PUCE Virtual

CLASE

1

ESTRUCTURA DE DATOS (PROGRAMACIÓN III)

Algoritmos y búsquedas

QS WORLD UNIVERSITY RANKINGS

Educación de calidad

INTRODUCCIÓN DE LA CLASE

GRADO / POSGRADO / TECNOLOGÍAS

Estudia a tu tiempo
son interrupciones

EDUCACIÓN EN LÍNEA DE CALIDAD

El estudio de los algoritmos y las estructuras de datos constituye una base esencial para el desarrollo de soluciones tecnológicas eficientes, especialmente en el ámbito de la ciberseguridad. Comprender qué es un algoritmo, cómo se representa una estructura de datos y de qué manera ambos interactúan permite diseñar sistemas capaces de procesar grandes volúmenes de información de forma rápida y segura. En este contexto, el análisis de complejidad se vuelve fundamental, ya que permite evaluar el rendimiento de los algoritmos en términos de tiempo y espacio, utilizando herramientas matemáticas como la notación Big-O, Ω y Θ. Estos conceptos facilitan la comparación entre diferentes soluciones y ayudan a seleccionar la más adecuada para procesar eventos de seguridad en tiempo real.

Así también, los algoritmos de búsqueda juegan un papel clave en tareas de ciberseguridad como la identificación de usuarios, el rastreo de direcciones IP y el análisis de registros de eventos. Métodos como la búsqueda lineal y la búsqueda binaria presentan diferencias importantes en eficiencia, especialmente cuando se trabaja con grandes volúmenes de datos. Mientras la búsqueda lineal examina los datos de forma secuencial, la búsqueda binaria requiere que la información esté previamente ordenada, pero ofrece tiempos de respuesta mucho más rápidos. Comprender estas diferencias permite optimizar los sistemas de monitoreo y detección, mejorando la capacidad de respuesta ante amenazas y fortaleciendo la protección de la información.

Semana 1:
Comprender los conceptos de algoritmos y estructuras de datos, analizando su funcionamiento y eficiencia computacional en la resolución de problemas de ciberseguridad.

1. Algoritmos y búsquedas

1.1. Fundamentos de algoritmos y análisis de complejidad

El estudio de los algoritmos y las estructuras de datos es fundamental en la informática, ya que permite diseñar soluciones eficientes, escalables y confiables para el procesamiento de información. Un algoritmo es una secuencia finita y ordenada de pasos lógicos que resuelve un problema específico, mientras que una estructura de datos es la forma en que la información se organiza y almacena para facilitar su acceso y manipulación. Ambos conceptos están estrechamente relacionados: un mismo problema puede resolverse con distintos algoritmos, pero su eficiencia dependerá en gran medida de cómo estén organizados los datos. Por ejemplo, buscar información en datos desordenados no ofrece el mismo rendimiento que hacerlo en datos previamente estructurados.

El análisis de algoritmos permite evaluar el consumo de recursos antes de implementar una solución. Este análisis se enfoca principalmente en el tiempo de ejecución (complejidad temporal) y el uso de memoria (complejidad espacial). No se mide el tiempo en segundos reales, sino el crecimiento del número de operaciones a medida que aumenta el tamaño de los datos. Esto permite predecir el comportamiento del algoritmo en escenarios de gran escala y comparar distintas alternativas de manera objetiva.

En ciberseguridad, donde se procesan grandes volúmenes de registros y eventos en tiempo real, la eficiencia algorítmica es crucial. Los sistemas deben analizar millones de datos rápidamente para detectar accesos no autorizados o patrones de ataque. Seleccionar algoritmos y estructuras adecuadas mejora el rendimiento y fortalece la protección de la información.

1.1.1. Concepto de algoritmo y estructura de datos

Algoritmo: Un algoritmo puede definirse como un conjunto de instrucciones diseñadas para localizar, calcular o procesar información. Por ejemplo, los algoritmos de búsqueda permiten localizar información dentro de grandes volúmenes de datos, como registros de usuarios o eventos de red. (Tymoschuk, Guzmán, & Frittelli. 2021).

Los algoritmos pueden representarse de diversas maneras según el nivel de formalidad y el propósito didáctico o técnico que se persiga. Las formas más comunes son el lenguaje natural, el seudocódigo, los diagramas de flujo y los lenguajes de programación. Cada una de estas representaciones cumple una función específica: el lenguaje natural facilita la comprensión inicial del problema, el seudocódigo permite estructurar la lógica de manera más formal sin depender de la sintaxis de un lenguaje específico, los diagramas de flujo ayudan a visualizar el proceso mediante símbolos gráficos, y los lenguajes de programación convierten el algoritmo en una solución ejecutable por la computadora.

