



DESARROLLO BACK END -INTERMEDIO

Autor de contenido

Luz Liliana Herrera Polo



Tabla de Contenido

2.14. Funciones



Presentación

En el presente documento está presentada de manera detallada la propuesta a desarrollar con el curso Desarrollador Backend, ofrecidos principalmente a personas con educación media culminada que vivan en Colombia y que estén interesados en aprender en el campo de las tecnologías de la información. Con este curso se busca que el estudiante al finalizar la formación sea capaz de desarrollar aplicaciones backend robustas implementando lógica de programación con JavaScript, bases de datos y sistemas de control de versiones.

Objetivos del curso (competencias)



Objetivo general

Aprender a desarrollar aplicaciones modernas para procesar, almacenar y proteger la información de un programa de manera robusta y escalable. Implementando buenas prácticas y la lógica de programación con JavaScript.

Objetivo específico

- Por medio del pensamiento lógico, el lenguaje de programación y las bases de datos diseñar y desarrollar aplicaciones de software que se ajusten a los requerimientos planteados en cualquier problema.
- Organizar la realización de pruebas que verifiquen el correcto funcionamiento de las aplicaciones de software desarrolladas y verificando también que el desarrollo se ajusta a los requisitos de análisis y diseño.
- Utilizar un sistema de control de versiones con el fin de gestionar los cambios en el código fuente a lo largo del tiempo.

Objetivo del módulo



El módulo tiene como eje principal aprender conceptos fundamentales e iniciales de la lógica de programación como lo son los sistemas binario, octales, hexadecimales, tablas de verdad y operadores lógicos. Por otro lado, también se desea explicar el proceso de creación de diagramas de flujo y algoritmos para poder aplicarlos.

