PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL

FORMATO GUÍA DE APRENDIZAJE

1. IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE

- Denominación del Programa de Formación: TÉCNICO PROGRAMACIÓN DE SOFTWARE
- Código del Programa de Formación: 233104
- Nombre del Proyecto Formativo: IMPLEMENTACIÓN DE UN SITIO WEB PARA LA I.E.
- Fase del Proyecto: EJECUCIÓN
- Actividad de Proyecto Formativo: IMPLEMENTAR Y PONER EN FUNCIONAMIENTO LA APLICACIÓN DE SOFTWARE CONFORME AL PLAN DE IMPLANTACIÓN ESTABLECIDO, GARANTIZANDO SU CORRECTA CONFIGURACIÓN, EJECUCIÓN Y VALIDACIÓN.
- Competencia: DESARROLLO DE LA SOLUCIÓN DE SOFTWARE
- Resultados de Aprendizaje: RESOLVER PROCESOS LÓGICOS A TRAVÉS DE LA IMPLEMENTACIÓN DE ALGORITMOS Y EL LENGUAJE DE PROGRAMACIÓN SELECCIONADO.
- Duración de la Guía de Aprendizaje (horas): 140 Horas

2. PRESENTACIÓN

¡Bienvenido(a) a esta aventura de aprendizaje en el mundo de la programación!

Esta guía ha sido diseñada para acompañarte paso a paso en el desarrollo de habilidades lógicas, creativas y analíticas que te ayudarán no solo en el área de tecnología, sino también en muchos aspectos de tu vida diaria.

Aprender programación no significa únicamente escribir código en un computador. Significa aprender a pensar, analizar situaciones y encontrar soluciones de manera organizada y eficiente. Cada reto que resuelvas fortalecerá tu capacidad para enfrentar problemas tanto académicos como cotidianos.

LA LÓGICA EN LA VIDA DIARIA

Aunque no siempre lo notemos, usamos lógica todos los días: cuando organizamos nuestro tiempo, seguimos una receta de cocina, planeamos una salida o resolvemos un problema en casa o en el colegio.

La programación toma esas mismas habilidades y las transforma en instrucciones que una computadora puede entender.

Durante esta guía descubrirás que los algoritmos son como mapas de solución: una serie de pasos ordenados que permiten alcanzar un objetivo. Aprenderás a analizar problemas, dividirlos en partes pequeñas y encontrar diferentes caminos para resolverlos de manera creativa.

APRENDER DE FORMA AUTÓNOMA

Uno de los objetivos más importantes de esta guía es ayudarte a desarrollar autonomía en tu aprendizaje. Esto significa que aprenderás a investigar, practicar, experimentar y mejorar por cuenta propia, fortaleciendo habilidades como la disciplina, la responsabilidad y la organización.

En el mundo de la tecnología, siempre aparecen nuevas herramientas y desafíos. Por eso, más que memorizar contenidos, lo importante es aprender a aprender. Cada actividad será una oportunidad para descubrir tus capacidades y avanzar a tu propio ritmo.

CRECER TRABAJANDO EN EQUIPO

La programación también es colaboración. Muchas de las aplicaciones y videojuegos que usamos diariamente fueron creados por equipos de personas que compartieron ideas y trabajaron juntas para alcanzar un mismo objetivo.

A través de actividades grupales, podrás intercambiar conocimientos, escuchar diferentes puntos de vista y fortalecer habilidades como la comunicación, el respeto y el pensamiento crítico. Aprender junto a otros hará el proceso más dinámico, enriquecedor y divertido.

TU CAMINO COMIENZA AQUÍ

Esta guía será una herramienta para explorar, crear y desarrollar nuevas habilidades. No necesitas ser un experto para empezar; lo más importante es la curiosidad, las ganas de aprender y la disposición para intentarlo una y otra vez.

Cada línea de código que escribas será un paso más hacia la construcción de soluciones, ideas y proyectos capaces de transformar tu entorno.

¡Prepárate para aprender, crear y descubrir todo lo que puedes lograr con la programación!

3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

3.1.1 Detente un momento y observa a tu alrededor. ¿Alguna vez has pensado que casi todo lo que haces, desde que te levantas hasta que decides qué ruta tomar para llegar a tu destino, es el resultado de un algoritmo mental?

A menudo creemos que la programación es un idioma extraño que solo las máquinas entienden. Sin embargo, la verdadera programación sucede primero en nuestra mente. Cada vez que divides un problema grande (como organizar un evento o mudarte de casa) en tareas pequeñas, estás aplicando pensamiento computacional.

3.1.2 Pregúntate lo siguiente:

- ¿Soy consciente de los pasos lógicos que sigo para resolver un conflicto cotidiano?
- ¿Qué pasaría si pudiera descomponer cualquier desafío complejo en "subtareas" tan simples que fueran imposibles de fallar?
- ¿Cómo cambiaría mi perspectiva si viera los errores no como fracasos, sino como "bugs" en un proceso lógico que simplemente necesita ser ajustado?

Ambiente requerido: Ambiente de formación

Estrategias o técnicas didácticas activas: Socialización de conceptos e ideas, aprendizaje colaborativo, resolución de problemas y participación mediante actividades prácticas.

Materiales: Videobeam, videos, marcadores, lápices, esferos, hojas, computador, internet

Duración de la actividad: 6 horas.

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

Descripción de la actividad:

Se realizará una actividad inicial de exploración y reconocimiento de conocimientos previos relacionados con la lógica matemática y la construcción de algoritmos. Los aprendices participarán

en ejercicios prácticos y situaciones cotidianas donde identifiquen secuencias lógicas, toma de decisiones y repetición de procesos, relacionándolos con estructuras secuenciales, condicionales, cíclicas y arreglos.

A través de preguntas orientadoras, resolución de problemas básicos y socialización grupal, los aprendices fortalecerán conceptos de lógica proposicional, formulación de algoritmos mediante pseudocódigo y prueba de escritorio. Además, se realizará una introducción al uso de herramientas como PSeInt y lenguaje de programación para la creación y validación de algoritmos.

3.2.1. Actividad de inicio – **"Pensando paso a paso"**

El instructor realizará preguntas orientadoras para que los aprendices relacionen la lógica con actividades diarias:

¿Qué pasos siguen para prepararse antes de ir al colegio?
¿Cómo preparan una receta o realizan una compra?
¿Qué decisiones toman diariamente?
¿Qué actividades repiten constantemente durante el día?