La Figura 1 presenta un mismo algoritmo representado en tres formatos distintos: lenguaje natural, seudocódigo y en el lenguaje Python. Esta comparación permite observar cómo una

misma lógica puede expresarse con diferentes niveles de precisión y formalidad. Mientras el lenguaje natural describe los pasos de manera general, el seudocódigo organiza las instrucciones siguiendo una estructura más cercana a la programación, y el código en Python traduce esas instrucciones en una forma que puede ser compilada o interpretada por el sistema.

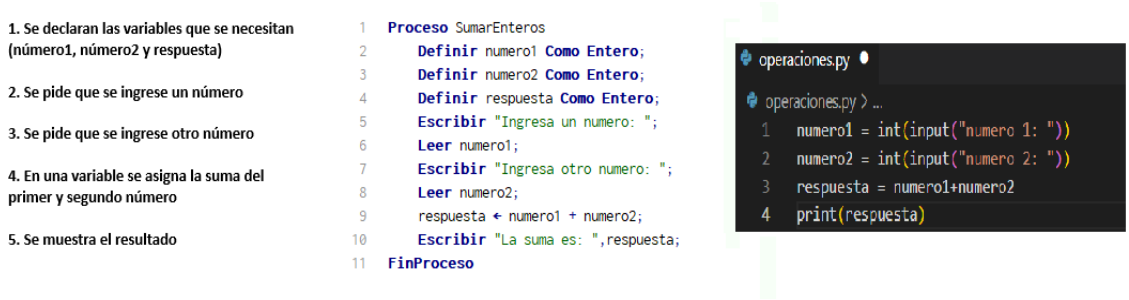


Figura 1. Representación de un algoritmo. Creación de autor Patricio Coba

Estructura de datos: Constituyen el mecanismo mediante el cual la información se organiza, almacena y gestiona dentro de un sistema informático, con el fin de facilitar operaciones fundamentales como inserción, eliminación, actualización y búsqueda de datos. Su diseño no es arbitrario, sino que responde a las necesidades específicas del problema que se desea resolver. Entre las estructuras más utilizadas se encuentran los arreglos, las listas enlazadas, las pilas, las colas, los árboles y las tablas hash, cada una con características particulares en términos de acceso, almacenamiento y eficiencia.

La elección adecuada de una estructura de datos influye directamente en el rendimiento del algoritmo que la utiliza. Por ejemplo, una tabla hash puede ofrecer búsquedas en tiempo promedio constante, mientras que un arreglo puede requerir recorridos completos si no está ordenado. De esta manera, no basta con diseñar un algoritmo correcto; también es necesario seleccionar la estructura de datos más conveniente para optimizar el uso de tiempo y memoria. Como señala Joyanes Aguilar (2003), la correcta combinación entre algoritmos y estructuras de datos es determinante para lograr soluciones eficientes y escalables en el desarrollo de software.

8 pasos para dominar algoritmos (en Programación). Video con una explicación breve, pero efectiva sobre conceptos básicos y avanzados para crear algoritmos, así como consejos útiles para quien inicia programando. https://www.youtube.com/watch?v=HaLLqlGn78M

1.1.2. Algoritmos en la resolución de problemas de ciberseguridad

En la Tabla 1 que se presenta a continuación, se muestran algunas de las aplicaciones más utilizadas en el campo de la ciberseguridad, junto con un ejemplo práctico de cada una. Estas aplicaciones son de gran importancia, ya que permiten fortalecer los sistemas de protección, mejorar la detección de amenazas y optimizar la gestión de la seguridad informática.

Tabla 1. Aplicación de algoritmos en ciberseguridad. Creación de autor Patricio Coba

Proceso	Detalle	Ejemplo
Detección patrones de ataques	Consiste en identificar comportamientos repetitivos o secuencias conocidas que coinciden con ataques previamente registrados.	Detectar múltiples intentos de login fallidos desde la misma IP en menos de 1 minuto.
Análisis de registros de eventos	Implica revisar logs del sistema para encontrar actividades sospechosas o errores de seguridad.	Revisar el log del servidor para encontrar accesos fuera del horario laboral.
Búsqueda de firmas de malware	Consiste en comparar archivos o datos contra bases de datos de código malicioso conocido.	Analizar un archivo descargado y compararlo con una base de datos de virus conocidos.
Identificación de anomalías en tráfico de red	Busca comportamientos inusuales en el flujo normal de datos de la red.	Detectar un equipo interno enviando grandes cantidades de datos a un servidor externo desconocido.