Mapa de contenido de la unidad

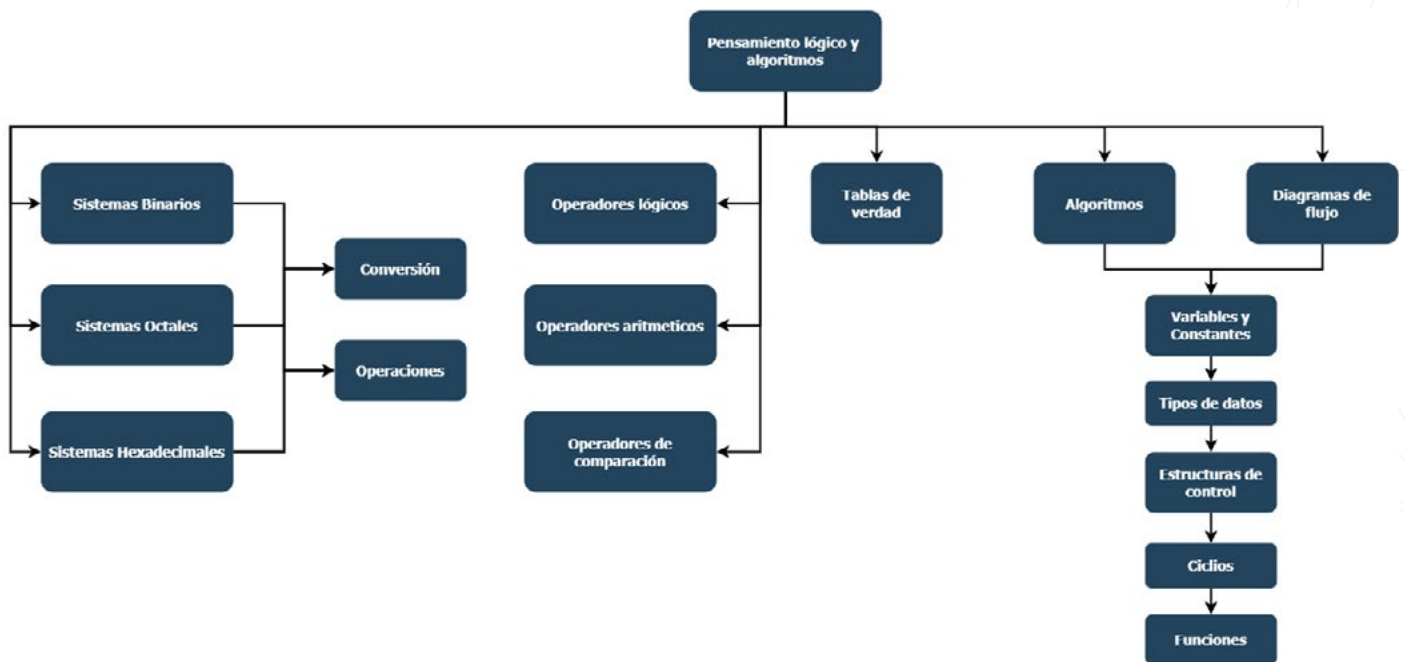


Figura 1. Mapa de contenido del módulo 2

Módulo 2

Pensamiento lógico y algoritmos

2.1. Sistemas binarios en la computación.

Clase teórica en donde se explica la importancia de los sistemas binarios, su diferencia con el sistema decimal y sus usos en las distintas ramas de la informática.

Sistema binario

El sistema binario es comúnmente conocido porque es el sistema que utilizan los computadores y demás dispositivos electrónicos. Este sistema es un sistema en base 2, esto quiere decir que este sistema solo utiliza dos cifras para representar todos sus números, estas dos cifras son el 1 y el 0, se suele utilizar en los computadores debido a que solo trabajan en dos niveles de voltaje: prendido o encendido (1), y apagado (0). Su notación por lo general es con un subíndice 2 en el número para diferenciarlo, cuando hablamos de sistemas en base 10 que es el sistema que usamos comúnmente la notación es con un subíndice 10. Por ejemplo $3_{10} = 11_2$

Para aprender más sobre este sistema puedes visitar el siguiente enlace en donde se explica de manera fácil los sistemas binarios <https://youtu.be/jKxxb0i-xlQ>

2.2. Sistemas octales, hexadecimales y ASCII

Clase teórica en donde se explica la importancia de los sistemas octales y hexadecimales, y las diferencias entre los distintos sistemas numéricos. También se hará énfasis en sus usos en las distintas ramas de la informática.

Sistema octal.

El sistema octal es un sistema en base 8, esto quiere decir que este sistema solo utiliza ocho cifras (0,1,2,3,4,5,6,7) Por lo tanto, cada dígito de un número octal puede tener cualquier valor de 0 a 7. Su notación por lo general es con un subíndice 8. Por ejemplo $11_8 = 13_{10}$.

Para aprender más sobre este sistema puedes visitar el siguiente enlace en donde se explica de manera fácil los sistemas octales https://youtu.be/IWE1jP5D_40

Sistema hexadecimal.

El sistema hexadecimal es un sistema comúnmente utilizado en la informática para facilitar la legibilidad de números grandes o secuencias de bits largas. En el sistema binario el número se agrupa en cuatro bits cada uno y se convierten al sistema hexadecimal. Con ello, a partir de una larga secuencia de unos y ceros del sistema binario, se obtiene un número hexadecimal más corto y de fácil lectura para el dispositivo. Con esto se entiende que el sistema hexadecimal es el sistema que nos ayuda a representar secuencias de bits muy largas de manera compacta. Los usos que encontramos para este sistema numérico en la informática son:

- Dirección de origen y de destino de protocolos de Internet (IP).
- Códigos ASCII.
- Descripción de los códigos de color en diseño web con el lenguaje de hojas de estilo CSS.

El sistema hexadecimal es un sistema en base 16, esto quiere decir que este sistema solo utiliza dieciséis cifras (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F) Por lo tanto, cada dígito de un número hexadecimal se puede tener cualquier valor entre 0 a 9 y de la A a la F. Su notación por lo general es con un subíndice 16. Por ejemplo $20_{10} = 14_{16} = 0001\ 0100_2$.