Los aprendices compartirán sus respuestas y se identificarán conceptos como:
Secuencia.
Decisión.
Repetición.
Organización de información.

Estas acciones se relacionan con los algoritmos y la programación porque en la vida diaria seguimos pasos organizados para cumplir una tarea o resolver un problema.

Por ejemplo, al preparar un alimento, al organizar el horario del día o al tomar decisiones, realizamos una secuencia de instrucciones de manera lógica.

De la misma forma, en programación un algoritmo es un conjunto de pasos ordenados y claros que permiten solucionar un problema o ejecutar una tarea específica. La programación utiliza esa lógica cotidiana para crear instrucciones que una computadora pueda entender y ejecutar correctamente.

3.2.2. Actividad grupal – "Resolviendo problemas lógicos"

3.2.2.1. Identificar proposiciones verdaderas y falsas

Ejercicio 1

Indica si las siguientes afirmaciones son verdaderas (V) o falsas (F):

a. 2 + 2 = 4

b. El computador es un dispositivo electrónico.

c. 10 es un número impar.

d. Internet sirve para buscar información.

e. Todos los animales vuelan.

Ejercicio 2

Lee las proposiciones y responde:

"Hoy está lloviendo".

"Tengo clase de programación".

¿Las dos afirmaciones pueden ser verdaderas al mismo tiempo? Explica.

3.2.2.2. Resolver secuencias lógicas

Ejercicio 1

Completa la secuencia:

2 – 4 – 6 – 8 – ___

Ejercicio 2

Observa la secuencia y encuentra el número faltante:

1 – 3 – 5 – 7 – ___

Ejercicio 3

Ordena correctamente los pasos para encender un computador:

( ) Encender el monitor.

( ) Presionar el botón de encendido del CPU.

( ) Ingresar la contraseña.

( ) Esperar que cargue el sistema.

3.2.2.3. **Analizar situaciones con condiciones**

Ejercicio 1

Lee la situación y responde:

Si una persona tiene 18 años o más, entonces puede votar.

Juan tiene 20 años. ¿Puede votar?

María tiene 15 años. ¿Puede votar?

Ejercicio 2

Completa la condición:

Si llueve, entonces _____.

Ejemplo:

llevo paraguas.

no salgo a jugar fútbol.

Ejercicio 3

Analiza la situación:

Si la nota es mayor o igual a 3.0, entonces el aprendiz aprueba la asignatura.

Carlos obtuvo 4.2 → _____

Ana obtuvo 2.5 → _____

3.2.2.4 **Analizar situaciones con ciclos repetitivos**

Ejercicio 1

Lee la situación y responde:

Una persona debe realizar una rutina de ejercicio durante 5 días seguidos.

Día 1: caminar 20 minutos

Día 2: caminar 20 minutos

Día 3: caminar 20 minutos

Día 4: caminar 20 minutos

Día 5: caminar 20 minutos

Responde:

¿Qué acción se repite en cada día?

¿Cuántas veces se repite la actividad en total?

¿Qué estructura de programación representa esta situación?

Ejercicio 2

Analiza la situación:

Un aprendiz debe realizar 10 ejercicios de programación. Por cada ejercicio, debe escribir el algoritmo y ejecutarlo en PSeInt.

Responde:

¿Qué acción se repite en cada ejercicio?

¿Cuántas veces se repite la actividad?

¿Qué estructura se utiliza para representar este proceso?

Ejercicio 3

Lee la situación y responde:

Un estudiante estudia para un examen repitiendo un tema hasta comprenderlo.

Mientras no entienda el tema, entonces _____.

Ejercicio 4

Analiza la situación:

Un sistema solicita ingresar números hasta que el usuario digite cero.

Responde:

¿Qué condición permite detener la repetición?

¿Qué estructura de ciclo se utiliza para este caso?

¿Qué acción se repite mientras la condición no se cumpla?

3.2.2.5 **Actividad adicional grupal (redacción mejorada)**

En equipos de 2 o 3 integrantes, los aprendices deberán desarrollar las siguientes actividades:

- ✓ Elaborar 2 proposiciones verdaderas.
- ✓ Elaborar 2 proposiciones falsas.
- ✓ Diseñar 1 secuencia lógica relacionada con situaciones de la vida cotidiana.
- ✓ Plantear 1 situación condicional contextualizada en un entorno cotidiano (estructura "si… entonces…").
- ✓ Plantear 1 situación que implique ciclos o tareas repetitivas, relacionada con actividades diarias.

Posteriormente, los grupos deberán resolver y analizar ejercicios básicos de lógica proposicional y razonamiento lógico, tales como:

- ✓ Identificación de proposiciones verdaderas y falsas.
- ✓ Resolución de secuencias lógicas.
- ✓ Análisis de situaciones condicionales (estructura "si sucede esto, entonces…").
- ✓ Análisis de situaciones que impliquen procesos repetitivos, aplicando la lógica de ciclos ("mientras se cumpla la condición, realizar…").

Finalmente, cada grupo socializará sus respuestas ante el grupo, explicando de manera clara el razonamiento lógico utilizado en la construcción y solución de cada ejercicio.

Ambiente requerido: Ambiente de formación
Estrategias o técnicas didácticas activas: Socialización de conceptos e ideas, aprendizaje colaborativo, resolución de problemas y participación mediante actividades prácticas.

Materiales de formación

- Computador o portátil.
- Guía de aprendizaje.
- Software PSeInt.
- Editor o compilador de lenguaje de programación.
- Tablero y marcadores.
- Internet y Videobeam.

Material de apoyo

- Presentaciones en diapositivas.
- Tutoriales de PSeInt.
- Ejemplos de algoritmos y diagramas de flujo.
- Documentos PDF y lecturas complementarias.
- Videos explicativos.

Evidencias de aprendizaje

- Algoritmos desarrollados en pseudocódigo.
- Ejercicios resueltos de lógica matemática y programación.
- Pruebas de escritorio realizadas.
- Programas funcionales en PSeInt o lenguaje de programación.
- Sustentación de actividades prácticas.