Entre las aplicaciones de los algoritmos en ciberseguridad, existen algoritmos especializados de búsqueda de patrones que permiten localizar múltiples firmas de malware dentro de grandes volúmenes de datos con alta eficiencia. Algunos algoritmos de búsqueda de cadenas pueden encontrar múltiples coincidencias en tiempo lineal respecto al tamaño del texto analizado. Esto es especialmente útil en sistemas de detección de intrusos, donde se deben analizar millones de registros en poco tiempo. (Stallings. 2021)

1.1.3. Complejidad temporal y espacial

La complejidad temporal evalúa cómo varía el tiempo de ejecución de un algoritmo en función del tamaño de los datos de entrada. Por su parte, la complejidad espacial analiza la cantidad de memoria que el algoritmo necesita para ejecutarse, considerando tanto las variables utilizadas como las estructuras de datos adicionales que pueda requerir. Ambos conceptos son fundamentales para determinar la eficiencia de una solución informática.

Es importante destacar que la complejidad temporal no se mide en segundos reales, sino en términos del crecimiento del número de operaciones a medida que aumenta el volumen de datos. Este enfoque abstracto permite comparar algoritmos de manera objetiva, independientemente del hardware, sistema operativo o lenguaje de programación utilizado. De esta forma, se pueden seleccionar soluciones más eficientes y escalables para entornos donde el procesamiento de grandes cantidades de información es crítico. La figura 2 muestra el gráfico de medición del Análisis de Complejidad Temporal y Espacial del Algoritmo Test de Primalidad Tesla Zollner.

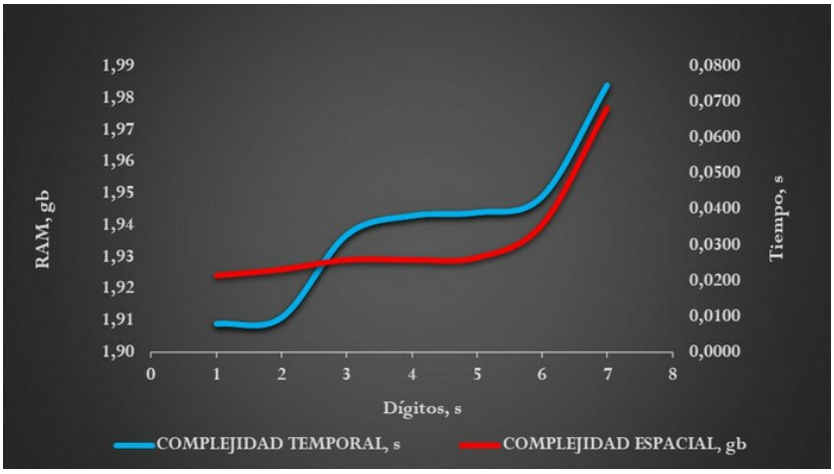


Figura 2. Análisis de Complejidad Temporal y Espacial del algoritmo test de Primalidad Tesla Zollner. Rosas Soto, T., & Zollner I. Castellano. (2024).

En ciberseguridad, optimizar tiempo y memoria es vital, porque los sistemas deben analizar información en tiempo real sin saturar recursos.

1.1.4. Notación Big-O, Ω y Θ

Las notaciones asintóticas se utilizan para describir el comportamiento de los algoritmos:

- Big-O (O): Representa el límite superior del crecimiento de un algoritmo, es decir, describe el peor escenario posible en términos de tiempo o espacio. Permite estimar el máximo número de operaciones que podría realizar el algoritmo cuando el tamaño de los datos aumenta, garantizando que su rendimiento no será peor que ese límite.

- Big-Omega (Ω): Representa el límite inferior, es decir, el mejor caso posible. Indica el mínimo número de operaciones que el algoritmo necesita para resolver el problema bajo condiciones ideales.
- Big-Theta (Θ): Representa el crecimiento ajustado o comportamiento típico del algoritmo cuando el límite superior e inferior coinciden. Describe de manera más precisa el rendimiento cuando el algoritmo mantiene un patrón de crecimiento estable en la mayoría de los casos.

Estas notaciones describen el crecimiento del tiempo o memoria en función del tamaño de la entrada.

Por ejemplo:

- O(n): crecimiento lineal.
- O(log n): crecimiento logarítmico.
- $O(n^2)$: crecimiento cuadrático.

Estas métricas permiten comparar algoritmos y seleccionar el más eficiente para cada escenario.