Para aprender más sobre este sistema puedes visitar el siguiente enlace en donde se explica de manera fácil los sistemas hexadecimales <https://youtu.be/69aj0Vs8D8c>

ASCII

ASCII (American Standard Code for Information Interchange) que en español traduce Código Estándar Norteamericano para Intercambio de Información. Es un estándar que busca crear código para los caracteres alfanuméricos, nace de la necesidad de que los distintos dispositivos electrónicos tuvieran un lenguaje en común para comunicarse. ASCII utiliza una escala decimal de 0 a 127 el cual es convertido de decimal a binario por la computadora para poder ser procesado.

...	100	101	110	...
...
0101	E	U	e	...
0110	F	V	f	...
0111	G	W	h	...
1000	H	X	g	...
...

Figura 2. ASCII. Fuente. <https://www.seobility.net/es/wiki/ASCII/>

2.3. Conversión entre sistemas numéricos

Clase teórica práctica en donde el estudiante aprender a realizar las conversiones entre los distintos sistemas numéricos (binario, octal, decimal y hexadecimal). Todo esto para darle una introducción al estudiante de cómo funcionan estos sistemas numéricos en la computación tradicional.

Conversión de decimal a binario, octal y hexadecimal

•**Conversión decimal a binario:** Para hacer la conversión de decimal a binario, hay que ir dividiendo el número decimal entre dos y anotar en una columna a la derecha el residuo, si el resto de la división es par se anota un 0 y si es impar se anota 1. Como se muestra en el siguiente video <https://youtu.be/3xGIXz2dza0>

•**Conversión decimal a octal:** Para hacer la conversión de decimal a octal, hay que ir dividiendo el número decimal entre ocho y anotar en una columna a la derecha el residuo, se deja de dividir hasta que el cociente sea menor a ocho y se anotan los números del valor más significativo al valor menos significativo. Como se muestra en el siguiente video <https://youtu.be/bKbuHjDUMdY>

•**Conversión decimal a hexadecimal:** Para hacer la conversión de decimal a hexadecimal, hay que ir dividiendo el número decimal entre ocho y anotar en una columna a la derecha el residuo, se deja de dividir hasta que el cociente sea menor a 16 y se anotan los números del valor más significativo al valor menos significativo guiándonos de la siguiente tabla dependiendo del valor que sobre se pone el dígito que le corresponde en hexadecimal. Como se muestra en el siguiente video <https://youtu.be/Tq40h-EBWmw>

Table 8.8 Hexadecimal values and bit patterns

Digit	Value	Pattern	Digit	Value	Pattern
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	A	10	1010
3	3	0011	B	11	1011
4	4	0100	C	12	1100
5	5	0101	D	13	1101
6	6	0110	E	14	1110
7	7	0111	F	15	1111

Figura 4. Valores hexadecimales. Fuente. <https://livebook.manning.com/book/modern-c/chapter-8/>

2.4. Operaciones de sistemas numéricos

Clase teórica práctica en donde el estudiante aprender a realizar las operaciones básicas (suma, resta, multiplicación y división) de los distintos sistemas numéricos (binario, octal, decimal y hexadecimal). Todo esto para darle una introducción al estudiante de cómo funcionan estos sistemas numéricos en la computación tradicional.

Operaciones de sistemas numéricos

Las operaciones aritméticas que ya conocemos de las matemáticas básicas como: suma, resta, multiplicación y división que se realizan en decimal también se pueden realizar en cualquier sistema numérico (binario, octal, hexadecimal), aplicando las mismas reglas y teniendo en cuenta la base en la que se realizan las operaciones.

Suma

Para realizar sumas en diferentes sistemas numéricos sigue siendo el mismo que en el sistema decimal, lo único es que se considera la base de la operación. Entonces, en general, al sumar dos dígitos, si el resultado de la suma excede el dígito más significativo de un sistema numérico en particular, se puede determinar que el resultado se divide entre la base del sistema y el resto de esta división, se coloca debajo de la línea y el cociente se suma a la siguiente columna desde la izquierda.

