L * L

// Si y solo si magnitud es Área y la figura Cuadrado la formula = L * L

SI Perimetro y Cuadrado ENTONCES formula = 4 * L

...

Guardar resultado de la formula

Imprime resultado

FIN

Código

DEV.F:
DESARROLLAMOS(PERSONAS);

Código

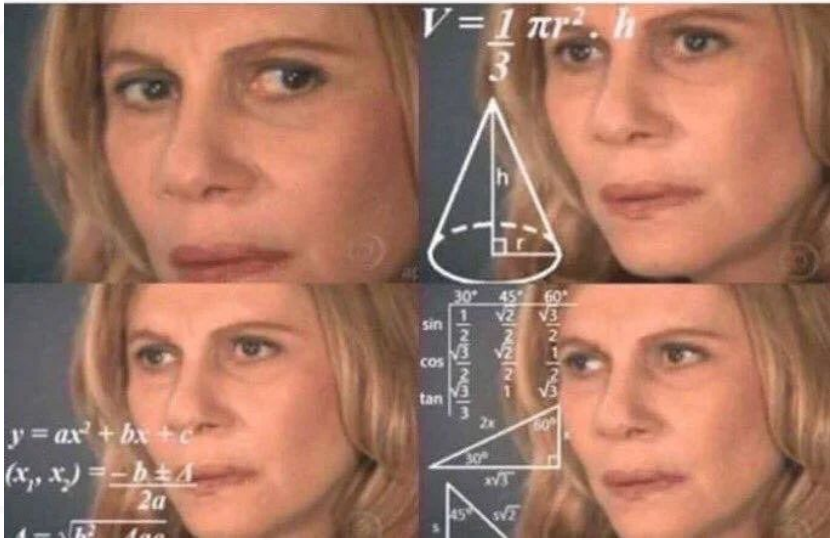
`await`	`break`	`case`	`catch`	`class`
`const`	`continue`	`debugger`	`default`	`delete`
`do`	`else`	`enum`	`export`	`extends`
`false`	`finally`	`for`	`function`	`if`
`implements`	`import`	`in`	`instanceof`	`interface`
`let`	`new`	`null`	`package`	`private`
`protected`	`public`	`return`	`super`	`switch`
`static`	`this`	`throw`	`try`	`true`
`typeof`	`var`	`void`	`while`	`with`
`yield`				

Repaso JS

DEV.F:
DESARROLLAMOS(PERSONAS);

¿Qué conocimientos previos necesito?

1. Tipos de datos (simples y complejos).
2. Operadores (asignación, aritméticos, comparación y lógicos).
3. Sentencias condicionales (if, else, switch).
4. Ciclos (for y while).
5. Objetos.
6. Arrays.
7. Funciones o métodos.



Prácticas

1. Simbología de diagrama de flujo.
2. Keywords de JS.
3. Diferencia entre condición y ciclo.
4. De la frase “Este es un mensaje secreto para mi crush” elimina todas las vocales e imprime la frase resultante en pantalla.
5. Dado un arreglo de números, realizar la suma de todos ellos.
6. Crear un objeto estudiante con las siguientes características:
    a. Al menos 6 propiedades.
    b. 1 propiedad de ser un conjunto de 7 calificaciones.
7. Crear un arreglo de 3 estudiantes.
8. Crear un html con 1 botones que al pulsarlo pinte la lista de estudiantes (mostrar nombre y edad).
9. Agregar un botón para borrar en c/estudiante.