Subir las evidencias al Drive del Instructor Germán Zarza

Instrumentos de evaluación

- Lista de chequeo.
- Talleres prácticos.
- Estudios de caso.
- Observación directa.
- Prueba práctica.

Duración de la actividad: 24 horas.

3.3 Actividades de apropiación:

Descripción de la actividad:

Los aprendices desarrollarán actividades prácticas orientadas a fortalecer la comprensión y aplicación de la lógica matemática y la construcción de algoritmos mediante pseudocódigo y herramientas digitales.

A través de ejercicios individuales y colaborativos, los aprendices diseñarán soluciones a problemas cotidianos utilizando estructuras secuenciales, condicionales, cíclicas y arreglos.

Además, implementarán pruebas de escritorio para verificar el funcionamiento de los algoritmos desarrollados en PSeInt.

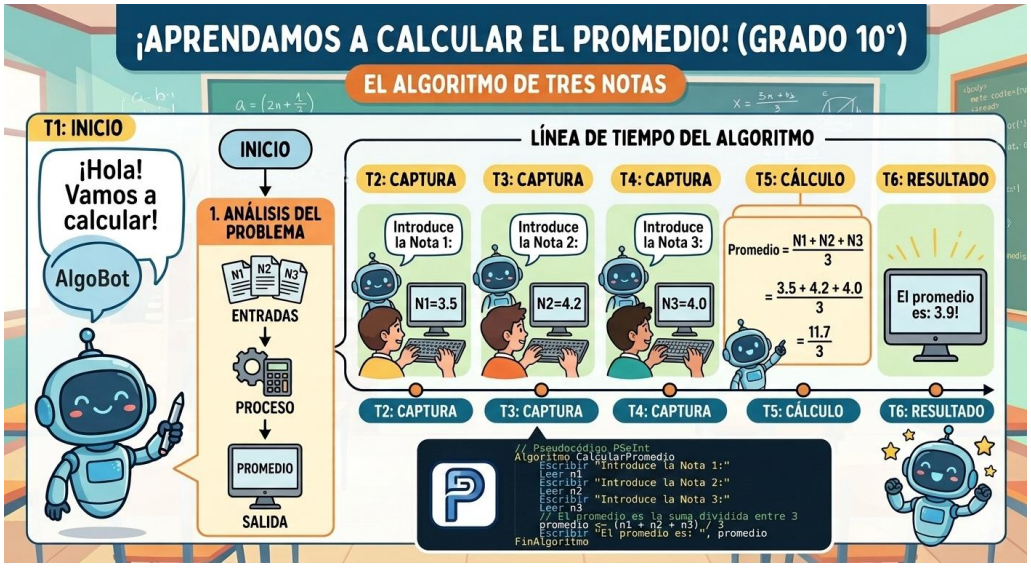
Las actividades permitirán que los aprendices analicen situaciones problema, organicen ideas de manera lógica y construyan soluciones estructuradas utilizando herramientas básicas de programación.

3.3.1 Actividades propuestas

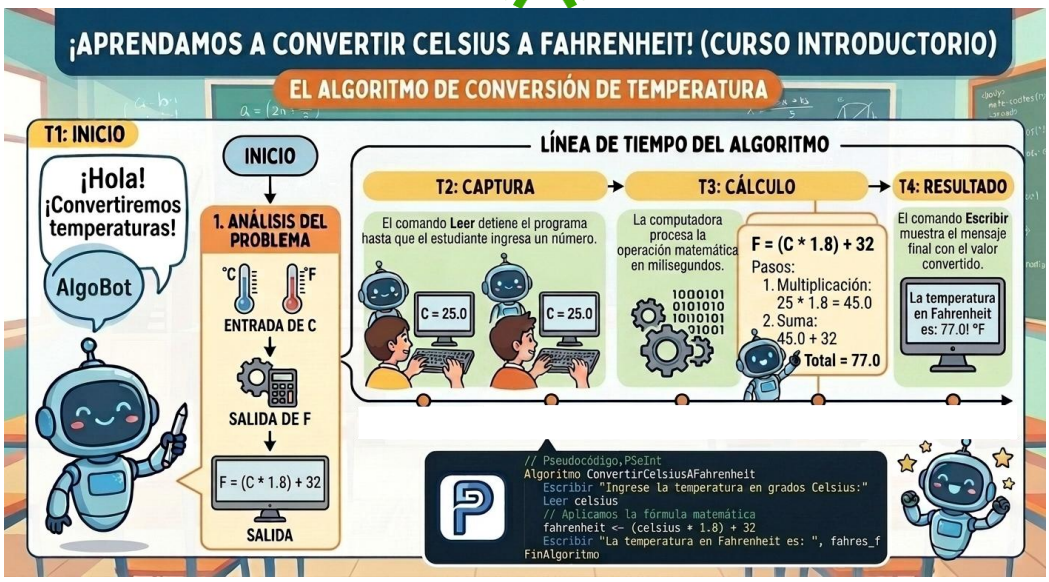
3.3.1.1. Construcción de algoritmos secuenciales

Los aprendices elaborarán algoritmos en pseudocódigo para resolver situaciones cotidianas, por ejemplo:

a) Desarrollar el algoritmo para calcular el promedio de tres notas.



b) Desarrollar el algoritmo para convertir grados Celsius a Fahrenheit.



c) Calcular el área de una figura Triángulo.

d) Calcular el área de una figura Rectángulo.

e) Calcular el área de una figura Cuadrado.

f) Realizar las operaciones de suma, resta y multiplicación de tres números.

g) Tienes un balde grande lleno con 24 litros de agua y tres baldes vacíos con capacidades de 13, 11 y 8 litros respectivamente.

El objetivo: Debes trasvasar el agua entre los recipientes de modo que al final queden exactamente 8 litros en tres de los baldes (es decir, dividir el contenido total de 24 litros en tres partes iguales).

Las reglas:

i. No hay marcas de medida en los baldes; solo sabes cuándo están totalmente llenos o vacíos.