1.1.5. Análisis de eficiencia en procesamiento de eventos de seguridad

Los sistemas de seguridad informática generan grandes volúmenes de datos. Analizar estos datos requiere algoritmos que mantengan rendimiento estable a medida que crece la información. Por ejemplo, los algoritmos de búsqueda eficientes permiten analizar logs rápidamente, detectar accesos sospechosos o identificar patrones de ataque. Los algoritmos lineales son útiles en pequeños volúmenes, mientras que los logarítmicos permiten escalar a sistemas masivos. (Cormen, Leiserson, Rivest, & Stein. 2022)

La Figura 3 muestra un panel en tiempo real (dashboard) con datos de acciones SSH fallidas y exitosas, esto muestra como la aplicación de los algoritmos en ciberseguridad puede apoyar a la mitigación de problemas de red.

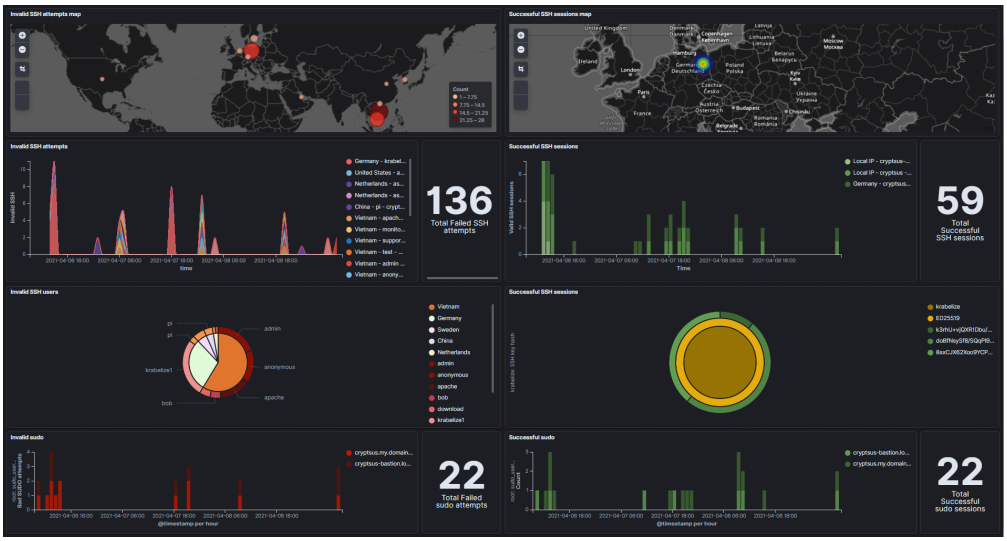


Figura 3. Real-time SSH Dashboard for Security Monitoring. (Cryptus. 2025)

1.2. Algoritmos de búsqueda

Los algoritmos de búsqueda permiten encontrar información dentro de estructuras de datos. Son esenciales en bases de datos, sistemas operativos y herramientas de ciberseguridad.

1.2.1. Búsqueda lineal

La búsqueda lineal consiste en recorrer los datos uno por uno hasta encontrar el elemento buscado o hasta llegar al final del conjunto de datos.

Características:

- Funciona en datos ordenados y no ordenados.
- Es simple de implementar.

- Tiene complejidad O(n).

Aunque es fácil de usar, es poco eficiente para grandes volúmenes de datos, ya que debe revisar cada elemento. Estudios comparativos muestran que la búsqueda lineal es sencilla pero ineficiente en grandes conjuntos de datos.

En la figura 4 se muestra un ejemplo de una búsqueda lineal donde se realiza la comparación desde el inicio hasta encontrar el elemento buscado (9).

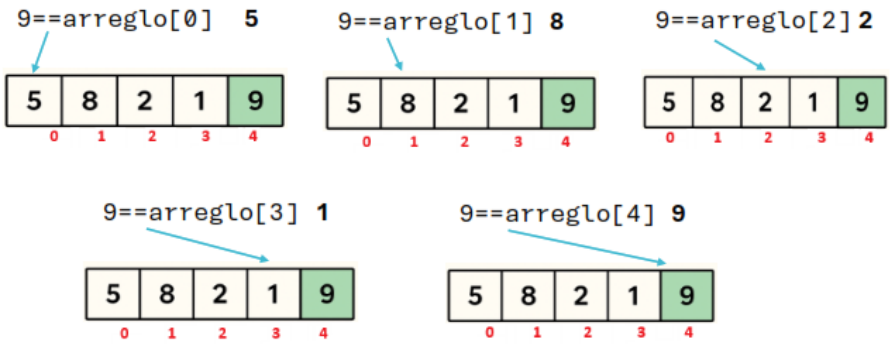


Figura 4. Ejemplo de búsqueda lineal. Creación de autor Patricio Coba

1.2.2. Búsqueda binaria

La búsqueda binaria es un método eficiente para encontrar elementos en listas ordenadas. Divide repetidamente el conjunto de datos a la mitad hasta encontrar el elemento.