Resta

Al restar, debemos verificar si el sustraendo de la operación es mayor que el minuendo, porque si es así se debe sumar la base al minuendo antes de llevar a cabo la resta de dos dígitos de cualquier columna. Cuando se comienza la operación de resta al minuendo se le suma la base, entonces el sustraendo de la columna izquierda próxima se le deberá sumar 1 antes de hacer la comparación entre el minuendo y sustraendo.

Multiplicación

Multiplicar en números decimales es lo mismo que multiplicar en otros sistemas numéricos, la única diferencia es la base. Como en cualquier sistema, la multiplicación, al multiplicar por 1 se obtiene la misma cantidad y al multiplicar por cero resulta cero.

División

La división es más complicada que las tres operaciones aritméticas anteriores porque se sabe que implica restas y multiplicaciones. En este caso se recomienda utilizar la llamada división desarrollada. Esto le permite realizar primero la multiplicación y luego la resta.

2.5. Tablas de verdad y lógica booleana

Con las tablas de verdad, al estudiante se le dará una introducción teórica a la lógica en la computación debido a que estas son estrategias de la lógica simple que permiten establecer la validez de varias propuestas en cuanto a cualquier situación.

Tablas de verdad

Una tabla de verdad es una estrategia lógica simple que nos permite establecer la validez de algunas proposiciones en todas las circunstancias. Es decir, nos permite definir las condiciones necesarias para que una proposición sea verdadera, falsa o contingente.

Negación		
A	\neg	A
V		F
F		V

Conjunción				
A	B	A	\wedge	B
V	V			V
V	F			F
F	V			F
F	F			F

Disyunción				
A	B	A	\vee	B
V	V			V
V	F			V
F	V			V
F	F			F

Condicionalidad				
A	B	A	\rightarrow	B
V	V			V
V	F			F
F	V			V
F	F			V

Bicondicionalidad				
A	B	A	\leftrightarrow	B
V	V			V
V	F			F
F	V			F
F	F			V



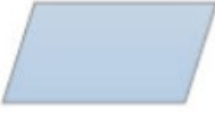


Figura 4. Conectores lógicos y tablas de verdad. Fuente: <https://tomi.digital/es/59552/valores-de-las-tablas-de-verdad>

2.6. Diagramas de flujo

Es importante para el estudiante en este punto del curso comenzar a desarrollar habilidades de lógica, los diagramas de flujo ayudarán a que el estudiante describa un proceso, sistema o algoritmo informático por medio de estos diagramas.

Diagramas de flujo

Un diagrama de flujo usa símbolos para representar una secuencia o pasos lógicos (ordenados) para realizar una tarea. Se compone de los siguientes símbolos:

Símbolo	Nombre	Función
	Inicio / Final	Representación del inicio y el fin de un diagrama de flujo
	Línea de flujo	Indica el orden de ejecución de las operaciones del diagrama de flujo.
	Entrada / Salida	Representa la entrada y salida de datos en un proceso.
	Proceso	Indica cualquier tipo de operación.
	Decisión	Permite analizar una situación, analizando los valores verdadero y falso.

El siguiente ejemplo ilustra el diagrama de flujo realizado para calcular el factorial de un número