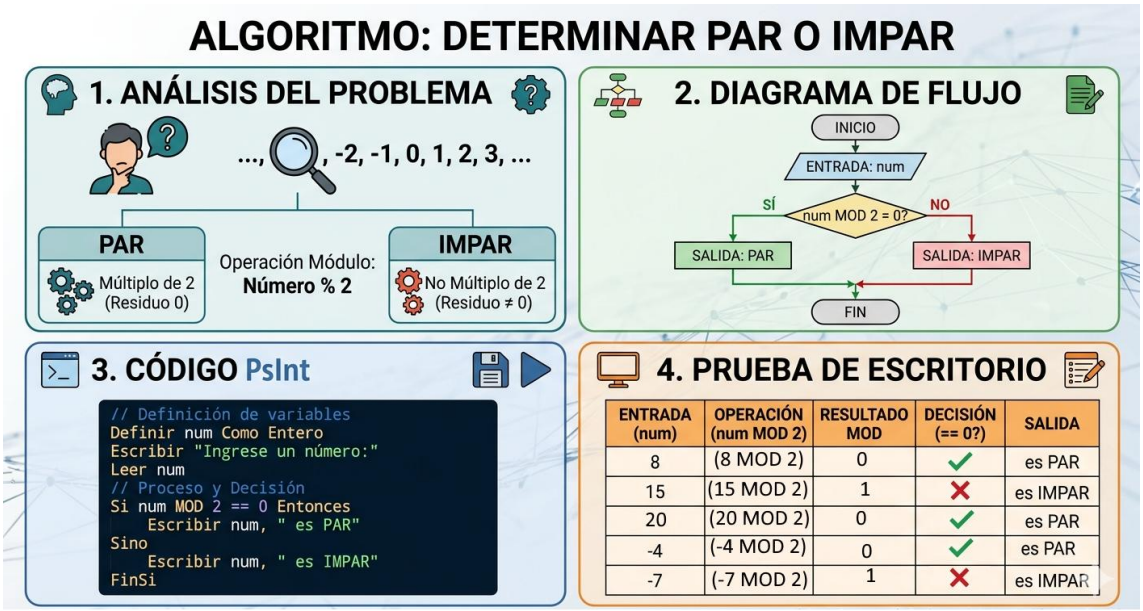
ii. No se puede desperdiciar agua.

iii. Solo puedes verter agua de un balde a otro hasta que el balde de origen se vacíe o el balde de destino se llene por completo.

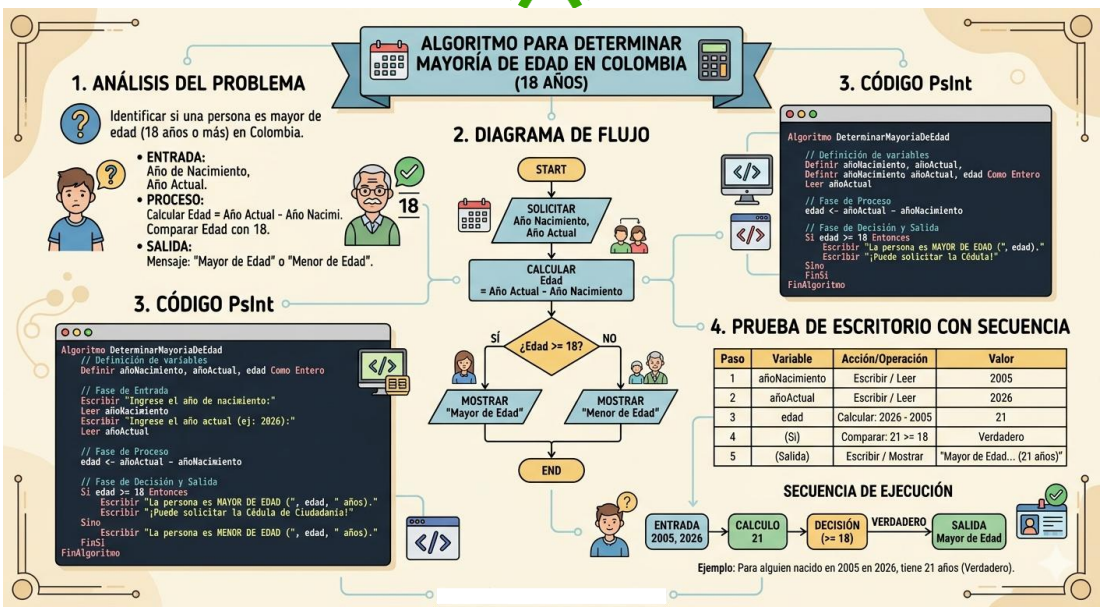
3.3.2. Aplicación de estructuras condicionales

Los aprendices resolverán ejercicios donde deban tomar decisiones lógicas:

a) Desarrollar el algoritmo para determinar si un número es par o impar.



b) Identificar si una persona es mayor de edad en Colombia.



c) Desarrollar un algoritmo que permita calcular el descuento aplicado a una compra. Si el valor de la compra es superior a $1.000.000, se debe aplicar un descuento al total de la compra y mostrar el valor final a pagar.

d) Realizar un algoritmo que permita digitar dos números enteros diferentes, visualizar en pantalla cuales es el número mayor.

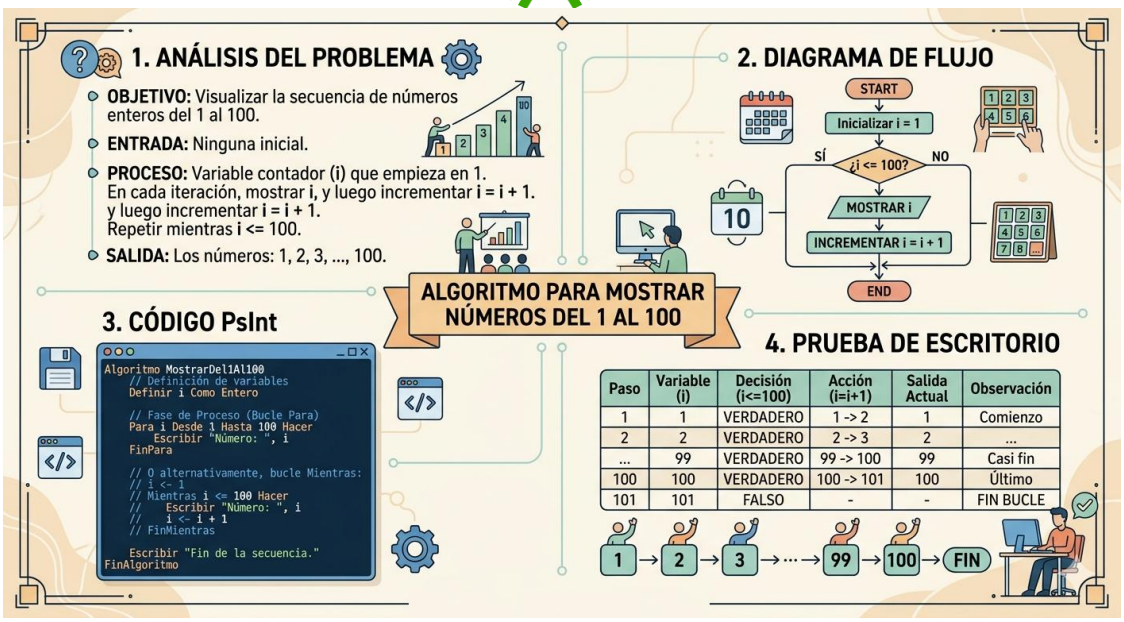
e) Realizar un algoritmo que permita digitar dos números enteros diferentes, visualizar en pantalla cuales es el número menor.