Funcionamiento:

- Se toma el elemento central.
- Se compara con el valor buscado.
- Se descarta la mitad donde no puede estar el dato.

Tiene complejidad logarítmica, lo que la hace muy eficiente.
La búsqueda binaria puede ejecutarse en complejidad Θ (log n), lo que la convierte en una de las técnicas más eficientes para búsqueda en datos ordenados. En la figura 5 se muestra un ejemplo de una búsqueda binaria donde se realiza la comparación del elemento buscado (9) desde la mitad del conjunto de elementos y de ahí se toma el lado derecho por ser mayor ele elemento buscado (9) y se repite el proceso.

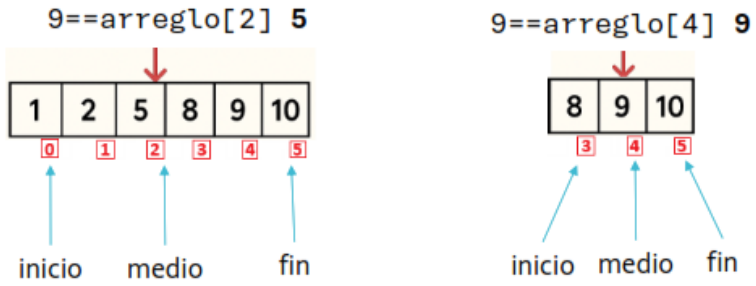


Figura 5. Ejemplo de búsqueda binaria. Creación de autor Patricio Coba

Algoritmos. Recurso proporcionado por Khan Academy (recurso gratuito) donde puede obtener más información sobre algoritmos, búsquedas y eficiencia de algoritmos. Enlace: https://es.khanacademy.org/computing/computer-science/algorithms

1.2.3. Requisitos de ordenamiento para búsqueda eficiente

Para aplicar búsqueda binaria, los datos deben estar ordenados previamente. Esto implica un costo adicional inicial de ordenamiento. Sin embargo, cuando se realizan muchas búsquedas sobre el mismo conjunto de datos, el costo del ordenamiento se compensa con la rapidez de las búsquedas posteriores.

1.2.4. Comparación de eficiencia entre métodos

En la Tabla 2 se muestra una comparación general de los métodos de búsqueda lineal y binaria:

Tabla 2. Comparativa entre métodos de búsqueda

Método	Complejidad (Peor Caso)	Complejidad (Mejor Caso)	Requisitos	Ventajas	Desventajas
Búsqueda lineal	O(n)	Ω(1)	Ninguno	Fácil de implementar. Funciona con datos ordenados o desordenados.	Poco eficiente en grandes volúmenes de datos.
Búsqueda binaria	O(log n)	Ω(1)	Datos previamente ordenados	Muy eficiente en grandes conjuntos de datos. Reduce rápidamente el espacio de búsqueda.	Requiere ordenamiento previo. Más compleja de implementar.

La búsqueda binaria es mucho más eficiente en grandes volúmenes de datos, mientras la lineal es útil en conjuntos pequeños o desordenados.

1.2.5. Aplicaciones en búsqueda de usuarios, IPs y eventos de seguridad

En ciberseguridad, los algoritmos de búsqueda se aplican en:

- Búsqueda de usuarios
- Verificación de autenticación.
- Control de accesos.
- Búsqueda de IPs
- Detección de direcciones sospechosas.
- Análisis de tráfico de red.
- Búsqueda en eventos de seguridad
- Análisis de logs.
- Detección de patrones de ataques.

Los sistemas modernos combinan algoritmos de búsqueda con estructuras optimizadas para manejar grandes volúmenes de datos en tiempo real.

En la Tabla 3 se simula una búsqueda de direcciones IP dentro de un registro de eventos de seguridad. En este ejemplo se busca la IP 192.168.1.25 dentro de un conjunto de datos.

Tabla 3. Ejemplo de simulación de búsqueda de direcciones IP.

