1. IDENTIFICACIÓN DE LA GUÍA DE APRENDIZAJE

- Denominación del Programa de Formación: ANALISIS Y DESARROLLO DE SOFTWARE
- Código del Programa de Formación: 233104
- Nombre del Proyecto Formativo (si aplica):
- Fase del Proyecto (si aplica): Análisis
- Actividad de Proyecto Formativo (si aplica): Interpretar los requerimientos de la solución de software aplicando el pensamiento lógico y la construcción de algoritmos.
- Competencia: Implementar la solución de software de acuerdo con los requisitos de operación y modelos de referencia.
- Duración de la Guía de Aprendizaje: 40 horas

2. PRESENTACIÓN

Apreciado aprendiz, bienvenido al punto de partida de su formación como desarrollador de software: el pensamiento lógico y la algoritmia. Antes de escribir código en cualquier lenguaje de programación, es necesario aprender a pensar de manera estructurada, a descomponer problemas, a identificar patrones y a expresar soluciones mediante pasos precisos y ordenados. Esa es, precisamente, la esencia de un algoritmo.

Esta guía de aprendizaje, con una duración de 40 horas, lo llevará por un recorrido progresivo: iniciará ejercitando el razonamiento lógico-deductivo con secuencias, patrones y acertijos; luego construirá el concepto de algoritmo y sus características; desarrollará la habilidad de dar instrucciones precisas y sin ambigüedad; conocerá las variables, las constantes y los tipos de datos, dando sus primeros pasos en JavaScript; y finalmente aplicará todo lo anterior resolviendo un banco de diez problemas de complejidad creciente que integran condicionales, contadores y acumuladores.

Las actividades se apoyan en dos materiales anexos que acompañan esta guía: el “Plan de Actividades Nivel Básico — Introducción a la Algoritmia” (con fichas de trabajo, lecturas, guías paso a paso y rúbricas de evaluación) y el banco de “Problemas de Algoritmos: variables, condicionales, contadores y acumuladores”. Desarrolle cada actividad de forma consciente: el pensamiento algorítmico que construya aquí será la base de toda su vida profesional como programador. ¡Éxitos!

3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

Descripción de la(s) Actividad(es)

3.1 Actividades de reflexión inicial

Descripción de la actividad: el instructor actuará como moderador presentando situaciones cotidianas que esconden algoritmos: seguir una receta de cocina, la ruta que sugiere el GPS, las canciones que recomienda una plataforma de música o los pasos para retirar dinero de un cajero automático. Posteriormente se exhorta al aprendiz a reflexionar y socializar sus impresiones acerca de las siguientes preguntas:

- ¿Qué pasos sigue usted, sin darse cuenta, desde que suena la alarma hasta que sale de su casa? ¿Podría escribirlos en orden y sin ambigüedad?

- ¿En qué se parece darle indicaciones a una persona que no conoce la ciudad a darle instrucciones a un computador?
- ¿Por qué cree que se dice que “el computador solo hace exactamente lo que se le ordena”?

Ambiente requerido: Convencional (aula) o ambiente de formación TIC.

Estrategias o técnicas didácticas activas: Mesa redonda y lluvia de ideas.

Materiales de formación: Video beam, computador, libretas, bolígrafos.

Material de apoyo: Video introductorio “¿Qué es un algoritmo?” y ejemplos cotidianos presentados por el instructor.

Duración de la actividad: 2 horas.

3.2 Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje

Descripción de la actividad: usted va a identificar los conocimientos y experiencias previas que posee sobre lógica y resolución de problemas. Resuelva individualmente el siguiente cuadro de saberes previos y luego socialice sus respuestas para establecer, a nivel grupal, los saberes comunes y los aspectos que requieren profundización:

Pregunta	Respuesta
¿Qué entiende usted por algoritmo? Dé un ejemplo de su vida cotidiana.	
Complete la secuencia y explique la regla: 2 – 4 – 8 – 16 – ___	
¿Ha escrito alguna vez instrucciones para otra persona (una receta, una guía)? ¿Qué dificultades tuvo?	
¿Ha tenido contacto con algún lenguaje de programación o herramienta como Scratch, PSeInt o JavaScript?	