f) Desarrollar un algoritmo que permita ingresar un número entero y determine si es positivo, negativo o cero.

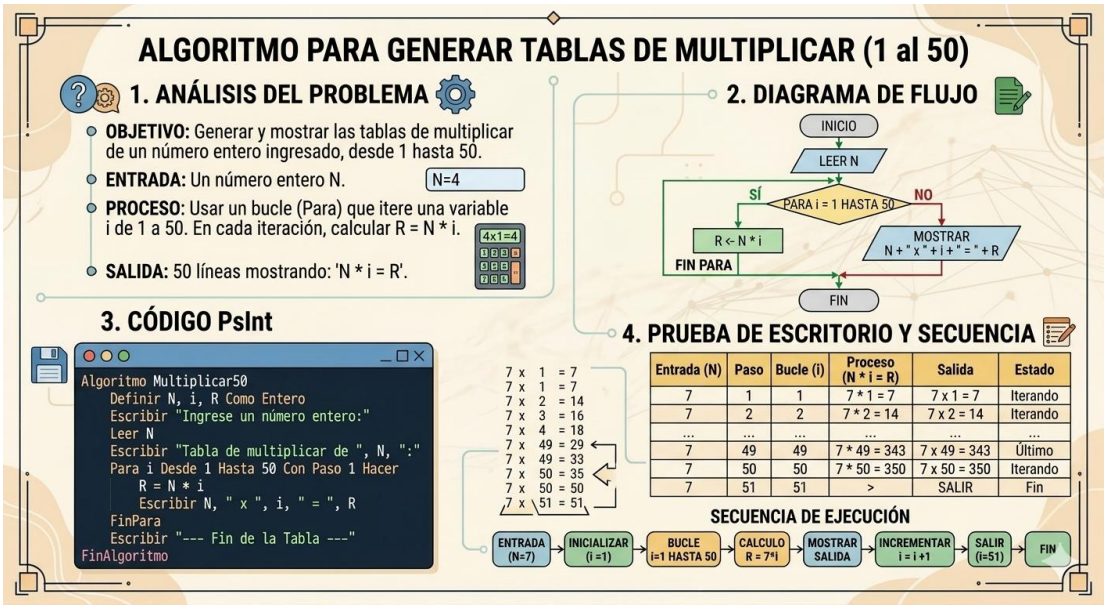
3.3.3. Uso de estructuras cíclicas

Los aprendices desarrollarán algoritmos repetitivos para:

a) Desarrollar el algoritmo para mostrar números del 1 al 100.



b) Realizar un algoritmo que permita digitar un numero entero y generar tablas de multiplicar del 1 al 50.



c) Realizar un algoritmo que permita generar los primero 100 números pares positivos.

d) Realizar un algoritmo que permita generar los primero 100 números impares positivos.

e) Realizar un algoritmo que permita generar los primero 100 números pares negativos.

f) Realizar un algoritmo que permita generar los primero 100 números impares negativos.

g) Desarrollar un algoritmo que permita generar los números del 1 al 100 y muestre cuáles son pares y cuáles son impares.

g) Desarrollar un algoritmo que permita sumar los primeros 100 números naturales.

h) Desarrollar un algoritmo que permita calcular el promedio de los primeros 100 números naturales.

i) Desarrollar un algoritmo que permita generar la tabla de multiplicar de un número ingresado por el usuario.

j) Desarrollar un algoritmo que permita mostrar los números del 100 al 1 en orden descendente.

k) Desarrollar un algoritmo que permita generar los primeros 50 números múltiplos de 3.

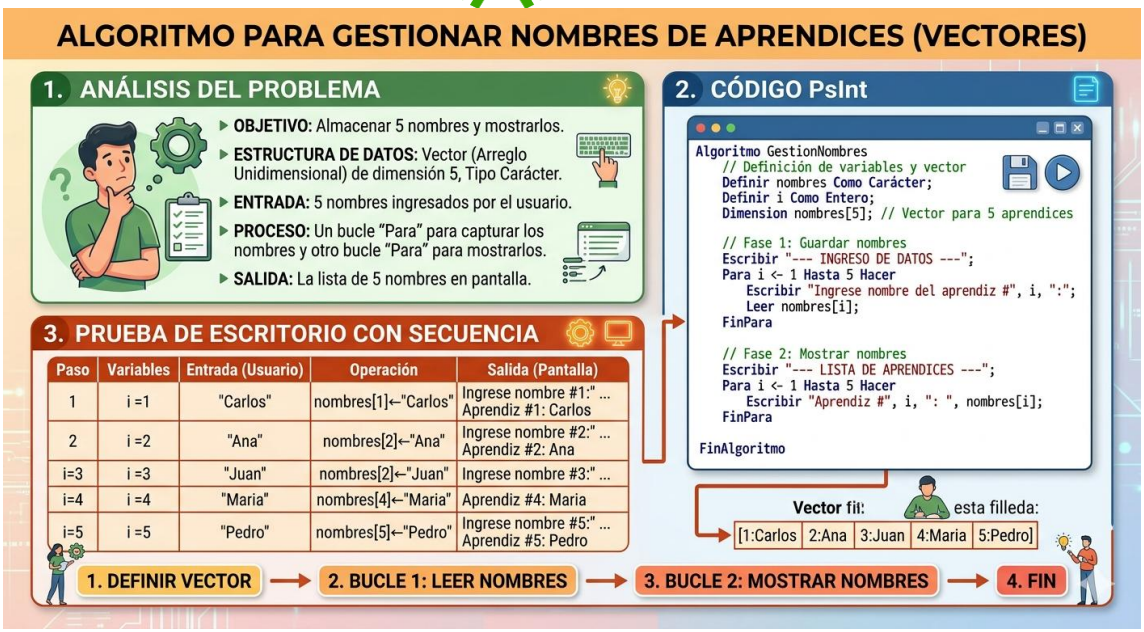
l) Desarrollar un algoritmo que permita generar los primeros 50 números múltiplos de 5.

m) Desarrollar un algoritmo que permita contar cuántos números pares e impares hay entre 1 y 100.