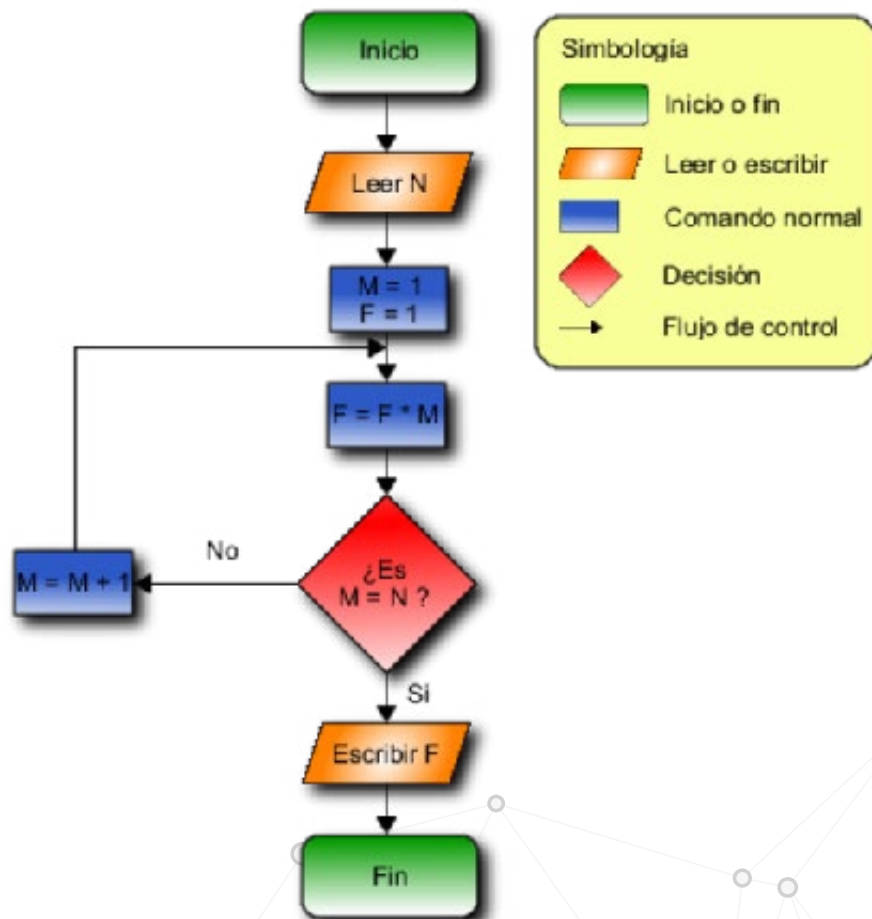


Figura 5. Diagrama de flujo de factorial. Fuente: https://commons.wikimedia.org/wiki/File:Diagrama_de_flujo_de_factorial.xcf

2.7. Creación de algoritmos

Clases teórica prácticas en donde se darán las herramientas necesarias para comenzar a desarrollar algoritmos y poder aplicar los conocimientos teóricos tales como (variables, funciones, ciclos...) que se explicarán en el desarrollo del módulo.

¿Qué es un algoritmo?

Un algoritmo es una secuencia finita de instrucciones que realiza una serie de pasos para resolver un problema en específico. En otras palabras, un algoritmo informático resuelve un problema mediante instrucciones y reglas sencillas para al final mostrar el o los resultados obtenidos. Los algoritmos son de mucha relevancia e importancia en el mundo de la informática ya que permiten a los programadores y desarrolladores resolver diferentes problemas antes de escribir código en un lenguaje de programación para que las computadoras puedan entender.

¿Cómo hacer un algoritmo?

Para realizar un algoritmo hay que tener en cuenta tres componentes importantes de estos.

- Entrada:** Componente en donde se introducen todos los datos e información que el algoritmo necesita para operar.
- Proceso:** Con la información y datos recibidos en la entrada del algoritmo se utiliza la lógica para resolver el problema.
- Salida:** Los resultados obtenidos en el proceso se muestran en la salida del algoritmo.

Características

- Secuenciales:** Sus pasos son ejecutados secuencialmente, es decir uno detrás de otro.
- Precisos y concretos:** Se espera que el algoritmo sea objetivo y preciso al resolver un problema.
- Ordenados:** Los pasos ejecutados en el algoritmo deben ser ordenados para que puedan ser leídos, ejecutados y comprendidos de manera sencilla.
- Finitos:** Tienen un número finito de pasos que ejecutar.

2.8. Variables y constantes

Clase teórica en donde el estudiante aprenderá la diferencia entre variables y constantes. También comprenderá las distintas constantes que puede utilizar en la programación.

