**PROGRAMACION NIVEL BASICO**

**Programación Nivel Básico**

**Información General del Bootcamp**

* **Nombre del bootcamp**:
* **Duración total:** (35 sesiones / 105 horas)
* **Horas de dedicación diaria:** 3 horas (1 teoría / 2 laboratorio)
* **Modalidad:** (presencial/virtual)
* **Nivel:** (**principiante**/intermedio/avanzado)
* **Tecnologías y lenguajes cubiertos**: *HTML5, CSS3, JavaScript, Git, Python, SQL, Frameworks.*

**Recomendaciones Generales**

1. La enseñanza debe tener un enfoque práctico y orientado a entornos laborales
2. Se cuenta con 35 sesiones de 3 horas c/u se recomienda como distribución 1 hora de teoría por 2 de práctica
3. Todas las sesiones deben tener al menos 1 laboratorio o caso de estudio
4. Se requiere mantener la atención y motivación de los participantes para evitar la deserción. Utilice estrategias que didácticas adecuadas a la modalidad de enseñanza
5. Cada sesión debe tener objetivos y actividades claras y conclusiones del tema tratado, mantenga un hilo conductor
6. Las sesiones requieren continuidad y un flujo lógico

|  |  |
| --- | --- |
| INSTRUCTIVO | AGREGUE SU CONTENIDO |
| NOMBRE DEL CURSO | Programación nivel Básico |
| OBJETIVO GENERAL DE APRENDIZAJE (*¿Qué se espera que los estudiantes aprendan?* | Proporcionar a los participantes una comprensión integral y práctica del manejo de programación con java, bases de datos relacionales, SQL, HTML5, CSS3, JavaScript y frameworks web. |
| OBJETIVOS ESPECÍFICOS DE APRENDIZAJE (*Actividades que permitan el cumplimiento del objetivo general (al menos 3)*). | * Desarrollar un portal web interactivo con la aplicación práctica de HTML5, CSS3, JavaScript avanzado, utilizando (React, frameworks web). * Utilizar software de control de versiones * Utilizar bases de datos relacionales. * Hacer consultas SQL básicas * Diseñar y hacer uso de estructuras de bases de datos relacionales |
| RESULTADOS ESPERADOS DE APRENDIZAJE *(Describir las habilidades o competencias aprendidas a través del curso)* | Al finalizar esta sesión, los participantes serán capaces de:   * ***Dominio de Lenguajes de Programación:***   *Los estudiantes serán capaces de escribir, compilar y depurar programas utilizando uno o más lenguajes de programación a nivel intermedio (por ejemplo, Python, JavaScript).*   * ***Resolución de Problemas Complejos:***   *Los estudiantes podrán analizar problemas complejos, diseñar soluciones algorítmicas eficientes y aplicarlas mediante el uso de estructuras de datos y algoritmos avanzados.*   * ***Desarrollo de Aplicaciones:***   *Los estudiantes desarrollarán aplicaciones funcionales que integren múltiples tecnologías, como bases de datos, frameworks, demostrando un entendimiento profundo del ciclo de vida del desarrollo de software.*   * ***Manejo de Versiones de Código:***   *Los estudiantes serán competentes en el uso de sistemas de control de versiones (por ejemplo, Git) para gestionar el código fuente, colaborar en proyectos de programación y mantener un historial de cambios detallado.*   * ***Diseño de Interfaces de Usuario:***   *Los estudiantes serán capaces de diseñar y desarrollar interfaces de usuario intuitivas y responsivas, aplicando principios de diseño y usabilidad utilizando HTML, CSS y JavaScript.*   * ***Aplicación de Buenas Prácticas de Programación:***   *Los estudiantes adoptarán y aplicarán buenas prácticas de programación, incluyendo la escritura de código limpio, modular y documentado, así como la realización de pruebas unitarias y de integración.*   * ***Trabajo en Equipo y Colaboración:***   *Los estudiantes desarrollarán habilidades para trabajar efectivamente en equipos de desarrollo, utilizando herramientas de colaboración y comunicación para gestionar proyectos y resolver conflictos.*   * ***Comprensión de Paradigmas de Programación:***   *Los estudiantes obtendrán una comprensión sólida de diferentes paradigmas de programación (orientado a objetos, funcional, entre otros) y aplicarán estos conocimientos en la resolución de problemas específicos.*   * ***Capacidad de Aprendizaje Autónomo:***   *Los estudiantes desarrollarán la habilidad de aprender nuevas tecnologías y lenguajes de programación de manera autónoma, manteniéndose actualizados en un campo en constante evolución.* |
