



# Análisis y desarrollo de software.

## ID: 2669959



[www.sena.edu.co](http://www.sena.edu.co)

@SENAComunica

# Instructor

---

Diego Fernando Calderón Silva  
Correo: dfcalderon@sena.edu.co

# Reflexión inicial

---

- **¿QUÉ ES SOFTWARE?**

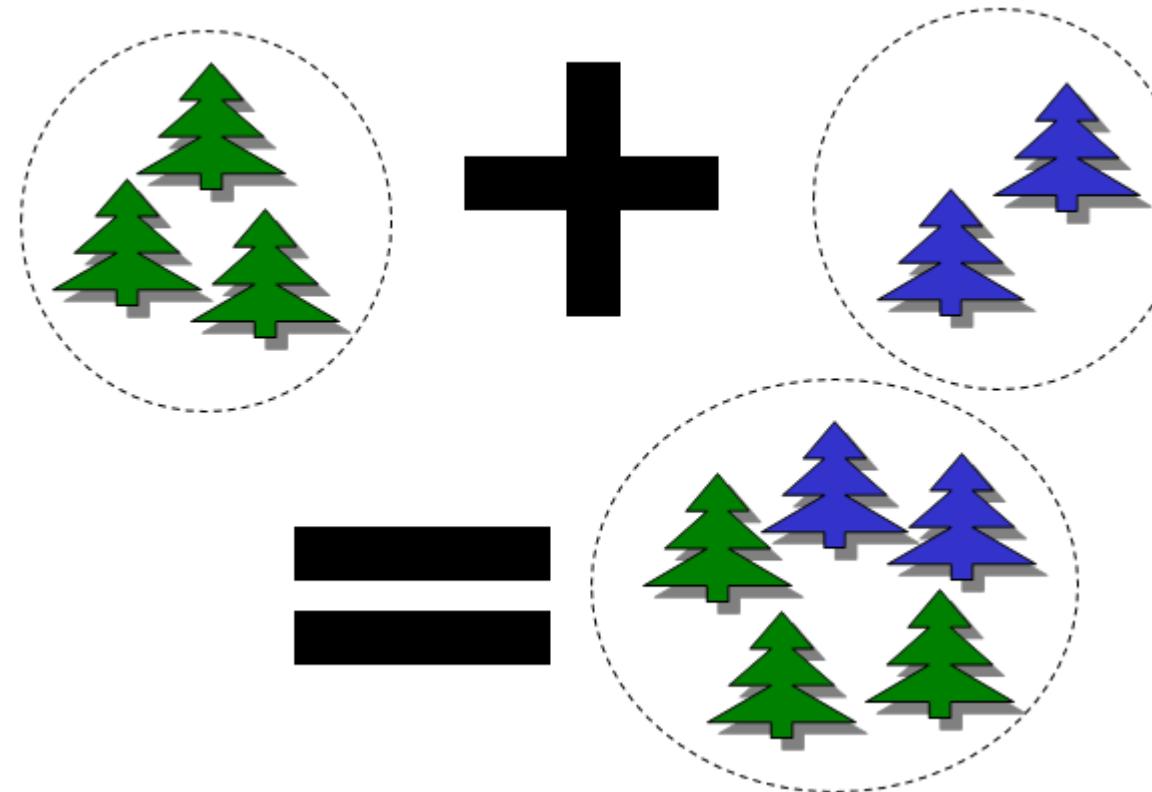
*“El software es un conjunto de reglas o programas que dan instrucciones a un ordenador para que realice tareas específicas.”*

## • ¿QUÉ ES UN ALGORITMO?

*“Es una lista de instrucciones donde se especifica una sucesión de operaciones necesarias para resolver cualquier problema de un tipo dado”.*

Ejemplo sumar dos números

- **¿QUÉ ES UN ALGORITMO?**



# • ¿QUÉ ES UN ALGORITMO?

## Entrada

- ✓ ¿Qué se necesita para realizar los pasos?

## Salida

- ✓ ¿Qué se obtiene al final del algoritmo?

## Tipos de datos

- ✓ Números: enteros, decimales
- ✓ Texto: letras, palabras, frases
- ✓ Otros

# • ¿QUÉ ES UN ALGORITMO?

- ✓ Sirven para resolver un tipo de problema específico.
- ✓ Son secuencias de pasos concretos.
- ✓ Requiere la definición de la entrada y la salida.
- ✓ Adecuados para ser ejecutados por un computador

- **¿QUÉ TIENE QUE VER LA PROGRAMACION?**

La programación consiste en crear programas de computador que resuelvan problemas específicos.

Un programa de computador es la implementación de un algoritmo.

# • ¿QUÉ UN PROGRAMA DE COMPUTADOR?

- ✓ Es una secuencia de pasos a ejecutar
- ✓ Los pasos están descritos en un lenguaje especial.
- ✓ Este lenguaje se puede traducir al lenguaje del computador.
- ✓ Por lo general es un archivo de texto.
- ✓ El texto escrito en dicho lenguaje se denomina el código del programa.