Variables

Se denomina variable al espacio en memoria donde se almacenan y recuperan los datos de un programa o aplicación de software. En programación las variables se utilizan para almacenar datos y estados, asignar los valores de unas variables a otras, representar valores en fórmulas matemáticas y mostrar valores en pantalla.

Constantes

En el entorno de desarrollo, se considera a una constante como un valor que no puede ser alterado o modificado durante la ejecución de un programa u algoritmo, únicamente puede ser leído.

Nomenclatura

Las variables se recomienda escribirlas en lowerCamelCase, es decir comenzar la primera letra de la primera palabra de la variable con minúscula y el resto de palabras comenzarlas con mayúsculas, por ejemplo: estaEsUnaVariable. Para las variables se deben tener en cuenta las siguientes recomendaciones:

- No deben comenzar por caracteres especiales.
- No debe comenzar por números.
- No utilizar espacios.

Se recomienda que los nombres de las variables sean breves pero significativos, y también que sean fáciles de recordar. Por otro lado, las constantes se recomienda que sean escritas en mayúsculas y separadas por guiones bajos. Por ejemplo: UNA_CONSTANTE.

2.9. Tipos de datos

Clase teórica en donde se explicaran los distintos tipos de datos que se pueden usar en la programación. Se explicarán las características de cada uno de estos tipos y se harán ejemplos prácticos.

Tipos de datos.

- Los tipos de datos más comunes en el desarrollo de software son:
- Números enteros.
- Números de coma flotante (decimales).
- Cadenas de caracteres.
- Caracteres.
- Booleanos.

2.10. Operadores lógicos

Una vez entendida la lógica proposicional y las tablas de verdad, se comenzará a desglosar los diferentes operadores que se pueden encontrar en el campo de la informática. En esta clase teórica se explicarán los operadores lógicos (AND, OR, XOR, NOT)

Operadores lógicos

Los operadores lógicos se usan comúnmente en desarrollo de software para combinar dos valores Booleanos y devolver un resultado verdadero o falso (TRUE o FALSE). Entre los operadores lógicos encontramos:

- **AND:** También llamado conjunción y es verdadero sólo si los dos elementos son verdaderos
- **OR:** También llamado disyunción y es falso sólo si los dos elementos son falsos.
- **XOR:** También llamado disyunción exclusiva y es verdadero si cualquiera de las expresiones es verdadera, pero no ambas pueden ser verdaderas.
- **NOT:** También llamado negación, cambia el valor de verdadero a falso y viceversa

2.11. Operadores aritméticos

Con el fin de que el estudiante se familiarice y sea capaz de comprender los usos de los operadores aritméticos se harán ejercicios lógicos con estos operadores para facilitar la comprensión de estos conceptos en la programación.

Operadores aritméticos

Los operadores aritméticos se utilizan en desarrollo de software para calcular el valor de dos o más números, o para cambiar el signo de los números de positivo a negativo o viceversa.

- **Suma (+):** Devuelve como resultado la suma entre dos números enteros o flotantes.
- **Resta (-):** Devuelve como resultado la resta entre dos números enteros o flotantes.
- **Multiplicación (*):** Devuelve como resultado la multiplicación entre dos números enteros o flotantes.
- **División (/):** Devuelve como resultado la división entre dos números enteros o flotantes.
- **Potencia (^):** Devuelve como resultado la operación de elevar un número a la potencia de un exponente.
- **Módulo (%):** Devuelve como resultado el residuo de la división entre dos números.

2.12. Operadores de comparación

Debido a que los operadores de comparación comparan dos expresiones y devuelven un valor Boolean que representa la relación de sus valores, el estudiante debe comprender y saber aplicar este tema para facilitar la comprensión de futuros temas como las funciones.

Operadores de comparación.

Los operadores aritméticos se utilizan en desarrollo de software para comparar entre dos valores y así poder devolver un valor booleano (true o false).