Ambiente requerido: Convencional.

Estrategias o técnicas didácticas activas: Cuadro descriptivo y socialización grupal.

Materiales de formación: Hojas en blanco, lápices.

Material de apoyo: Conocimientos previos del aprendiz.

Duración de la actividad: 2 horas.

3.3 Actividades de apropiación del conocimiento

Descripción de la actividad: a continuación se desarrollarán las actividades requeridas para la apropiación del pensamiento lógico y la algoritmia, apoyadas en los materiales anexos. Duración total: 28 horas.

3.3.1 Despertando el pensamiento lógico: secuencias y patrones

El aprendiz ejercitará el razonamiento lógico-deductivo como base cognitiva del pensamiento algorítmico. Utilizando la Ficha de Acertijos Lógicos (Recurso 1.1 del material anexo "Plan de Actividades Nivel Básico"), desarrollará:

- Parte A — Secuencias numéricas: identificar el patrón, completar cuatro secuencias y explicar la regla de cada una.
- Parte B — Acertijos lógicos situacionales: resolver los tres acertijos (posiciones en la fila, contenido de las cajas y tabla de verdad de la lámpara con interruptores), justificando el razonamiento paso a paso.
- Parte C — Crear dos acertijos lógicos originales con solución verificable.
- Responder las preguntas de reflexión escrita (Recurso 1.2): estrategias utilizadas, semejanza entre resolver acertijos y dar instrucciones a un computador, y relación entre pensamiento lógico y programación.
- Socializar en plenaria las estrategias de solución empleadas.

Ambiente requerido: Convencional o ambiente de formación TIC.

Estrategias o técnicas didácticas activas: Resolución de acertijos y aprendizaje basado en problemas.

Materiales de formación: Ficha impresa o digital, lápices, computador.

Material de apoyo: Material anexo "Plan de Actividades Nivel Básico" — Actividad 1 (Recursos 1.1, 1.2 y 1.3).

Duración de la actividad: 4 horas.

Evidencias de aprendizaje: Ficha de acertijos resuelta con justificaciones y dos acertijos originales.

Instrumentos de evaluación: Rúbrica de evaluación (Recurso 1.3 del material anexo).

3.3.2 ¿Qué es un algoritmo? Concepto, características y tipos

A partir de la lectura base "¿Qué es un algoritmo?" (Recurso 2.1 del material anexo), el aprendiz identificará la definición, las cuatro características (finito, preciso, definido y general) y los tres tipos según su estructura de control (secuencial, condicional e iterativo). Posteriormente desarrollará:

- Construcción de un mapa mental en MindMeister, Coggle o Canva siguiendo la guía paso a paso (Recurso 2.2), con cinco ramas: definición, características, tipos, ejemplos tecnológicos y aplicación en su contexto.
- En cada tipo de algoritmo, proponer un ejemplo de la vida cotidiana diferente a los de la lectura.
- Redacción del párrafo de reflexión de mínimo ocho líneas (Recurso 2.3), integrando: la característica más importante, la rama más difícil de construir y un ejemplo propio de algoritmo cotidiano.
- Socialización de los mapas mentales en plenaria.

Ambiente requerido: Ambiente de formación TIC con acceso a internet.

Estrategias o técnicas didácticas activas: Lectura dirigida y mapa mental.

Materiales de formación: Computadores, acceso a internet, herramienta de mapas mentales (MindMeister, Coggle o Canva).

Material de apoyo: Material anexo "Plan de Actividades Nivel Básico" — Actividad 2 (Recursos 2.1 a 2.4).

Duración de la actividad: 4 horas.

Evidencias de aprendizaje: Mapa mental exportado en PDF o imagen y párrafo de reflexión.

Instrumentos de evaluación: Rúbrica de evaluación (Recurso 2.4 del material anexo).