# • DESCRIPCION DE UN ALGORITMO

**Es necesario contar con formas de expresar algoritmos**

- ✓ Diseño del algoritmo antes de codificar
- ✓ Diseño del algoritmo de manera independiente del lenguaje de programación

**Diferentes alternativas**

- ✓ Diagramas de flujo
- ✓ Pseudo – código

# • DESCRIPCION DE UN ALGORITMO

## Pseudo – código

- ✓ El algoritmo se expresa en lenguaje natural
- ✓ Expresa de manera genérica los pasos del algoritmo
- ✓ No provee detalles de la implementación particular del código final

# • DESCRIPCION DE UN ALGORITMO

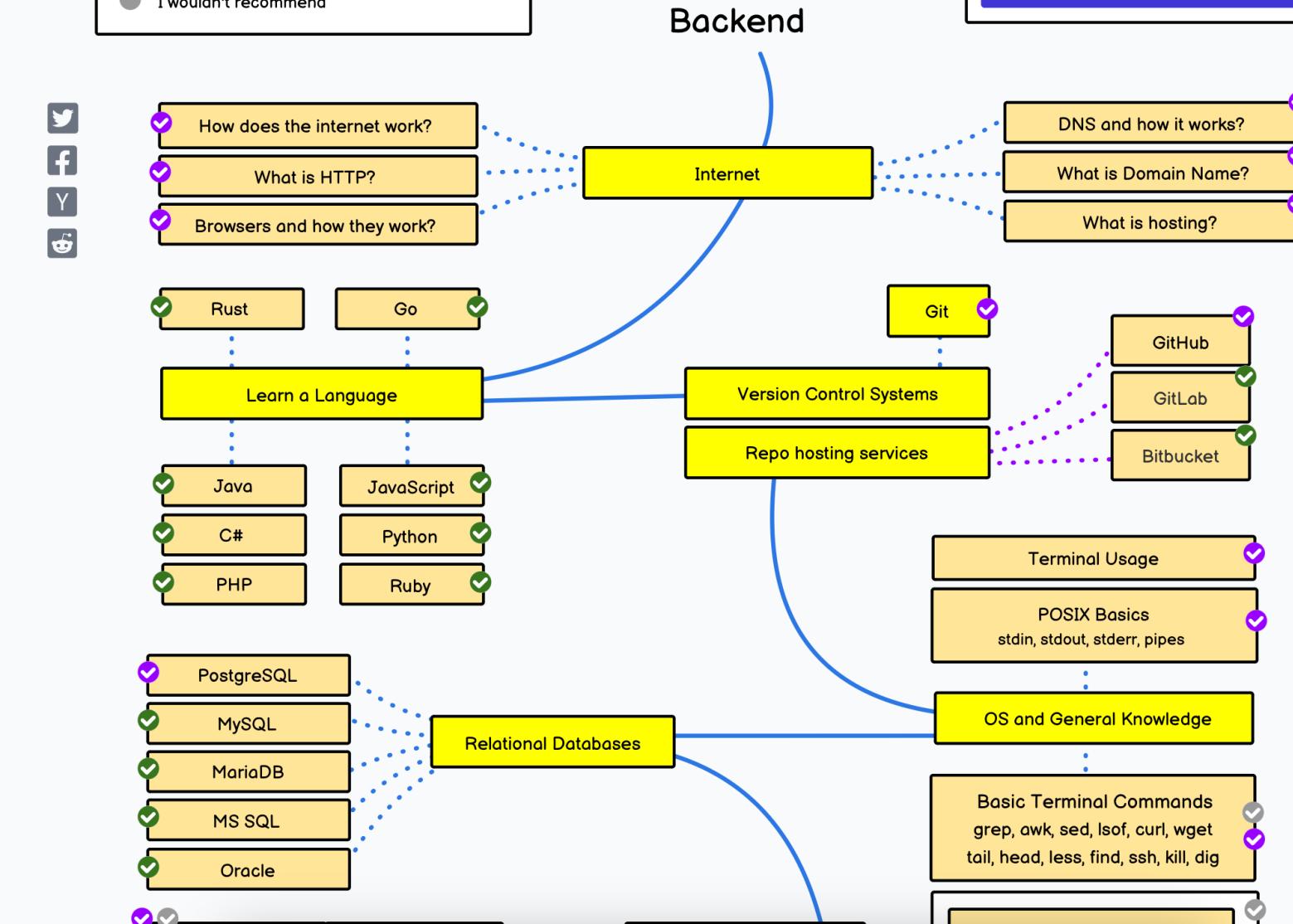
## Diagramas de flujo

- ✓ Presentan el algoritmo de manera gráfica.
- ✓ De gran utilidad para seguir la “ruta” de un algoritmo.
- ✓ Aplicables a muchas otras disciplinas.

# • DESCRIPCION DE UN ALGORITMO

I wouldn't recommend

<https://roadmap.sh>



Tomado de: <https://roadmap.sh/backend>

# • CONSTRUCCION DE UN ALGORITMO



1. Definir el problema a resolver
2. Identificar las entradas del algoritmo
3. Identificar la salida del algoritmo
4. Definir los pasos a seguir para convertir las entradas en la salida
5. Seguir los pasos y comprobar que el algoritmo sea correcto analizando la salida.
6. Revisar los pasos y hacer las correcciones.
7. Resolver el problema.