- **Mayor que (>):** Devuelve True si el primer valor es mayor que el segundo valor.
- **Menor que (<):** Devuelve True si el primer valor es menor que el segundo valor.
- **Mayor o igual que (>=):** Devuelve True si el primer valor es mayor o igual que el segundo valor.
- **Menor o igual que (<=):** Devuelve True si el primer valor es menor o igual que el segundo valor.
- **Igualdad (==):** Devuelve True si los dos valores son iguales.
- **Desigualdad (!=):** Devuelve True si los dos valores son desiguales o diferentes.

2.13. Estructuras de control

Hasta este punto el estudiante deberá ser capaz de desarrollar algoritmos consistido en simples secuencias de instrucciones, con las estructuras de control se pretende que el estudiante sea capaz de desarrollar algoritmos que involucren tomar decisiones dependiendo de ciertos parámetros o resultados o algoritmos que sean capaces de repetir una misma acción un número determinado de veces.

Estructuras de control

Un algoritmo está compuesto por pasos ordenados los cuales se deben ejecutar en orden para poder obtener resultados, sin embargo existen las estructuras de control en los algoritmos las cuales ayudan a llevar un control en el algoritmo, estas permiten que se realice una instrucción u otra de acuerdo a la evaluación de una condición. Se pueden clasificar en condicionales e iterativas.

Estructuras de control condicionales

Son estructuras de control de flujo que permiten que desarrollemos lógica de programación en el algoritmo, son instrucciones que permiten dirigir la ejecución del algoritmo hacia uno u otro proceso. Es decir, evaluar una condición para poder ejecutar un proceso pero no pueden ejecutarse los dos procesos al tiempo procesos al tiempo.

```

si (expresion_condicional) entonces
    proceso
sino
    proceso
fin_si
    
```


Estructuras de control iterativas

Las estructuras de control iterativas son una secuencia de instrucciones orientadas a ejecutarse varias veces, a esta estructura se les denomina comúnmente en el entorno de desarrollo como ciclo. Su objetivo principal es realizar un bloque de instrucciones repetitivamente hasta que se cumpla una condición.

```
mientras (expresion_condicional) hacer  
    proceso  
fin_mientras
```

2.14. Funciones

Las funciones en el mundo de la programación son de vital importancia, debido a que empaquetan una parte del código con una tarea específica del resto del programa. Estas funciones ayudarán a que el estudiante aprenda a manipular, organizar y reutilizar código de una manera acertada. El fin principal de este módulo es realizar algoritmos más complejos aplicando el concepto de función.

Funciones

Las funciones son una sección de un programa que se centra en calcular o resolver un proceso de forma independiente del resto del programa, esto ayuda en el código y en el algoritmo a reutilizar procesos. Sus tres componentes principales son:

- Parámetros (valores de entrada de la función)
- Proceso (código de la función)
- Resultado (valor de retorno o valores de salida)

Otros materiales para profundizar

Recursos de video



- Educación Activa. (2020, 6 octubre). Sistema Binario y Sistema Decimal (Educación Tecnológica) [Vídeo]. YouTube. <https://www.youtube.com/watch?v=jKxx-b0i-xlQ>
- alowallace14. (2019, 2 noviembre). Sistema Binario 05 [Vídeo]. YouTube. <https://www.youtube.com/watch?v=3MsT4lPAkbo>
- Shakmuria. (2017, 1 octubre). 19.-¿Qué es el sistema octal? Y su conversión al sistema decimal. [Vídeo]. YouTube. <https://www.youtube.com/watch?v=yc6aBN7DqnU>
- Carlos Guerrero UIB. (2022, 24 marzo). 1.3 Base 3, octal y hexadecimal [Vídeo]. YouTube. <https://www.youtube.com/watch?v=30eTvoBBTCU>
- videoconferencias. (2012, 19 abril). UTPL CONVERSIÓN DE SISTEMAS DE NUMERACIÓN [(CCEE)(LÓGICA MATEMÁTICA)] [Vídeo]. YouTube. <https://www.youtube.com/watch?v=l6uSJdm-uus>
- Aprende Sin Espinas. (2021, 25 septiembre). TABLAS DE VERDAD [Vídeo]. YouTube. <https://www.youtube.com/watch?v=MkAoAALyTRc>
- Matemáticas 1 - Universidad de Alicante. (2019, 9 septiembre). Ejercicios de tablas de verdad [4.2|2018] [Vídeo]. YouTube. <https://www.youtube.com/watch?v=lmb-MhyCpQbM>
- Cefuve. (2020, 7 enero). #4 Operadores lógicos, comparativos y aritméticos - Curso Arduino [Vídeo]. YouTube. <https://www.youtube.com/watch?v=Hdf4OT0JznQ>
- ARUMALS. (2022, 5 enero). 030. OPERADORES LÓGICOS [Vídeo]. YouTube. <https://www.youtube.com/watch?v=6Oh1ZZoVixA>
- Cefuve. (2019, 9 diciembre). #3 Variables y Constantes [int, float, bool, char] - Curso Arduino [Vídeo]. YouTube. <https://www.youtube.com/watch?v=K82MBHzG7mY>
- Educación Activa. (2020a, mayo 7). 03 VB.NET Variables, Constantes y Tipos de Datos [Vídeo]. YouTube. <https://www.youtube.com/watch?v=YMoXyML4IAo>