3.3.4. Construcción de algoritmos con arreglos

Los aprendices almacenarán y procesarán información utilizando arreglos:

- **Vectores (arreglos de una sola dimensión)**
    a) Desarrollar un algoritmo para guardar 5 nombres de aprendices y mostrarlos.



b) Ingresar 5 notas y calcular el promedio.

c) Buscar el número mayor de una lista pequeña.

- Matrices (tablas de datos):

  Los aprendices organizarán información en forma de tabla (filas y columnas), por ejemplo, notas de aprendices en diferentes materias.

  a) Desarrollar un algoritmo que permita ingresar los valores de una matriz de 3x3 y luego mostrarlos en forma de tabla.

  b) Guardar calificaciones de 3 aprendices en 2 materias.

  c) Mostrar todos los datos almacenados en una tabla.

3.3.5. Prueba de escritorio y validación

Cada aprendiz realizará pruebas de escritorio para verificar:

Entrada de datos.

Procesamiento lógico.

Resultados esperados.

Prueba de escritorio

Entrada de datos	Procesamiento lógico	Resultado
nota1 = 4.0	(4.0 + 3.5 + 5.0) / 3 = 12.5 / 3	4.17
nota2 = 3.5		
nota3 = 5.0		

Interpretación:

- **Entrada de datos:** se ingresan las notas del estudiante.
- **Procesamiento lógico:** se suman las notas y se divide entre 3.
- **Resultado esperado:** el promedio final del estudiante.

Ambiente requerido: Ambiente de formación
Estrategias o técnicas didácticas activas: Socialización de conceptos e ideas, aprendizaje colaborativo, resolución de problemas y participación mediante actividades prácticas.
Materiales de formación

- Computador o portátil.
- Guía de aprendizaje.
- Software PSeInt.
- Editor o compilador de lenguaje de programación.
- Tablero y marcadores.
- Internet y Videobeam.

Material de apoyo

- Presentaciones en diapositivas.
- Tutoriales de PSeInt.
- Ejemplos de algoritmos y diagramas de flujo.
- Documentos PDF y lecturas complementarias.
- Videos explicativos.

Evidencias de aprendizaje

- Algoritmos desarrollados en pseudocódigo.
- Ejercicios resueltos de lógica matemática y programación.
- Pruebas de escritorio realizadas.

- Programas funcionales en PSeInt o lenguaje de programación.
- Sustentación de actividades prácticas.

Subir las evidencias al Drive del Instructor Germán Zarza

Instrumentos de evaluación

- Lista de chequeo.
- Talleres prácticos.
- Estudios de caso.
- Observación directa.
- Prueba práctica.

Duración de la actividad: 50 horas.

3.4 Actividades de Transferencia el Conocimiento:

Descripción de la actividad:

En esta etapa, los aprendices aplicarán los conocimientos adquiridos sobre lógica proposicional, algoritmos, estructuras secuenciales, condicionales, cíclicas y arreglos, para resolver problemas más cercanos a situaciones reales de su entorno.

Los aprendices deberán diseñar, desarrollar y validar algoritmos sencillos en pseudocódigo y en PSeInt, demostrando la capacidad de transferir lo aprendido a nuevos contextos. Además, socializarán sus soluciones explicando el proceso lógico utilizado.

Ejercicios:

1. Se dispone de tres recipientes con capacidades máximas de 8, 5 y 3 litros. Inicialmente, la jarra de 8 litros está completamente llena, mientras que las de 5 y 3 litros están vacías.
    a. **Objetivo**:

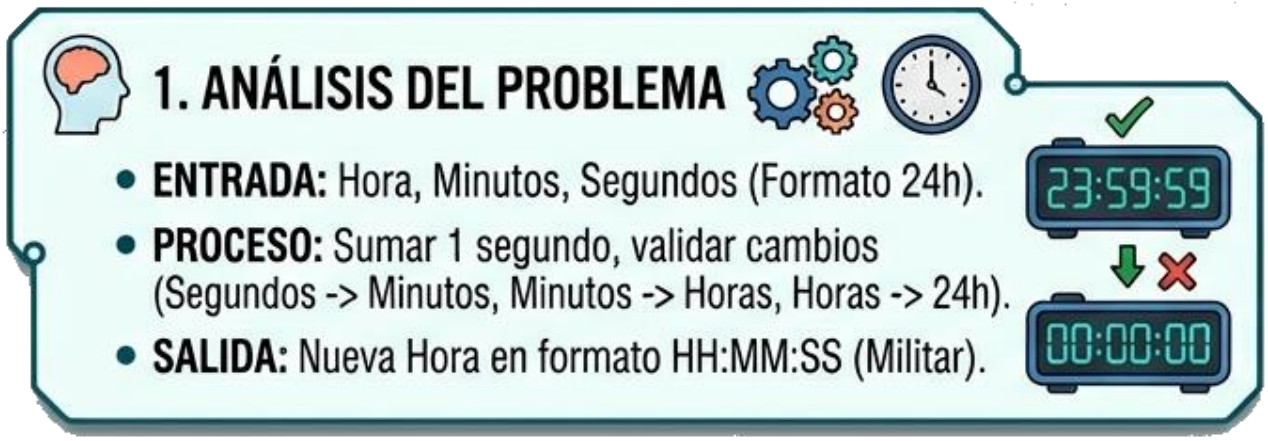
        es diseñar un algoritmo que, mediante una serie de trasvases, logre obtener exactamente 4 litros en dos de las jarras (8L y 5L).

    b. Restricciones:

        i. No se permiten medidas intermedias a ojo; solo se puede verter agua hasta que la jarra de origen se vacíe o la de destino se llene.

ii. La cantidad total de agua (8 litros) debe conservarse en todo momento (sistema cerrado).