# • CONSTRUCCION DE UN ALGORITMO



Inicio

Variables Peso, Precio, Pago

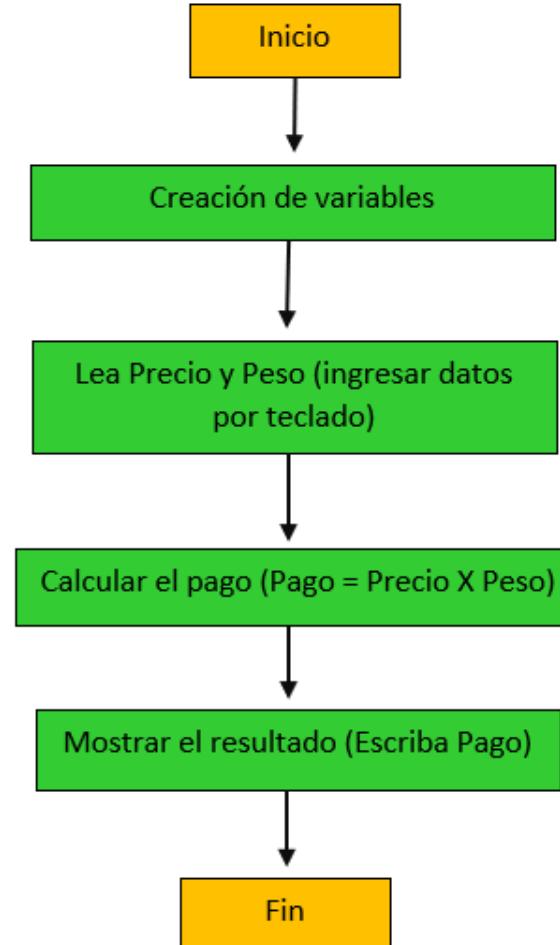
Lea Precio y Peso

Calcular el Pago= Precio X Peso

Escriba Pago

Fin

# • CONSTRUCCION DE UN ALGORITMO



# • CONSTRUCCION DE UN ALGORITMO



## Operaciones básicas

- ✓ Entrada de datos
- ✓ Salida de datos
- ✓ Utilización de variables
- ✓ Utilización de constantes
- ✓ Aplicación de operadores
- ✓ Asignación de valores

## Combinación de operaciones básicas

- ✓ Secuencial
- ✓ Selectiva
- ✓ Repetitiva

## • ENTRADA DE DATOS

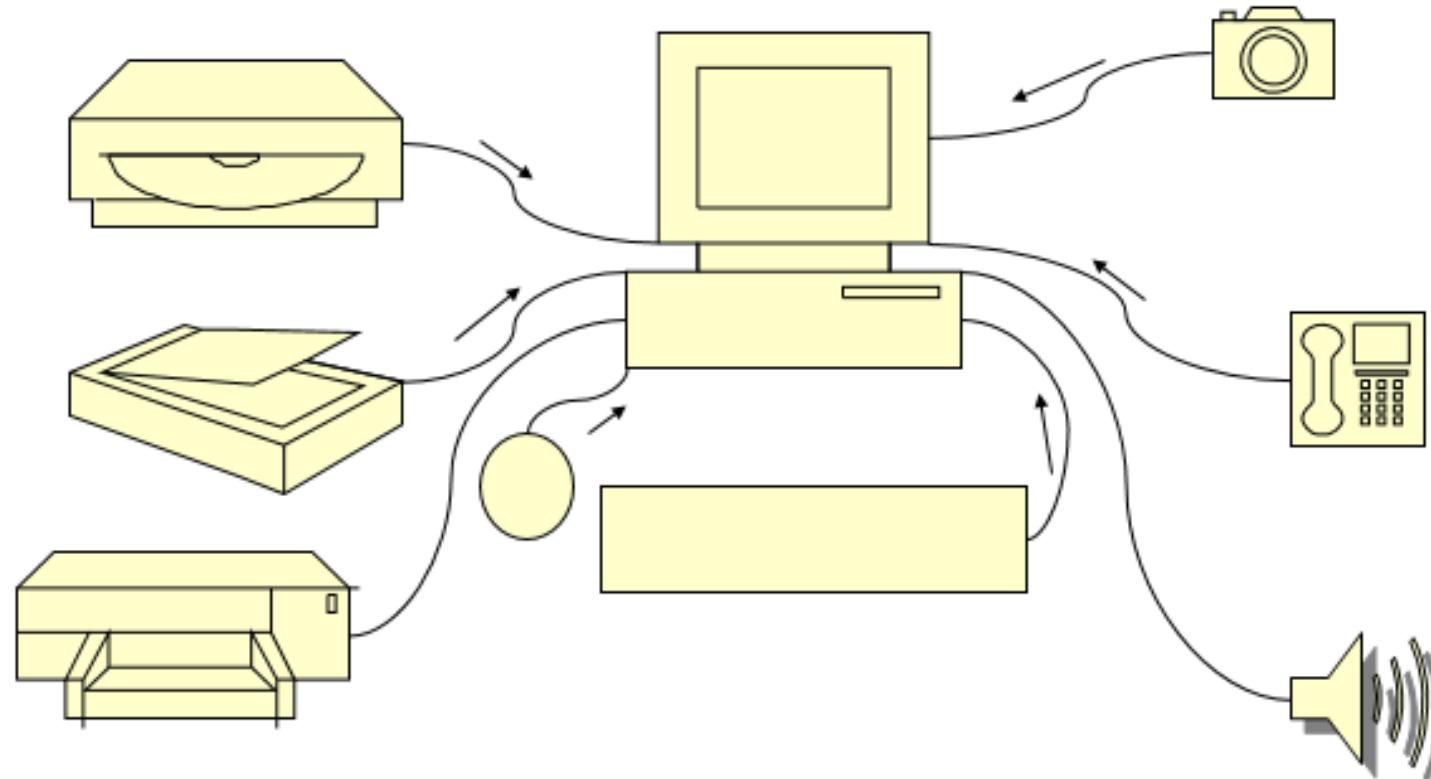
- ✓ Los algoritmos son para solucionar tipos de problemas
- ✓ Es imprescindible poder entregar entradas distintas en cada ejecución
- ✓ La entrada de datos se realiza mediante algún dispositivo