3.3.3 Instrucciones precisas: base del pensamiento algorítmico

El aprendiz desarrollará la capacidad de expresar procesos de forma clara, ordenada y sin ambigüedad. Siguiendo la Actividad 3 del material anexo, el ejercicio se realiza en parejas: un aprendiz escribe las instrucciones y el otro actúa como "ejecutor", siguiéndolas literalmente como si fuera un computador:

- Elegir una tarea de la lista disponible (Recurso 3.1): doblar un avión de papel, dibujar una casa, preparar un vaso de agua con azúcar u ordenar cinco números de mayor a menor.
- Escribir la Versión 1 de las instrucciones (mínimo diez pasos) en la hoja de registro (Recurso 3.2).
- El compañero ejecuta las instrucciones al pie de la letra; se registran mínimo tres problemas o ambigüedades detectadas.
- Escribir la Versión 2 corregida, eliminando las ambigüedades encontradas.
- Diligenciar el análisis de conexión con algoritmos (Recurso 3.3): comparación de versiones, ejemplos de instrucciones vagas frente a precisas y semejanza con escribir un programa.

Ambiente requerido: Convencional.

Estrategias o técnicas didácticas activas: Juego de roles (programador-ejecutor) y taller práctico.

Materiales de formación: Hojas de registro, papel, lápices, materiales de la tarea elegida.

Material de apoyo: Material anexo "Plan de Actividades Nivel Básico" — Actividad 3 (Recursos 3.1 a 3.4).

Duración de la actividad: 4 horas.

Evidencias de aprendizaje: Hoja de registro con las dos versiones de instrucciones, problemas detectados y análisis de conexión.

Instrumentos de evaluación: Rúbrica de evaluación (Recurso 3.4 del material anexo).

3.3.4 Variables y constantes: cajas con nombres — primeros pasos en JavaScript

A partir de la lectura "Variables y constantes en JavaScript" (Recurso 4.1 del material anexo), el aprendiz comprenderá los conceptos de variable (let), constante (const) y los tipos de datos number, string, boolean y undefined. Desarrollará:

- Parte A — Clasificación de tipos de datos sin computador: clasificar ocho datos según su tipo.
- Parte B — Ejercicios en Visual Studio Code (Recurso 4.2): crear el archivo variables.js y resolver los cinco ejercicios (declaración de variables, constantes y el error al modificarlas, variable sin valor, datos de un estudiante con console.log y el reto de concatenación), comentando cada línea de código y verificando los resultados en la consola del navegador (F12).
- Construir el mapa mental "Tipos de datos en JavaScript" con las cinco ramas indicadas en la guía (Recurso 4.3): number, string, boolean, undefined y let vs const.
- Publicar el archivo variables.js en el repositorio GitHub y el mapa mental en la plataforma indicada por el instructor.

Ambiente requerido: Ambiente de formación TIC.

Estrategias o técnicas didácticas activas: Taller práctico de codificación y mapa mental.

Materiales de formación: Computadores con Visual Studio Code, navegador con DevTools, acceso a internet, cuenta de GitHub.

Material de apoyo: Material anexo “Plan de Actividades Nivel Básico” — Actividad 4 (Recursos 4.1 a 4.4).

Duración de la actividad: 6 horas.

Evidencias de aprendizaje: Archivo variables.js publicado en GitHub, hoja de trabajo diligenciada y mapa mental de tipos de datos.

Instrumentos de evaluación: Rúbrica de evaluación (Recurso 4.4 del material anexo).

3.3.5 Taller de problemas: condicionales, contadores y acumuladores

El aprendiz aplicará los conceptos apropiados resolviendo el banco de problemas del material anexo “Problemas de Algoritmos: variables, condicionales, contadores y acumuladores”, que contiene diez problemas organizados en tres niveles de complejidad. Cada solución debe presentarse en pseudocódigo (o diagrama de flujo en PSeInt) y, para los problemas básicos y medios, también codificada en JavaScript:

- Nivel básico (problemas 01 a 03): Clasificador de Temperatura, Mayor de Tres Números y Cajero de Supermercado — variables, condicionales simples y compuestos, primer uso de acumuladores.
- Nivel medio (problemas 04 a 07): Contador de Aprobados y Reprobados, Control de Inventario de Tienda, Análisis de Edades por Rango y Simulador de Semáforo Inteligente — contadores múltiples, acumuladores, máximos y mínimos, condicionales anidados.
- Nivel avanzado (problemas 08 a 10): Sistema de Calificación con Bonificación, Planilla de Nómina Simplificada y Análisis de Ventas Mensuales — integración de todos los conceptos.
- Para cada problema: identificar las variables sugeridas, construir la solución, probarla con los ejemplos de entrada/salida del material y documentar los casos de prueba.
- Socialización: cada equipo sustenta ante el grupo la solución de un problema asignado por el instructor, explicando la traza de sus variables.

Ambiente requerido: Ambiente de formación TIC.

Estrategias o técnicas didácticas activas: Aprendizaje basado en problemas, prueba de escritorio (traza de variables) y trabajo colaborativo.

Materiales de formación: Computadores con PSeInt y Visual Studio Code, hojas para pruebas de escritorio.

Material de apoyo: Material anexo “Problemas de Algoritmos: variables, condicionales, contadores y acumuladores” (problemas 01 a 10 con variables sugeridas y ejemplos de entrada/salida).

Duración de la actividad: 10 horas.

Evidencias de aprendizaje: Portafolio con las soluciones de los diez problemas (pseudocódigo, código JavaScript y casos de prueba) y sustentación de la solución asignada.

Instrumentos de evaluación: Lista de chequeo y prueba de escritorio verificada por el instructor.

3.4 Actividades de transferencia del conocimiento

Descripción de la actividad: Proyecto integrador “Mi propio banco de problemas”. En parejas, los aprendices asumirán el rol de diseñadores de ejercicios de programación para transferir lo aprendido a una situación real de su entorno:

- Identificar dos situaciones reales de su comunidad o entorno laboral que puedan resolverse con un algoritmo (por ejemplo: control de asistencia, caja de una tienda, préstamo de implementos deportivos, recaudo de una ruta de transporte).
- Redactar cada situación como un problema formal siguiendo el formato del banco anexo: enunciado, conceptos involucrados, variables sugeridas y ejemplo de entrada/salida.
- Construir la solución completa de cada problema: pseudocódigo o diagrama de flujo, código en JavaScript y prueba de escritorio con al menos dos casos.
- Un problema debe usar condicionales y el otro debe integrar contadores o acumuladores.
- Intercambiar los problemas con otra pareja, resolver los recibidos y retroalimentar la claridad de los enunciados.
- Sustentar el proyecto ante el grupo en una presentación de máximo diez minutos.

Ambiente requerido: Ambiente de formación TIC.

Estrategias o técnicas didácticas activas: Aprendizaje basado en proyectos, evaluación entre pares y sustentación.

Materiales de formación: Computadores, PSeInt, Visual Studio Code, video beam.

Material de apoyo: Materiales anexos de la guía y talleres desarrollados en las actividades de apropiación.

Evidencias de aprendizaje: Documento del proyecto integrador (dos problemas formulados con su solución completa y pruebas de escritorio) y sustentación.

Instrumentos de evaluación: Rúbrica de proyecto y lista de chequeo de sustentación.

Duración de la actividad: 8 horas.

4. PLANTEAMIENTO DE EVIDENCIAS DE APRENDIZAJE PARA LA EVALUACIÓN EN EL PROCESO FORMATIVO

Fase del proyecto formativo	Actividad del proyecto formativo	Actividad de Aprendizaje	Evidencias de Aprendizaje	Criterios de Evaluación	Técnicas e Instrumentos de Evaluación
Análisis	Interpretar los requerimientos de la solución de software aplicando el pensamiento lógico y la construcción de algoritmos.	Aplicar el razonamiento lógico y los conceptos de algoritmo, instrucciones precisas, variables y tipos de datos.	De conocimiento: ficha de acertijos, mapa mental de algoritmo y clasificación de tipos de datos. De desempeño: hoja de registro de instrucciones precisas y ejercicios en variables.js	Identifica patrones y relaciones lógicas justificando su razonamiento. Describe las características y tipos de algoritmos con ejemplos propios. Declara y utiliza variables, constantes y	Rúbricas de evaluación del material anexo (Recursos 1.3, 2.4, 3.4 y 4.4).