2. Desarrollar un algoritmo que permita ingresar la hora, minutos, segundo, pasado un segundo mostrar la nueva hora. La hora se muestra en formato militar.



3. Desarrollar un algoritmo que permita ingresar el nombre de un estudiante y las tres notas correspondientes a la asignatura de Tecnología e Informática.

   El algoritmo debe calcular el promedio de las tres notas y determinar el nivel de desempeño del estudiante, de acuerdo con los siguientes rangos:

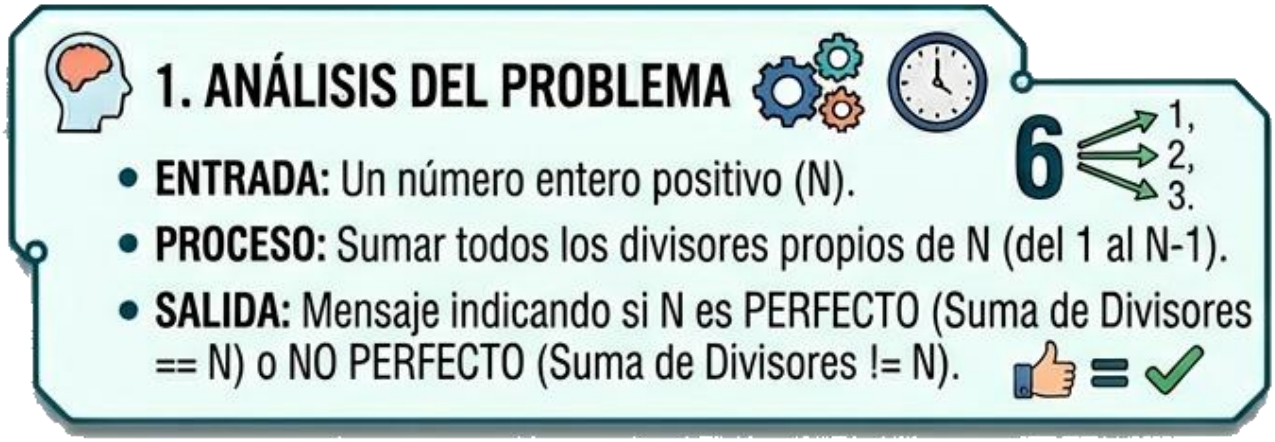
Bajo: promedio mayor o igual a 1.0 y menor que 3.0

Básico: promedio mayor o igual a 3.0 y menor que 4.0

Alto: promedio mayor o igual a 4.0 y menor que 4.6

Superior: promedio mayor o igual a 4.6 y menor o igual a 5.0

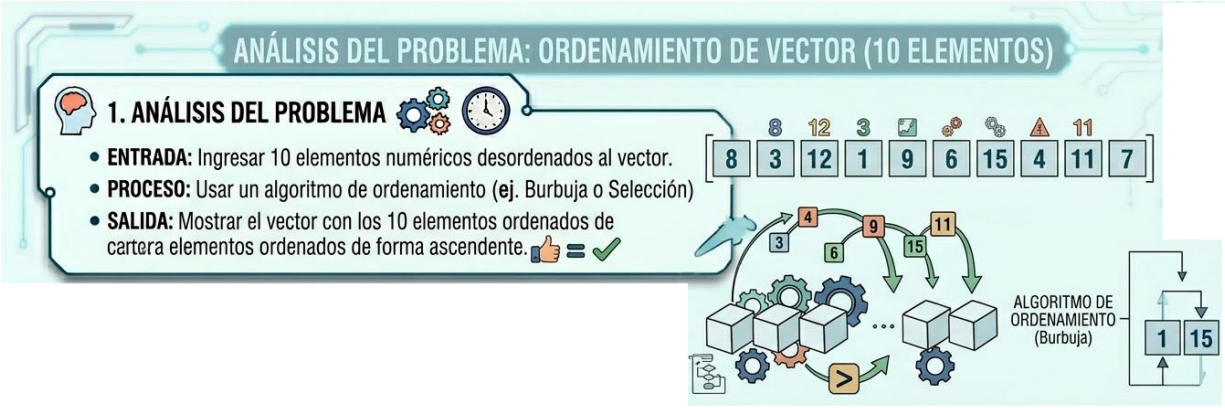
4. Finalmente, se debe mostrar el nombre del estudiante, el promedio obtenido y el nivel de desempeño correspondiente.
5. Desarrollar un algoritmo que permita ingresar el mes (en letras o en número) y el año, y determine la cantidad de días correspondientes a ese mes, teniendo en cuenta si el año es bisiesto en el caso del mes de febrero.
6. Desarrollar un algoritmo que permita ingresar la fecha y la hora en formato militar (día, mes, año, horas, minutos y segundos) y, al transcurrir un segundo, muestre en pantalla la fecha y hora actualizada, considerando los cambios correspondientes en segundos, minutos, horas, días, meses y años según corresponda.
7. Desarrollar un algoritmo que permita ingresar una cadena de texto y determinar si es un palíndromo, es decir, si se lee igual de izquierda a derecha que de derecha a izquierda, ignorando espacios y caracteres especiales si es necesario.
8. Desarrollar un algoritmo que permita ingresar un número y determinar si el número es perfecto o no.



2. PRUEBA DE ESCRITORIO CON SECUENCIA

Paso	N	Divisores Propios	Suma Actual	Comparación (Suma==N?)	Salida (Mensaje)
Caso 1 (Perfecto)	6	[1,2,3]	6	6==6?	6 ES UN NÚMERO PERFECTO
Caso 2 (No Perfecto)	8	[1,2,4]	7	7==8?	8 NO ES UN NÚMERO PERFECTO
Caso 3 (Perfecto)	28	[1,2,4,7,14]	28	28==28?	28 ES UN NÚMERO PERFECTO

9. Desarrollar un algoritmo que permita ingresar 10 números en un arreglo y determine cuál es el número mayor.
10. Desarrollar un algoritmo que permita ingresar 10 elementos al vector de forma desordena y mostrar el vector ordenado.