# • ENTRADA DE DATOS

## Dispositivos de entrada

- ✓ Teclado
- ✓ Mouse
- ✓ Botones
- ✓ Censores de tacto
- ✓ Cámaras digitales
- ✓ Scanners
- ✓ Archivos

# • ENTRADA DE DATOS



## • SALIDA DE DATOS

De nada sirve implementar un algoritmo si no podemos saber su resultado.

Al finalizar el algoritmo (o durante), es imprescindible obtener la información resultante de su ejecución.

La salida de datos se realiza mediante dispositivos.

# • SALIDA DE DATOS

## Dispositivos de salida

- ✓ Pantalla
- ✓ Impresora
- ✓ Parlantes
- ✓ Tableros luminosos
- ✓ Motores
- ✓ Tarjeta de red
- ✓ Archivos

# • UTILIZACION DE VARIABLES

K es un dato de entrada, y también  
Se considera una variable

$$G=K/1000$$

Esta variable se denomina G y  
se utiliza para recordar el valor  
de un gramo de manzana.

# • UTILIZACION DE CONSTANTES

$$G=K/1000$$



La constante “1000” sirva para transformar el valor Por kilo a un valor por gramo

# • APLICACIÓN DE OPERADORES

- ✓ Para obtener resultados, generalmente es necesario “transformar” las entradas en la salida.
- ✓ Para esto se aplican operadores de distinta índole
  - ✓ Aritméticos ( + , - , \* , / )
  - ✓ Lógicos (igual que, mayor que, menor que, y, o, no)
- ✓ Los operadores requieren de operandos y entregan un resultado.
- ✓ Por lo general, los operadores son unarios o binarios.

- Revisemos

<https://kodingergoy.arkivert.no>

# ¿Qué es PHP?

PHP, es un lenguaje de programación de **código abierto** que permite el desarrollo de aplicaciones WEB o aplicaciones WEB dinámicas, el cual es apto para incrustar en lenguaje HTML

Tomado de:  
<https://www.php.net/manual/es/index.php>





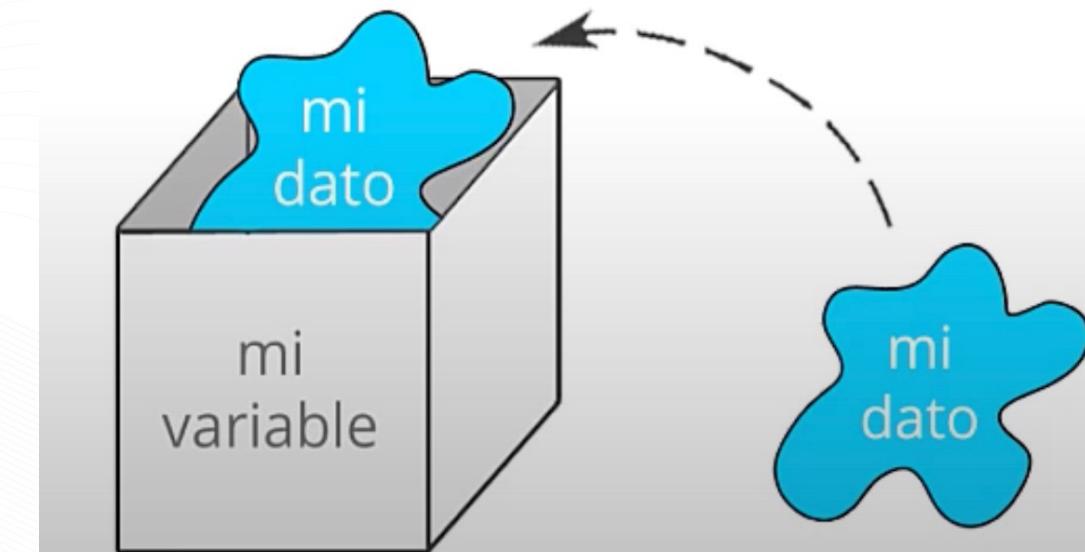
# 1. Conceptos básicos

- ¿Qué es una variable?