| Mercadeo del bootcamp  (Video máximo 1 minuto que resalte la importancia del bootcamp)  *Ejemplo: Las ventajas del bootcvamp son: (1, 2, 3…)* | **Desarrollo de Habilidades Técnicas**   * Desarrollo de Aplicaciones Web * Comprensión de Estructuras de Datos * Desarrollo y Gestión de Bases de Datos * Programación con Python * Manejo de Versionamiento y Colaboración con Git * Enfoque en la resolución de problemas reales |

|  |  |
| --- | --- |
| **DESARROLLO DE LA CLASE** | |
| **INTRODUCCIÓN Y PRESENTACIÓN**   * **Presentación del instructor** | **Presentación del Instructor:**  Hola, mi nombre es Pablo Enrique Carreño Hernandez, soy Ingeniero de sistemas con énfasis en telecomunicaciones, especialista en multimedia para la docencia, magíster en Informática aplicada a la educación, tengo más de 20 años de experiencia trabajando en el sector público y privado como docente universitario y como Ingeniero de Desarrollo de aplicaciones y sistemas de TI , actualmente soy docente en el programa de Ingeniería de Sistemas en la Universidad Libre y Desarrollador Full-Stack en el servicio público de empleo, Me apasiona la programación, el internet de las cosas y la inteligencia artificial.  **¿Sabías que?**  ¿Sabías qué? HTML no es un lenguaje de programación porque no tiene la capacidad de realizar cálculos o tomar decisiones lógicas. HTML solo describe la estructura y el contenido de una página web a través de etiquetas. Aunque HTML tiene reglas gramaticales definidas y es capaz de definir la estructura de una página web, no tiene la capacidad para escribir instrucciones o algoritmos y no tiene control sobre la computadora para obtener datos o ejecutar tareas. |
|  |
|  |
| **DESARROLLO DEL MÓDULO EN SESIONES** (*¿Cuáles son los módulos para lograr el objetivo de aprendizaje, cuáles son sus subtemas y/o ideas fuerza?*) | **Modulo 1:**   1. Introducción 2. Propósito de la sesión 3. Objetivos de aprendizaje 4. Objetivos de aprendizaje 5. Definición y Conceptos 6. Qué es la programación 7. Conceptos básicos de programación 8. Tipos de lenguajes de programación 9. Aplicaciones de la programación 10. Importancia de la programación en la actualidad 11. Habilidades necesarias para la programación 12. Pasos para comenzar a programar 13. ¿Qué es HTML? 14. Profundización en Conceptos Claves 15. Preguntas Orientadoras 16. Título del Laboratorio: Creando tu Primera Página Web Interactiva 17. Conclusiones 18. Resumen |
| **Nombre del módulo** | * **Nombre del módulo**: Conceptos básicos de programación |
| **Introducción:** | La programación es una habilidad fundamental en el mundo moderno, un lenguaje que nos permite interactuar con las máquinas e instruirlas para realizar tareas complejas. Es el arte de traducir ideas abstractas en instrucciones comprensibles para las computadoras. A través de la programación, creamos software, aplicaciones web, juegos y una miríada de herramientas que hacen que nuestras vidas sean más eficientes, divertidas y conectadas. |
| **Clase 1.** | **1.1 Propósito de la sesión**  En esta sesión, exploraremos conceptos básicos de programación y conoceremos el HTML, proporcionando una base sólida para mejorar la funcionalidad y accesibilidad de tus proyectos web.  **1.2 Objetivos de aprendizaje**   * Comprender los conceptos fundamentales de la programación * Desarrollar habilidades para escribir y comprender código * Aplicar conceptos de programación en proyectos prácticos. * Conocer el HTML     **2. Definición y Conceptos**    **2.1 Qué es la programación**  La programación es el proceso de diseñar, escribir, probar y depurar código fuente para crear programas de software. Se trata de un proceso creativo que implica la resolución de problemas, la lógica y el pensamiento computacional. El código fuente se escribe en un lenguaje de programación específico que las computadoras pueden entender y ejecutar.   * **Lógica y algoritmos:** Para programar, es esencial tener una comprensión básica de la lógica y los algoritmos. Un algoritmo es una secuencia de pasos bien definidos que resuelven un problema específico. La programación consiste en traducir estos algoritmos en código. * **Instrucciones para las máquinas:** en esencia, la programación consiste en escribir instrucciones detalladas para las computadoras. Estas instrucciones, escritas en un lenguaje específico que la máquina entiende, le dicen qué hacer, cómo hacerlo y en qué orden. * **Creación de software y aplicaciones:** A través de la programación, podemos crear programas de computadora, aplicaciones web, juegos, sistemas operativos y una gran variedad de herramientas digitales que usamos a diario.   **2.2 Conceptos básicos de programación**   * **Algoritmo**: Un algoritmo es una secuencia de pasos bien definidos que resuelven un problema específico. Es la base de la programación, ya que describe cómo se debe realizar una tarea. * **Variables:** Las variables son contenedores que almacenan datos. Pueden contener información de diferentes tipos, como números, texto o incluso listas de elementos. * **Operadores:** Los operadores permiten realizar operaciones matemáticas, lógicas o de comparación entre variables y datos. * **Sentencias:** Las sentencias son instrucciones que la computadora ejecuta en un orden específico. Cada sentencia realiza una acción, como asignar un valor a una variable o mostrar un mensaje en la pantalla. * **Estructuras de control:** Las estructuras de control determinan el flujo de ejecución de un programa. Permiten tomar decisiones, ejecutar código repetidamente y controlar el orden de las operaciones. Las estructuras de control más comunes son:   + *Sentencia IF: Ejecuta un bloque de código si se cumple una condición.*   + *Sentencia FOR: Repite un bloque de código un número determinado de veces.*   + *Sentencia WHILE: Repite un bloque de código mientras se cumple una condición.* * **Tipos de datos:** Los tipos de datos determinan el tipo de información que puede contener una variable. Algunos tipos de datos comunes son:   + *Entero: Números enteros, como 1, 2, 3.*   + *Decimal: Números con decimales, como 1.5, 2.75.*   + *Texto: Cadenas de caracteres, como "Hola" o "Programación".*   + *Booleano: Valores verdaderos o falsos, como True o False.* * **Operaciones con variables:** Las variables se pueden usar en operaciones matemáticas, lógicas y de comparación. Por ejemplo, se puede sumar el contenido de dos variables o comparar si una variable es mayor que otra. * **Funciones:** Las funciones son bloques de código que realizan una tarea específica. Se les asigna un nombre y pueden recibir parámetros como entrada. Las funciones retornan un valor como salida. * **Subrutinas**: Las subrutinas son similares a las funciones, pero no retornan un valor. Se usan para realizar tareas específicas dentro del programa principal. Beneficios de usar funciones y subrutinas:   + *Reusabilidad: Se pueden reutilizar en diferentes partes del programa*   + *Organización: Mejoran la organización y la legibilidad del código*   + *Modularidad: Permiten dividir el programa en partes más pequeñas y manejables.*   **Tipos de lenguajes de programación**  Existen diferentes tipos de lenguajes de programación, cada uno con sus características y aplicaciones específicas. Algunos ejemplos son:    **Aplicaciones de la programación**   * **Desarrollo Web:** La programación es esencial para crear sitios web interactivos, aplicaciones web, plataformas de comercio electrónico y una variedad de servicios en línea. Crear sitios web, aplicaciones web y plataformas online. * **Desarrollo de aplicaciones móviles:** Con la programación, se crean aplicaciones móviles para teléfonos inteligentes y tabletas, abarcando áreas como juegos, redes sociales, compras y entretenimiento. Crear aplicaciones para teléfonos inteligentes y tablets. * **Desarrollo de juegos:** La programación es crucial para crear juegos de computadora y videojuegos, desde juegos simples hasta títulos complejos de AAA. * **Ciencia de datos y aprendizaje automático**: Analizar datos, crear modelos predictivos y desarrollar sistemas inteligentes. * **Robótica e inteligencia artificial**: Controlar robots, desarrollar sistemas de IA y automatizar tareas complejas. * **Ciberseguridad:** Proteger sistemas informáticos, redes y datos de amenazas cibernéticas.   **Importancia de la programación en la actualidad**  La programación es cada vez más importante en la era digital. El mundo se está volviendo cada vez más automatizado y conectado, y la programación es la clave para impulsar esta transformación. La programación es esencial para:   * **Innovación**: Crear nuevas soluciones y tecnologías para mejorar la vida humana. La programación es el motor de la innovación tecnológica. Permite crear soluciones a los problemas del mundo real, desde la automatización de tareas hasta la creación de nuevos sistemas de comunicación. * **Automatización**: Optimizar procesos, reducir errores humanos y aumentar la eficiencia. * **Conectividad**: Crear plataformas y servicios que conecten a las personas y la información. * **Oportunidades** **laborales**: Ofrece una gran demanda de profesionales calificados en el mercado laboral. * **Crecimiento económico:** La industria de la tecnología y la información es una de las más dinámicas y de rápido crecimiento. Los programadores son altamente demandados y bien remunerados. * **Resolución de problemas:** La programación desarrolla habilidades de pensamiento crítico, lógica y resolución de problemas. Es una habilidad valiosa en cualquier campo profesional. * **Creatividad y expresión:** La programación es una forma de expresión creativa. Permite a las personas dar vida a sus ideas y construir nuevas realidades digitales.   **Habilidades necesarias para la programación**  Para convertirse en un programador exitoso, se requieren habilidades técnicas y de pensamiento crítico. Algunas de las habilidades más importantes incluyen:   * **Pensamiento lógico:** La capacidad de descomponer problemas complejos en pasos más pequeños y solucionarlos de forma sistemática * **Habilidades de resolución de problemas:** La capacidad de identificar problemas, analizarlos y encontrar soluciones creativas. * **Dominio de lenguajes de programación:** La capacidad de escribir código en diferentes lenguajes de programación y entender cómo funciona el código. * **Conocimiento de estructuras de datos y algoritmos:** La capacidad de organizar datos de manera eficiente y desarrollar algoritmos para resolver problemas.   **Pasos para comenzar a programar**  Si estás interesado en aprender a programar, puedes empezar con estos pasos:   * **Elige un lenguaje de programación:** Hay muchos lenguajes de programación disponibles, elige uno que sea adecuado para tus intereses y objetivos. Algunos lenguajes populares son Python, Java, JavaScript y C++. * **Aprende los conceptos básicos:** Estudia los conceptos básicos de la programación, como variables, operadores, estructuras de control y funciones. Hay muchos recursos disponibles en línea y en libros. * **Practica regularmente:** La práctica es fundamental para aprender a programar. Elige proyectos pequeños para empezar y ve aumentando la complejidad gradualmente. Hay muchos desafíos y proyectos en línea que puedes probar. * **Únete a una comunidad de programadores:** Busca comunidades en línea o grupos de programación locales donde puedes compartir conocimientos, obtener ayuda y colaborar con otros programadores. * **Mantente actualizado:** La tecnología evoluciona rápidamente, por lo que es importante mantenerte actualizado con las últimas tendencias y herramientas en programación.   **¿Qué es HTML?**  El documento que lee el navegador está escrito en un lenguaje de marcado llamado HTML, que son las siglas de HyperText Markup Language (Lenguaje de marcas de hipertexto), o lo que es lo mismo, un lenguaje de etiquetas que permite incluir o hacer referencia a todo tipo de información.  HTML no es un lenguaje de programación porque no tiene la capacidad de realizar cálculos o tomar decisiones lógicas. HTML solo describe la estructura y el contenido de una página web a través de etiquetas. Aunque HTML tiene reglas gramaticales definidas y es capaz de definir la estructura de una página web, no tiene la capacidad para escribir instrucciones o algoritmos y no tiene control sobre la computadora para obtener datos o ejecutar tareas.  Estructura básica de un documento HTML    **El DOCTYPE (tipo de documento):** es una etiqueta especial que se escribe en la primera línea del documento HTML. Debe ir especificado siempre para que el navegador sepa de que tipo de documento HTML se trata.  **La etiqueta head (metadatos):** Cuando hablamos de la etiqueta <head> de un documento HTML, muchas veces la denominamos cabecera HTML.  **La etiqueta body (cuerpo**): La otra parte principal de un documento HTML es la etiqueta <body>. Todos los elementos visuales de una página se encuentran en el interior de la etiqueta <body>, por lo que es una de las partes más importantes de una página web.    **Profundización en Conceptos Claves**     * **Un IDE:** (Entorno de Desarrollo Integrado) es un sistema de software para el diseño de aplicaciones que combina herramientas comunes para desarrolladores en una sola interfaz de usuario gráfica (GUI). Normalmente, un IDE consiste en un editor de código fuente, herramientas de construcción automáticas y un depurador. La mayoría de los IDE tienen auto-completado inteligente de código (IntelliSense). Algunos IDE contienen un compilador, un intérprete, o ambos. Ejemplos: Visual Studio Code, Notepad++, Sublime Text, Eclipse, otros. * **La interfaz de usuario, IU** (del inglés User Interface, UI), es el medio que permite la comunicación entre un usuario y una máquina, equipo, computadora o dispositivo, y comprende todos los puntos de contacto entre el usuario y el equipo. Normalmente suelen ser fáciles de entender y fáciles de accionar, aunque en el ámbito de la informática es preferible referirse a que suelen ser "usables", "amigables e intuitivos" porque es compleja. * El **código fuente:** de un programa informático (o software) es un conjunto de líneas de texto con los pasos que debe seguir la computadora para ejecutar un cargador * **Un compilador:** es un pequeño programa informático, que se encarga de traducir (compilar) el código fuente de cualquier aplicación que se esté desarrollando. * **Un intérprete**: es un programa que convierte y ejecuta código escrito en lenguajes de programación. A diferencia de un compilador, que traduce el código de un lenguaje de programación a código de máquina para su posterior ejecución, el intérprete ejecuta el código directamente. |
| **Preguntas de Control**  Al finalizar la sesión elabore dos o tres preguntas de tal forma que se pueda constatar que se transmitieron los conocimientos esenciales o básicos del tema (Retención de conocimiento). Selección múltiple | **Preguntas Orientadoras**  Preguntas de Control y Respuestas  Pregunta 1:  ¿Cuál es la estructura básica de un documento HTML? Describe las etiquetas principales.  Respuesta:  *La estructura básica de un documento HTML incluye las siguientes etiquetas principales:*  *- `<!DOCTYPE html>`: Declara el tipo de documento y la versión de HTML.*  *- `<html>`: La raíz del documento HTML.*  *- `<head>`: Contiene metadatos sobre el documento, como el título y enlaces a hojas de estilo.*  *- `<title>`: Define el título del documento que aparece en la pestaña del navegador.*  *- `<body>`: Contiene el contenido visible de la página web, como texto, imágenes, enlaces, etc.*  Pregunta 2:  ¿Qué es una variable en programación y cómo se declara en JavaScript?  Respuesta:  *Una variable es un contenedor para almacenar datos que pueden cambiar durante la ejecución del programa. En JavaScript, una variable se declara utilizando las palabras clave `var`, `let` o `const`. Ejemplo:*  *```javascript*  *let nombre = 'Juan';*  *const edad = 30;*  *```*  Pregunta 3:  ¿Cómo se utiliza una estructura condicional `if-else` en JavaScript? Proporciona un ejemplo.  Respuesta:  *Una estructura condicional `if-else` se utiliza para ejecutar un bloque de código si una condición es verdadera y otro bloque si la condición es falsa. Ejemplo:*  *```javascript*  *let edad = 18;*  *if (edad >= 18) {*  *console.log('Eres mayor de edad.');*  *} else {*  *console.log('No eres mayor de edad.');*  *}*  *```*  Pregunta 4:  ¿Qué es un bucle `for` en JavaScript y cómo funciona? Proporciona un ejemplo.  Respuesta:  *Un bucle `for` se utiliza para ejecutar un bloque de código un número específico de veces. Se compone de tres partes: inicialización, condición y actualización. Ejemplo:*  *```javascript*  *for (let i = 0; i < 5; i++) {*  *console.log('Este es el mensaje número ' + i);*  *}*  *```*  *En este ejemplo, el bucle se ejecutará cinco veces, mostrando un mensaje cada vez.*  Pregunta 5:  ¿Qué es el DOM (Document Object Model) y cómo se puede acceder a un elemento HTML usando JavaScript? Proporciona un ejemplo.  Respuesta:  *El DOM (Document Object Model) es una interfaz de programación para documentos HTML y XML. Representa la estructura de un documento como un árbol de nodos. Se puede acceder a los elementos HTML y modificarlos usando JavaScript. Ejemplo:*  *```javascript*  *// Acceder a un elemento por su ID*  *let elemento = document.getElementById('miElemento');*  *// Modificar el contenido del elemento*  *elemento.innerHTML = 'Nuevo contenido';*  Estas preguntas y respuestas cubren conceptos fundamentales de programación y HTML que son esenciales para un curso básico, asegurando que los estudiantes entiendan la estructura del HTML, la declaración y uso de variables, condicionales, bucles, y la manipulación del DOM con JavaScript. |