ID Evento	Dirección IP	Usuario	Tipo de Evento	Fecha	Resultado
1	192.168.1.10	user01	Login correcto	15/2/2026 8:10	No coincide
2	192.168.1.15	user02	Intento fallido	15/2/2026 8:15	No coincide
3	192.168.1.20	user03	Login correcto	15/2/2026 8:20	No coincide
4	192.168.1.25	user04	Acceso remoto	15/2/2026 8:25	Coincidencia
5	192.168.1.30	user05	Cambio contraseña	15/2/2026 8:30	No coincide
6	192.168.1.35	user06	Intento fallido	15/2/2026 8:35	No coincide

Definición de los términos citados en la Semana 1

Algoritmo: Conjunto de pasos lógicamente ordenados, tal que, partiendo de ciertos datos o estados iniciales, permite obtener ciertos resultados o estados finales.
En ciberseguridad, con la aplicación de algoritmos se puede procesar información, analizar datos y ejecutar acciones automatizadas, como detectar accesos sospechosos, cifrar información o analizar patrones de ataques.

Búsqueda: Proceso mediante el cual se localiza información específica dentro de un conjunto de datos como arreglos o estructuras de datos dinámicas. Existen principalmente dos técnicas de búsqueda: secuencial y binaria.

En ciberseguridad, los algoritmos de búsqueda se utilizan para encontrar registros de usuarios, direcciones IP, firmas de malware o eventos sospechosos dentro de grandes volúmenes de información.

Educación de calidad

REFERENCIAS

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Algoritmos: Teoría y práctica. MIT Press / Pearson Educación.

González-Rodríguez Martín, (2023). Estructuras de Datos - Fundamentos y Aplicaciones. Universidad De Oviedo. https://hdl.handle.net/10651/74701

Joyanes Aguilar, L. (2003). Fundamentos de programación: algoritmos, estructuras de datos y objetos. Madrid, España: McGraw-Hill. ISBN: 84-481-3664-0.

Real-time SSH dashboard for security monitoring — Cryptsus blog. (n.d.). Cryptsus. https://cryptsus.com/blog/ssh-security-siem-dashboard-kibana.html

Rosas Soto, T., & Zollner I. Castellano. (2024). Espiral de multiplicación de Tesla, secuencia de raíz digital Tesla-Zollner, y su relación con los números primos y compuestos. Revista Bases de la Ciencia, 9(2), 14–24. https://doi.org/10.33936/revbasdelaciencia.v9i2.6693

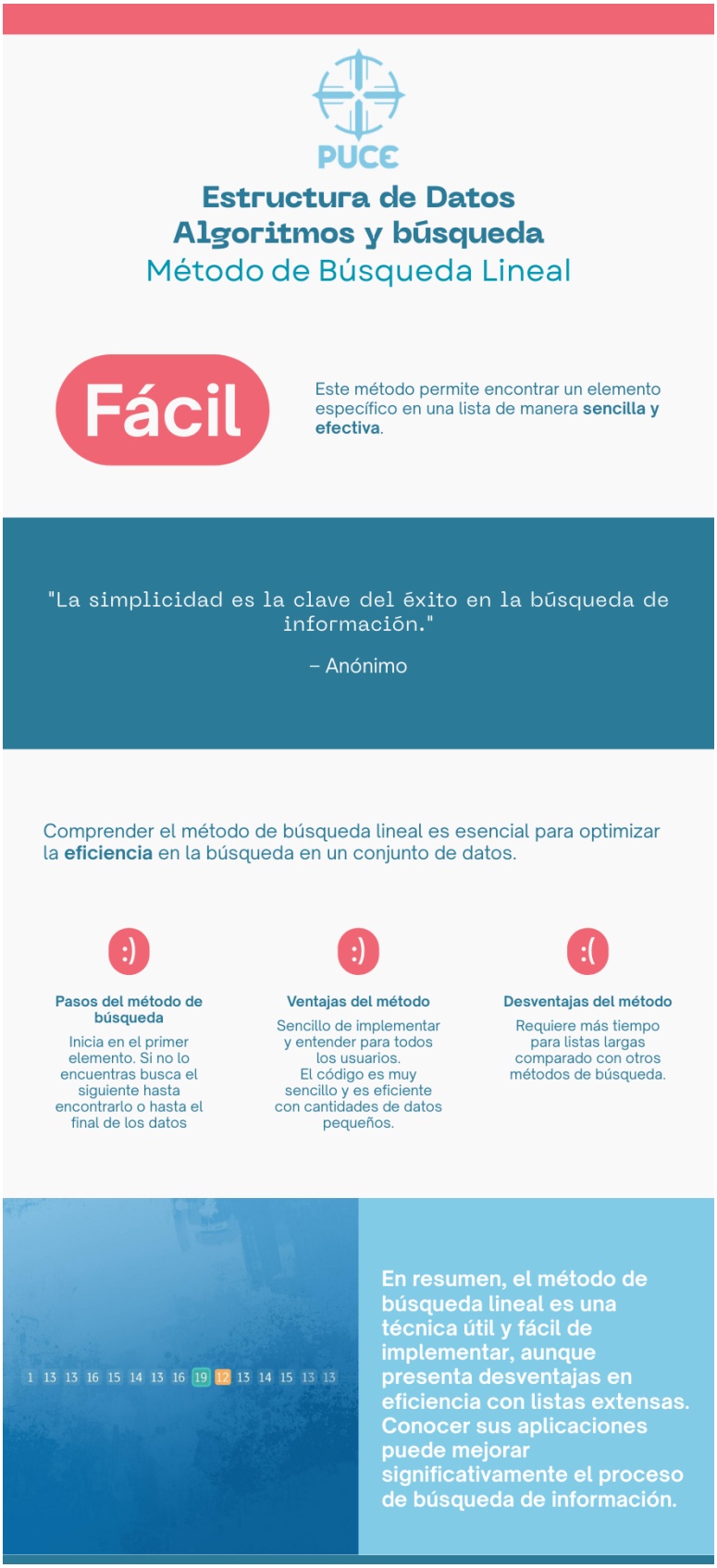
Stallings, W. (2021). Criptografía y seguridad en redes. Pearson Educación.