Una variable es un espacio en memoria con un nombre reservado para almacenar un valor correspondiente a un tipo de dato

El valor de una variable, puede cambiar durante la ejecución del código.

Tomado de:  
<https://www.php.net/manual/es/language.variables.basics.php>



# 1. Conceptos básicos

- **Reglas para definir una variable.**
  1. En PHP las variables se presentan con un signo de dólar seguido por el nombre de la variable
  2. El nombre de una variable debe empezar con una letra un un guion bajo( \_ ), seguido de cualquier número, letra. **NO PUEDE EMPEZAR CON NUMEROS.**
  3. El nombre de las variables es sensible a mayúsculas y minúsculas

Tomado de:

<https://www.php.net/manual/es/language.variables.basics.php>

# 1. Conceptos básicos

- ¿Qué es una constante?

En programación una constante es un valor que no puede ser alterado o modificado durante la ejecución de un programa.

Una constante corresponde a una longitud fija de un área reservada en memoria.

Ej: PI, DNI

# 1. Conceptos básicos

## • Reglas para definir una constante.

1. El nombre de una constante sigue las mismas reglas de una variable de PHP, es decir que un nombre valido de una constante empieza por una letra o un guion bajo, seguido por la asignación de nombre que se le quiera dar.
2. Por defecto una constante distingue mayúsculas y minúsculas.
3. Por convención, los identificadores de constantes siempre se declaran en **MAYUSCULA**

 sintaxis.php X

clas >  sintaxis.php

```
1  <?php  
2  
3  define("CONSTANTE1", "Hola Mundo");  
4  
5  echo CONSTANTE1;  
6  
7  
8  ?>
```

 localhost/clas/sintaxis.php

Hola mundo



localhost/clas/sintaxis.php



Hola mundo

 sintaxis.php X

clas >  sintaxis.php

```
1  <?php  
2  
3  $const = 'Hola mundo';  
4  
5  echo $const;  
6  
7  ?>
```

# 1. Conceptos básicos

- Operadores aritméticos.

Nombre	Símbolo en PHP
Suma	+
Resta	-
División	/
Multiplicación	*
Modulo	%
Exponenciación	**

# 1. Conceptos básicos

- Jerarquía de los operadores aritméticos.

Jerarquía	Operador
1	( )
2	**
3	*, /
4	+, -
5	%

# 1. Conceptos básicos

- Operadores de comparación.

Nombre	Símbolo en PHP	Respuesta
Igual	<code>==</code>	Booleano
Idéntico	<code>===</code>	Booleano
Diferente	<code>!=</code>	Booleano
Menor que	<code>&lt;</code>	Booleano
Mayor que	<code>&gt;</code>	Booleano
Menor o igual que	<code>&lt;=</code>	Booleano
Mayor o igual que	<code>&gt;=</code>	Booleano

# 1. Conceptos básicos

- Desarrolle (Esto es un solo ejercicio)
  1. Asigne valor a dos variables \$a y \$b.
  2. Opere las variables (+, -, \*, /).
  3. Guarde el resultado en una tercera variable (\$res).
  4. Imprima en pantalla TODOS los resultados.

# 1. Conceptos básicos

- **Desarrolle**

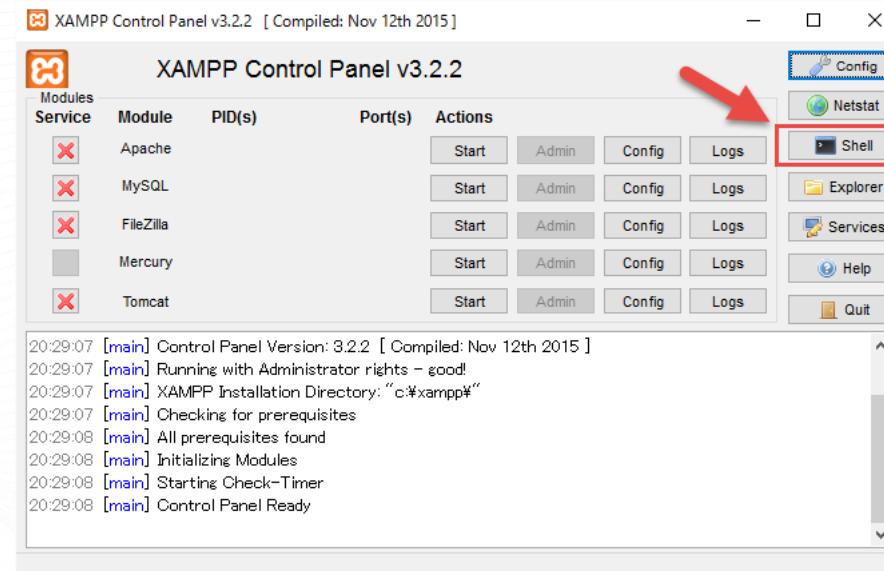
Un empleado el cual tiene un sueldo base de \$2.500.000 necesita saber cuánto será su pago, si de su sueldo base le descuenta un 10% por aportes. Diseñe un algoritmo lógico en donde asigne valores a diferentes variables y guarde datos resultantes para posteriormente ser mostrados en pantalla y ayudar al empleado con sus finanzas.