11. Desarrollar un algoritmo que permita ingresar los valores de una matriz cuadrada (3x3) y calcular la **diagonal principal** y la **diagonal secundaria**, mostrando sus valores y sumas.

Ambiente requerido: Ambiente de formación

Estrategias o técnicas didácticas activas: Socialización de conceptos e ideas, aprendizaje colaborativo, resolución de problemas y participación mediante actividades prácticas.

Materiales de formación

- Computador o portátil.
- Guía de aprendizaje.
- Software PSeInt.
- Editor o compilador de lenguaje de programación.
- Tablero y marcadores.
- Internet y Videobeam.

Material de apoyo

- Presentaciones en diapositivas.
- Tutoriales de PSeInt.
- Ejemplos de algoritmos y diagramas de flujo.
- Documentos PDF y lecturas complementarias.
- Videos explicativos.

Evidencias de aprendizaje

- Algoritmos desarrollados en pseudocódigo.
- Ejercicios resueltos de lógica matemática y programación.
- Pruebas de escritorio realizadas.
- Programas funcionales en PSeInt o lenguaje de programación.
- Sustentación de actividades prácticas.

Subir las evidencias al Drive del Instructor Germán Zarza

Instrumentos de evaluación

- Lista de chequeo.
- Talleres prácticos.
- Estudios de caso.
- Observación directa.
- Prueba práctica.

Duración de la actividad: 60 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
Ejecución	Implementar la aplicación de software de	creación y aplicación de algoritmos	**Algoritmos elaborados**	Emplea lenguaje de programación	Estudios de casos

	acuerdo con el plan de implantación.	estructura ciclo y arreglos para el manejo del lenguaje de programación	**en pseudocódigo**	para la solución de problemas de lógica proposicional según el análisis realizado	Lista de chequeo

5. GLOSARIO DE TÉRMINOS

A

Algoritmo
Conjunto ordenado y finito de pasos que permiten resolver un problema o realizar una tarea.

Arreglo (Array)
Estructura de datos que almacena varios elementos del mismo tipo en posiciones consecutivas de memoria.

B

Bug
Error, falla o defecto presente en un algoritmo, programa o sistema de software que provoca un funcionamiento incorrecto o resultados inesperados. Los bugs pueden originarse por errores de lógica, sintaxis o ejecución durante el desarrollo de un programa.

C

Ciclo
Estructura que permite repetir un bloque de instrucciones varias veces.

Condicional
Estructura de control que ejecuta instrucciones dependiendo de si una condición es verdadera o falsa.

D

Diagrama de flujo
Representación gráfica de un algoritmo mediante símbolos conectados por flechas que indican el flujo de ejecución.

E

Entidades primitivas
Elementos básicos utilizados en programación, como datos, constantes, variables y operadores.

Estructura cíclica
Tipo de estructura de control que repite instrucciones mientras se cumpla una condición.

Estructura condicional
Estructura que permite tomar decisiones en un algoritmo.

Estructura secuencial
Conjunto de instrucciones que se ejecutan una tras otra en orden.

H

Herramientas de programación
Aplicaciones o programas utilizados para diseñar, desarrollar y probar algoritmos y software.

J

Jerarquía de operadores
Orden de prioridad con el que se ejecutan los operadores en una expresión matemática o lógica.

L

Lenguaje de programación
Sistema de símbolos y reglas que permite escribir instrucciones para que una computadora ejecute tareas.

Lógica matemática
Rama de las matemáticas que estudia el razonamiento correcto mediante símbolos y reglas formales.

Lógica proposicional
Parte de la lógica matemática que trabaja con proposiciones y operadores lógicos.

O

Operadores lógicos
Símbolos utilizados para relacionar proposiciones, como AND, OR y NOT.

Operadores relacionales
Operadores que comparan valores, por ejemplo: mayor que (>), menor que (<) o igual (=).

P

PSeInt
Herramienta educativa utilizada para escribir y probar algoritmos en pseudocódigo.

Prueba de escritorio
Técnica para verificar el funcionamiento de un algoritmo ejecutándolo manualmente paso a paso.

Pseudocódigo
Forma de describir algoritmos utilizando lenguaje similar al humano y estructuras de programación.

Proposición
Enunciado que puede ser verdadero o falso.

S

Software
Conjunto de programas e instrucciones que permiten el funcionamiento de un sistema informático.

T

Tipos de algoritmos
Clasificación de algoritmos según su comportamiento, como secuenciales, condicionales o repetitivos.

Tipos de Variables

Variable Entera (Integer)
Tipo de variable que almacena números enteros sin decimales.
Variable Real (Float o Double)
Variable que almacena números con parte decimal.
Variable Carácter (Char)
Variable que almacena un único carácter, letra o símbolo.
Variable Cadena (String)
Tipo de variable que almacena palabras, frases o conjuntos de caracteres.
Variable Lógica o Booleana (Boolean)
Variable que solo puede almacenar dos valores: verdadero o falso.

Variable Constante
Dato cuyo valor no cambia durante la ejecución de un programa.

V

Variable
Espacio en memoria donde se almacena un dato que puede cambiar durante la ejecución de un programa.

6. REFERENTES BILBIOGRÁFICOS

Programa PsInt https://pseint.sourceforge.net/

https://editorial.uaa.mx/docs/algoritmos.pdf

https://repository.unad.edu.co/bitstream/handle/10596/11970/Chaves%20A%20%282017%29%20Aprenda%20a%20dise%C3%B1ar%20algoritmos%20%281%29.pdf?sequence=3&isAllowed=y

https://digitk.areandina.edu.co/server/api/core/bitstreams/8bb839a6-a4af-4449-b151-7258eb83b6fb/content

https://libros.metabiblioteca.org/bitstream/001/169/8/AlgoritmosProgramacion.pdf

https://dialnet.unirioja.es/descarga/articulo/6595073.pdf

• Real Academia Española. (2020). Algoritmo. Diccionario de la lengua española. RAE. https://dle.rae.es/algoritmo

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	**German Zarza Acosta**	**Instructor**	**Articulación**	**01-03-2026**

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

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					