			publicados en GitHub.	tipos de datos en JavaScript.	
Análisis	Interpretar los requerimientos de la solución de software aplicando el pensamiento lógico y la construcción de algoritmos.	Resolver problemas del contexto mediante algoritmos con condicionales, contadores y acumuladores.	De producto: portafolio con las soluciones de los diez problemas del banco anexo y documento del proyecto integrador con dos problemas propios formulados, resueltos y sustentados.	Construye algoritmos que resuelven correctamente los problemas propuestos verificándolos con casos de prueba. Aplica condicionales, contadores y acumuladores según la necesidad del problema. Sustenta con propiedad la traza de sus soluciones.	Lista de chequeo. Prueba de escritorio. Rúbrica de proyecto y de sustentación.

5. GLOSARIO DE TÉRMINOS

Acumulador: variable que suma o agrega valores sucesivamente durante la ejecución de un algoritmo (por ejemplo, total ← total + precio).

Algoritmo: conjunto de instrucciones ordenadas, finitas y precisas que permiten resolver un problema o realizar una tarea.

Condicional: estructura de control que evalúa una condición para decidir qué camino de ejecución seguir (SI … SINO).

Constante: espacio de memoria con nombre cuyo valor no puede cambiar durante la ejecución; en JavaScript se declara con const.

Contador: variable que se incrementa en una cantidad fija (normalmente 1) cada vez que ocurre un evento (por ejemplo, aprobados ← aprobados + 1).

Pensamiento lógico: capacidad de razonar de manera ordenada identificando patrones, secuencias y relaciones causales para llegar a conclusiones válidas.

Prueba de escritorio: técnica de verificación manual de un algoritmo en la que se sigue paso a paso la ejecución registrando el valor de cada variable.

Pseudocódigo: descripción de un algoritmo en lenguaje natural estructurado, independiente de cualquier lenguaje de programación.

PSeInt: herramienta educativa gratuita para escribir y ejecutar algoritmos en pseudocódigo y diagramas de flujo en español.

Tipo de dato: clasificación del valor que puede almacenar una variable: number (números), string (texto), boolean (true/false) y undefined (sin valor asignado).

Traza: seguimiento del valor que toman las variables de un algoritmo a lo largo de su ejecución.

Variable: espacio de memoria con nombre cuyo valor puede cambiar durante la ejecución; en JavaScript se declara con let.

6. REFERENTES BIBLIOGRÁFICOS

- Material anexo 1: Plan de Actividades Nivel Básico — Unidad 1: Introducción a la Algoritmia (fichas de trabajo, lecturas, guías paso a paso y rúbricas de evaluación).
- Material anexo 2: Problemas de Algoritmos — Variables, Condicionales, Contadores y Acumuladores (banco de 10 problemas con niveles básico, medio y avanzado).
- Joyanes Aguilar, L. (2008). Fundamentos de Programación: Algoritmos, Estructura de Datos y Objetos (4.ª ed.). McGraw-Hill.
- Cairó Battistutti, O. (2005). Metodología de la Programación: Algoritmos, Diagramas de Flujo y Programas (3.ª ed.). Alfaomega.
- MDN Web Docs — JavaScript: https://developer.mozilla.org/es/docs/Web/JavaScript
- PSeInt — intérprete de pseudocódigo: http://pseint.sourceforge.net/
- Visual Studio Code: https://code.visualstudio.com/
- MindMeister: https://www.mindmeister.com/ — Coggle: https://coggle.it/
- Biblioteca digital SENA: http://biblioteca.sena.edu.co/

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)		Instructor		Julio 2026

8. CONTROL DE CAMBIOS (diligenciar únicamente si realiza ajustes a la guía)

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					