**En un diagrama de flujo deje ver el ciclo de trabajo**

# 1. Conceptos básicos

- **Ingresar datos por consola**

Una vez abrexampp se encontraran en la parte superior derecha un botón llamado “Shell”. Al oprimir este botón les aparecerá una consola de comando (CMD)

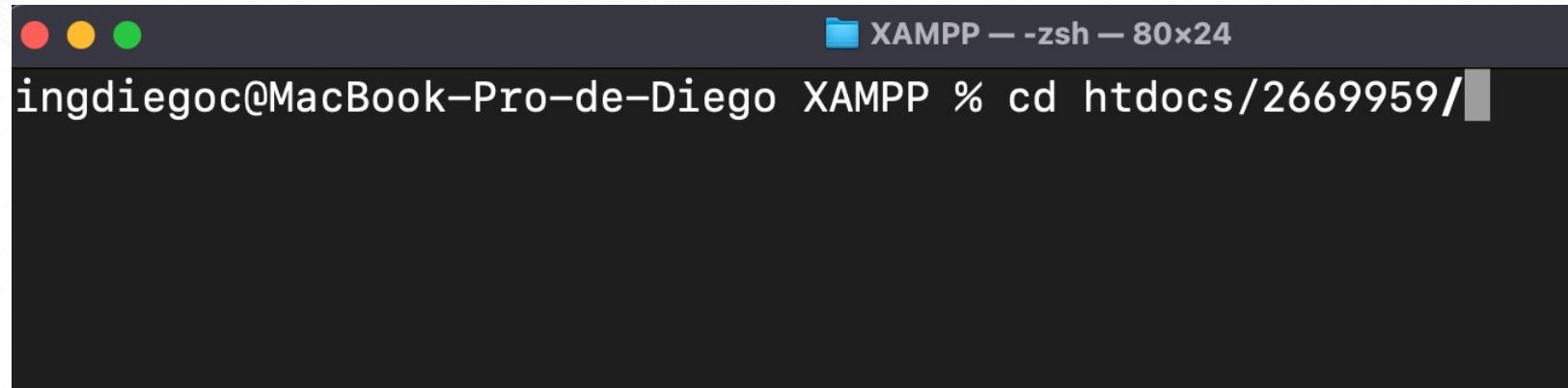


# 1. Conceptos básicos

- Ingresar datos por consola

Esta consola se abrirá en la ubicación “C: \xampp” por lo que debe navegar a “htdocs” y posteriormente a la carpeta que contiene el proyecto que esta ejecutando, haciendo uso del siguiente comando:

**cd htdocs/nombreDelProyecto**

A screenshot of a terminal window titled "XAMPP — -zsh — 80x24". The window has three colored window control buttons (red, yellow, green) at the top left. The title bar also shows the XAMPP logo. The main area of the terminal shows the command "cd htdocs/nombreDelProyecto" being typed by a user named "ingdiegoc" on a MacBook Pro. The command is partially completed, showing "/". The background of the terminal is dark, and the text is white.

# 1. Conceptos básicos

- Ingresar datos por consola

Una vez dentro deben escribir el comando reservado “php nombreDelScript.php” para que muestre en pantalla lo que desean



A screenshot of a terminal window on a Mac OS X system. The window title bar shows three colored dots (red, yellow, green) and the path "2669959 — zsh — 80x24". The main area of the terminal displays the command "ingdiegoc@MacBook-Pro-de-Diego 2669959 % php logica1.php" followed by the output "El resultado de la suma entre 3.14 y 1 es: 4.14%".

# 1. Conceptos básicos

- **readline**

“readline” es una palabra reservada de PHP que permite ingresar por consola diferentes tipos de datos como números o letras. Su forma de uso es:

```
❶  readLine.php
1   <?php
2
3   $a = readline("Ingrese su nombre \n");
4
5   echo $a;
6
7   ?>
```

# 1. Conceptos básicos

- **readline**

**“readline” Solo funciona cuando abre desde la consola desde XAMPP.**

# Desarrolle

1. Escriba un algoritmo que solicite su nombre y apellido por teclado, almacénelos en variables diferentes, posteriormente imprímalos por pantalla en una sola línea con el mensaje “BIENVENIDO **nombre y apellido**”.
  
2. Escriba un algoritmo que le permita ingresar dos números enteros por el teclado, posteriormente calcule la suma e imprima los números leídos y el resultado de la suma.
  
3. Escriba un algoritmo que le permita ingresar dos números enteros por el teclado, posteriormente calcule la suma, la resta y la multiplicación de los dos números e imprima los números leídos y los resultados de las tres operaciones.
  
4. Una empresa está realizando un aumento del 15% al sueldo base de sus trabajadores, escriba un algoritmo que solicite el nombre de un trabajador y su sueldo base, a continuación, el algoritmo debe calcular el aumento y el valor final a pagar al trabajador

# 1. Conceptos básicos

- Jerarquía de los operadores aritméticos.