| Laboratorio 1 | Título del Laboratorio: Creando tu Primera Página Web Interactiva  Duración: 2 horas  Objetivos del Laboratorio:   * 1. *Comprender los conceptos básicos de programación: Variables, condicionales, bucles, funciones.*   2. *Entender los fundamentos de HTML: Estructura básica, etiquetas comunes.*   3. *Aplicar conocimientos de HTML y programación para crear una página web.*   4. *Familiarizarse con un entorno de desarrollo integrado (IDE)*   Materiales Necesarios:   1. *Computadora con acceso a internet.* 2. *Entorno de desarrollo integrado (IDE): Recomendado Visual Studio Code o cualquier otro de preferencia.* 3. *Documentación y recursos de referencia: Manuales y tutoriales en línea.* 4. *Ejemplos de código proporcionados durante el curso (carpeta archivos de ayuda).*   Ayudas   1. *Guía de instalación del IDE.* 2. *Ejemplos de código básico: Variables, condicionales, bucles, funciones en JavaScript; etiquetas y estructura básica en HTML.* 3. *Manual de uso del IDE.* 4. *Hoja de referencia rápida: Sintaxis básica y comandos útiles de HTML y JavaScript.*   Estructura del Laboratorio:  Introducción (15 minutos):  *- Presentación del laboratorio y objetivos.*  *- Explicación del proyecto a desarrollar.*      Desarrollo (70 minutos):  - Paso 1: Instalación del IDE (si no está previamente instalado).  - Paso 2: Configuración del entorno y familiarización con el IDE.  - Paso 3: Creación de una nueva página web:  *- HTML: Estructura básica del documento (`<html>`, `<head>`, `<body>`).*  *- JavaScript: Programación básica para interacción.*  *- CSS (opcional): Estilización básica.*    Revisión y Ajustes (20 minutos):  - Revisión del código con el instructor o compañeros.  - Pruebas de la página web y corrección de errores.  Conclusión (10 minutos)  - Presentación de los resultados.  - Retroalimentación del instructor y discusión de mejoras. |
| ***Conclusión*** | Conclusión de la Clase 1: Conceptos básicos de programación y HTML   1. *Evaluación del entendimiento de los conceptos básicos de programación y HTML.* 2. *Revisión del código desarrollado por los estudiantes.* 3. *Discusión sobre las dificultades encontradas y cómo fueron resueltas.* 4. *Sugerencias y retroalimentación para mejorar futuras prácticas de programación y desarrollo web.*   Resumen de la clase:  Esta sesión ha proporcionado una visión general de los conceptos básicos de programación y HTML, estableciendo un marco conceptual sólido para las sesiones futuras. A medida que avancemos en el curso, profundizaremos en cada uno de estos aspectos, explorando las tecnologías subyacentes, las mejores prácticas de implementación y los casos de uso en el mundo real.  Este laboratorio permitió a los estudiantes aplicar los conceptos básicos de programación y HTML aprendidos durante el curso, familiarizándose con el proceso de desarrollo web desde cero, incluyendo la instalación de herramientas, escritura de código, pruebas y depuración. |
| **Material descargable: (Realice guías para cumplir la sesión)**Para complementar los temas desarrollados usted aporta material complementario de fácil comprensión. | Carpeta: Archivos de ayuda  Ejemplo 1: index.html – script.js  Ejemplo 2: index.html – script.js |

|  |  |
| --- | --- |
| MATERIAL COMPLEMENTARIO Y KIT DE HERRAMIENTAS | Material de fácil comprensión para complementar los temas abordados o sugeridos para quienes quieran profundizar.  Ver carpeta Archivos de ayuda |
| Material de apoyo | A qué sesión corresponde |
| Archivos PDF |  |
| Archivos en Word | Modulo 1 - Laboratorio 1.docx |
| Archivos en Excel |  |
| Presentaciones | 1. Modulo 1 - Conceptos básicos de programación.pptx |
| Libros |  |
| Links |  |
| Videos |  |
| Otros | Ejemplo1: Index.html, script.js, Ejemplo 2: Index.html, script.js |