Tymoschuk, J., Guzmán, A., & Frittelli, V. (2021). Algoritmos y estructura de datos (2.ª ed.). Ariel Publisher. Disponible en: https://elibro.puce.elogim.com/es/ereader/puce/175249

Profundización Semana 1

Temas principales. Infografía sobre los temas principales de la semana 1, la cual tiene información básica de algoritmos, métodos de búsqueda y la aplicación de los algoritmos en Ciberseguridad.

Método de búsqueda lineal. Infografía sobre el método de búsqueda lineal (secuencial), características principales, ventajas y desventajas.



Laboratorio de contenido: Concepto de Algoritmo
Explicación básica aplicada sobre el concepto de algoritmo.

Objetivo
Comprender qué es un algoritmo mediante un caso cotidiano y su aplicación informática.

Caso práctico
Problema: Un sistema debe validar si un usuario puede ingresar a una plataforma.
Ejemplo paso a paso (Algoritmo básico)

1. Pedir usuario
2. Pedir contraseña
3. Verificar si existen en la base de datos
4. Si coinciden: permitir acceso
5. Si no coinciden: denegar acceso

Reflexión
Esto es un algoritmo porque:

- Tiene pasos ordenados
- Resuelve un problema
- Tiene inicio y fin
- Produce un resultado

Laboratorio de contenido: Método de Búsqueda Lineal

Explicación básica aplicada sobre el método de búsqueda lineal.

Objetivo
Comprender cómo funciona la búsqueda secuencial.

Caso práctico
Un analista debe buscar si una IP sospechosa está en un registro de accesos.
Datos de ejemplo
IP registradas:
192.168.1.10
192.168.1.15
192.168.1.20
192.168.1.25
192.168.1.30
IP buscada: **192.168.1.25**

Procedimiento
El sistema revisa una por una:

- Compara con 192.168.1.10: No
- Compara con 192.168.1.15: No

- Compara con 192.168.1.20: No
- Compara con 192.168.1.25: Sí

Conclusión

- Funciona con datos ordenados o no
- Fácil de implementar
- Lento con muchos datos

Laboratorio de Contenido: Método de Búsqueda Binaria

Explicación básica aplicada sobre el método de búsqueda binaria.

Objetivo
Comprender cómo funciona la búsqueda eficiente en datos ordenados.
Caso práctico
Buscar un usuario en una base de datos ordenada alfabéticamente.
Datos ordenados

Ana	Carlos	Daniel	Fernanda	Luis	María	Pedro	Sara	Renata	Miguel

Buscar: **Pedro**
Procedimiento

1. Tomar el centro: Luis
2. Pedro está después: buscar en mitad derecha
3. Nuevo centro: Sara
4. Pedro está antes: buscar izquierda
5. Resultado: Pedro encontrado

Conclusión

- Muy rápida en grandes datos
- Necesita datos ordenados

Laboratorio de Contenido: Notación Big-O

Explicación básica aplicada para entender la notación Big-O.