Jerarquía	Operador
1	( )
2	**
3	*, /
4	+, -
5	%

# 1. Conceptos básicos

- Operadores lógicos.

Símbolo	Nombre
AND, and	And (y)
OR, ir	Or (ó)
!	Not (!)
&&	And (y)
	Or (o)

# 1. Conceptos básicos

- Operadores lógicos (AND).
- True = 1; False = 0

V1	V2	Resultado
0	0	0
0	1	0
1	0	0
1	1	1

# 1. Conceptos básicos

- Operadores lógicos (OR).
- True = 1; False = 0

V1	V2	Resultado
0	0	0
0	1	1
1	0	1
1	1	1

# 1. Conceptos básicos

- Operadores lógicos (NOT).
- True = 1; False = 0

V1	Resultado
!0	1
!1	0

# 1. Conceptos básicos

- Operadores de incremento/decremento

Símbolo	Nombre
<code>++</code>	Incremento
<code>--</code>	Decremento

## EJEMPLOS

<code>++\$variable</code>	Pre-Incremento
<code>\$variable++</code>	Post-Incremento
<code>--\$variable</code>	Pre-Decremento
<code>\$variable--</code>	Post-Decremento

# 1. Conceptos básicos

- **Tipos de datos:**

- booleanos (Boolean)
- Enteros (Integer)
- Flotante (Double)
- Cadena de caracteres (String)
- Array
- Iterable
- Objetos
- Recursos
- NULO



# 1. Conceptos básicos

- **Tipos de datos (var\_dump):**

A large magnifying glass is positioned over the list of data types, focusing on the text inside the lens.

- booleanos (Boolean)
- Enteros (Integer)
- Flotante (Double)
- Cadena de caracteres (String)

# 2. Estructuras condicionales

---

## 2. Estructuras condicionales



- Las estructuras condicionales son utilizadas para tomar decisiones en función de que se cumpla o no una determinada condición.

## 2. Estructuras condicionales



## • Estructura condicional simple (if)

## 2. Estructuras condicionales



- Estructura condicional if

Ejercicio:

En una fabrica de computadores se plantea ofrecer a los clientes un descuento que dependerá del numero de computadores que compre. Si los computadores son menos de 5, se le dará un descuento del 10%, si el numero de computadores comprados es mayor o igual a 5 pero menos de 10 se le otorga un 20% de descuento; y si son mas de 10 se les da un 40% de descuento. El precio de cada computador es de \$700 USD.

## 2. Estructuras condicionales



- Estructura condicional multiple if-else-elseif

Método 1	Método 2
<pre>If(\$a&gt;\$b){ echo "\$a es mayor que \$b"; }elseif (\$a == \$b) { echo "\$a es igual que \$b"; } Else { Echo "Ninguna es correcta"; }</pre>	<pre>If(\$a&gt;\$b): echo "\$a es mayor que \$b"; elseif (\$a == \$b): echo "\$a es igual que \$b"; Else : Echo "Ninguna es correcta"; endif;</pre>

# Ejercicios

1- Ejercicio de comparación numérica:  
Escribe un programa que solicite al usuario que ingrese un número. Luego, el programa debe imprimir si el número ingresado es mayor, menor o igual a 10 utilizando la sentencia if.

2-Ejercicio de comparación de cadenas:  
Escribe un programa que solicite al usuario que ingrese su nombre. Si el nombre ingresado es "Juan", el programa debe imprimir "¡Hola Juan!" en la pantalla. Si el nombre es diferente, el programa debe imprimir "Lo siento, no te conozco".

# Ejercicios

3- Ejercicio de comparación múltiple:

Escribe un programa que solicite al usuario que ingrese un número del 1 al 7. Luego, el programa debe imprimir el día de la semana correspondiente al número ingresado. Por ejemplo, si el usuario ingresa "1", el programa debe imprimir "Lunes". Si el número ingresado no está en el rango válido, el programa debe imprimir "Número inválido".

4- Ejercicio de anidación de sentencias if:

Escribe un programa que solicite al usuario que ingrese su edad. Si la edad es mayor o igual a 18 años, el programa debe preguntar si tiene licencia de conducir. Si la respuesta es "sí", el programa debe imprimir "Puedes conducir". Si la respuesta es "no", el programa debe imprimir "Debes obtener una licencia de conducir primero". Si la edad es menor de 18 años, el programa debe imprimir "No puedes conducir".

5- Ejercicio de uso de operadores lógicos:

Escribe un programa que solicite al usuario que ingrese dos números. Luego, el programa debe imprimir si ambos números son mayores que 10 utilizando la sentencia if y los operadores lógicos "&&" y ">". Si ambos números son mayores que 10, el programa debe imprimir "Ambos números son mayores que 10". Si solo uno de los números es mayor que 10, el programa debe imprimir "Solo uno de los números es mayor que 10". Si ninguno de los números es mayor que 10, el programa debe imprimir "Ninguno de los números es mayor que 10".

## 2. Estructuras condicionales



- Operador ternario

El operador ternario puede considerarse como una estructura if de una sola línea.  
Este está compuesto por tres partes:

```
<OPERADOR> ? <TRUE VALUE> : <FALSE VALUE>;
```

Se usa para operaciones lógicas por lo que el “echo” está rotundamente prohibido.