- Instituto de Informatica UACH. (2021, 24 agosto). 01 Estructuras de Control y Funciones [Vídeo]. YouTube. https://www.youtube.com/watch?v=U5fnK-_DXtM
- Luis Cobián. (2020, 16 junio). [Swift]: Estructuras de control (Ciclos) [Vídeo]. YouTube. <https://www.youtube.com/watch?v=jdWrzgCVrvQ>
- asaeldev - Crypto & Tech. (2015, 15 julio). 2.- Diagramas de flujo | Fundamentos de programación [Vídeo]. YouTube. <https://www.youtube.com/watch?v=awGN89eAni8>
- Daniel PC. (2020, 2 septiembre). 9.- Curso Programación - Pseudocódigo - Diagramas de Flujo en programación [Vídeo]. YouTube. <https://www.youtube.com/watch?v=Rs9Z8x9es30>
- Daniel PC. (2020a, agosto 17). 1.- Curso de Programación - Algoritmos -Temas, Partes del Algoritmo. [Vídeo]. YouTube. <https://www.youtube.com/watch?v=6PvUP-bu1-S0>



Material complementario

- Westreicher, G. (2022, 24 noviembre). Sistema binario. Economipedia. <https://economipedia.com/definiciones/sistema-binario.html>
- Sistema de numeración Octal-Hexadecimal - Tema 2: HARDWARE. (s. f.). <https://sites.google.com/site/tema2informatica1bt/datos-e-informacion/sistema-de-numeracion-octal-hexadecimal>
- Conversiones de sistemas de numeración. (2016, 12 enero). <https://www.ladelec.com/teoria/electronica-digital/343-conversiones-de-sistemas-de-numeracion>
- Operadores aritméticos, lógicos y de comparación. (s. f.). <https://docs.kde.org/stable5/es/kturtle/kturtle/operators.html>
- Tipos de datos, variables y constantes - Programación en C. (s. f.). Solución ingenieril. http://solucioningenieril.com/programacion_en_c/tipos_de_datos_variables_y_constantes
- Qué es un algoritmo informático. (2022, 3 noviembre). OpenWebinars.net. <https://openwebinars.net/blog/que-es-un-algoritmo-informatico/>

Referencias bibliográficas de la unidad



- Cárdenas González, D., Esparza Martínez, M. I., Muñoz Lezama, J. A., Muñoz Roman, O. A., Quezada Mora, S., & Del Vivar Plascencia, V. (2015). Pensamiento lógico computacional.



**ALCALDÍA MAYOR
DE BOGOTÁ D.C.**
SECRETARÍA DE EDUCACIÓN



ATENEA
AGENCIA DISTRITAL PARA LA EDUCACIÓN
SUPERIOR LA CIENCIA Y LA TECNOLOGÍA



**UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS**
Acreditación Institucional de Alta Calidad