Objetivo
Entender cómo medir eficiencia de algoritmos.
Caso práctico
Comparar dos formas de buscar una IP.
Ejemplo
Si hay 1,000,000 registros:
Búsqueda Lineal
Puede revisar hasta 1,000,000 registros
Complejidad: O(n)

Búsqueda Binaria
Puede encontrar en aprox. 20 pasos
Complejidad: O(log n)
Interpretación
Big-O indica el peor escenario posible del algoritmo.

Laboratorio de Contenido: Notación Omega (Ω)

Explicación básica aplicada para entender la notación Omega (**Ω**)

Objetivo
Entender el mejor caso posible.
Caso práctico
Buscar IP en lista donde está en la primera posición.
Ejemplo
Si el dato está primero:

- Solo hace 1 comparación
- Ω(1)

Interpretación
Omega indica el mejor escenario.

Laboratorio de Contenido: Notación Theta (Θ)

Explicación básica aplicada para entender la notación Theta (**Θ**)

Objetivo
Comprender el comportamiento promedio.
Caso práctico
Buscar IP en posición media de lista.
Ejemplo
Si normalmente el dato está en la mitad:

- Θ(n/2) ≈ Θ(n)
- **Interpretación**

Theta describe el comportamiento típico.

Estudio de Caso 1: Detección de IP sospechosa en registros de acceso

En el estudio de caso presentado se muestra un ejemplo claro de cómo los algoritmos de búsqueda permiten identificar y localizar actividad sospechosa dentro de grandes volúmenes de información. Además, el uso de algoritmos de búsqueda eficientes permite optimizar los tiempos de análisis y mejorar la capacidad de respuesta ante incidentes. Esto resulta especialmente importante en entornos de ciberseguridad, donde es necesario procesar grandes cantidades de datos en tiempo real.

Contexto
Una empresa monitorea accesos a su servidor. El analista de seguridad necesita verificar si una dirección IP reportada como sospechosa aparece en los registros del día.
Datos disponibles
Registro de 5,000 direcciones IP almacenadas en un archivo de logs.
Problema
Buscar la IP: **185.43.21.90**
Solución aplicada
Se utiliza **búsqueda lineal**, porque:

- Los logs no están ordenados.
- Se agregan registros constantemente.

Funcionamiento
El sistema revisa cada registro hasta encontrar la IP o terminar la lista.
Resultado

- Se encuentra la IP en la posición 4,200.
- Se detectan 15 intentos de acceso fallidos.

Análisis

- Complejidad: O(n)
- Ventaja: No requiere ordenar datos
- Desventaja: Puede ser lenta en grandes volúmenes

Aprendizaje clave
La búsqueda lineal es útil cuando los datos cambian constantemente o no están organizados.

Estudio de Caso 2: Búsqueda rápida de usuario en base de datos corporativa

Ejemplo de cómo los algoritmos de búsqueda permiten localizar información de manera rápida y eficiente dentro de grandes bases de datos organizadas. En este caso, se utiliza la búsqueda binaria para encontrar un usuario dentro de millones de registros ordenados, lo que permite reducir significativamente el número de comparaciones necesarias y mejorar el tiempo de respuesta del sistema. Además, este caso demuestra la importancia de utilizar algoritmos eficientes cuando se trabaja con grandes volúmenes de datos y sistemas que requieren respuestas en tiempo real, como los sistemas de autenticación o verificación de accesos.

Contexto
Una empresa tiene una base de datos ordenada alfabéticamente con 2 millones de usuarios. El sistema debe verificar rápidamente si un usuario existe durante el login.
Datos disponibles
Lista ordenada de usuarios.
Problema
Buscar usuario: **mlopez**
Solución aplicada
Se usa **búsqueda binaria**, porque:

- Los datos están ordenados.
- Se realizan miles de búsquedas por minuto.

Funcionamiento

El sistema:

1. Busca el usuario del centro.
2. Descarta la mitad incorrecta.
3. Repite hasta encontrar coincidencia.

Resultado

- Usuario encontrado en 21 comparaciones.
- Tiempo de respuesta casi instantáneo.

Análisis

- Complejidad: O(log n)
- Ventaja: Muy rápida con grandes datos
- Desventaja: Requiere ordenamiento previo

Aprendizaje clave

La búsqueda binaria es ideal cuando se hacen muchas búsquedas sobre datos organizados.

Comparación entre ambos casos

Característica	Caso 1	Caso 2
Tipo de búsqueda	Lineal	Binaria
Datos ordenados	No	Sí
Volumen de datos	Medio	Muy grande
Velocidad	Media	Muy alta

PUCE Virtual

síguenos en:

www.pucevirtual.puce.edu.ec