Ej:

```
(9>7) ? $total=10*7 : $total= 10*5;  
echo $total;
```

## 2. Estructuras condicionales



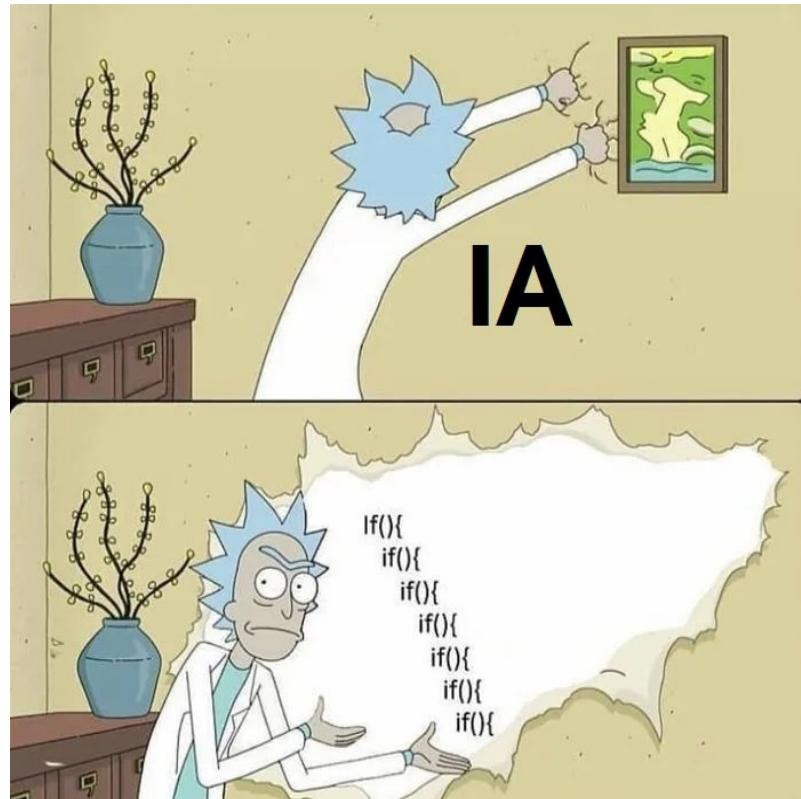
- Operador ternario

Ejercicio:

Hacer un programa que calcule el total a pagar por la compra de camisas. Si se compran 3 camisas o mas se debe aplicar un descuento del 20% sobre el total de la compra y si son menos de 3 camisas un descuento del 10%

## 2. Estructuras condicionales

- # • Estructura condicional anidada



## 2. Estructuras condicionales



- Estructura condicional anidada

```
if (condicion) {  
    if(condicion){  
        Bloque de código a ejecutar si la condición es TRUE  
    }else{  
        Bloque de código a ejecutar si la condición es FALSE  
    }  
} elseif (condicion) {  
    Bloque de código a ejecutar si la condición es TRUE  
} else {  
    Bloque de código a ejecutar si la condición es FALSE  
}
```

## 2. Estructuras condicionales



- Estructura condicional anidada
- Ejercicio:

Pida a un usuario la edad y genero, para que el sistema le indique si ya esta en la edad para pensionarse. Tenga en cuenta que un Hombre se puede pensionar a los 60años o mas, mientras que las mujeres lo pueden hacer después de los 54 años.

## 2. Estructuras condicionales



### • Desarrolle

1. Escribir un programa que verifique si un número es positivo, negativo o cero.
2. Crear un programa que determine si un número es par o impar.
3. Desarrollar un programa que determine si un número es divisible por 3 y por 5.
4. Escribir un programa que clasifique a un estudiante en Aprobado (nota mayor o igual a 60), Reprobado (nota menor a 60) o Sobresaliente (nota mayor o igual a 90).
5. Crear un programa que determine si un año es bisiesto o no. Un año es bisiesto si es divisible por 4 y no divisible por 100, o si es divisible por 400.

# 2. Estructuras condicionales



## • Desarrolle

6. Desarrollar un programa que indique el rango de una calificación de acuerdo a la siguiente tabla:

- 90-100: Excelente
- 80-89: Bueno
- 70-79: Regular
- 60-69: Aprobado
- 0-59: Reprobado

7. Escribir un programa que muestre el día de la semana correspondiente a un número del 1 al

7. Considera que 1 es Lunes y 7 es Domingo.

8. Crear un programa que determine si una letra ingresada es una vocal o una consonante.

9. Desarrollar un programa que determine si un número es primo o no. Un número es primo si solo es divisible por 1 y por sí mismo.

10. Escribir un programa que calcule el descuento aplicable a una compra de acuerdo al monto total:

- Si el monto es mayor o igual a \$1000, aplicar un descuento del 10%.
- Si el monto es mayor o igual a \$500, aplicar un descuento del 5%.
- Si el monto es menor a \$500, no aplicar descuento.



# G R A C I A S

Línea de atención al ciudadano: 01 8000 910270

Línea de atención al empresario: 01 8000 910682



[www.sena.edu.co](http://www.sena.edu.co)