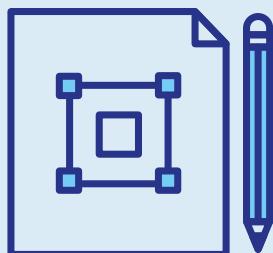




PENSAMIENTO ALGORÍTMICO
**ALGORITMOS EN LA
AUTOMATIZACIÓN DE PROCESOS
INDUSTRIALES**

ALGORITMOS EN LA AUTOMATIZACIÓN DE PROCESOS INDUSTRIALES

Anteriormente se definió al algoritmo como una secuencia de pasos lógicos finitos que tienen como objetivo realizar una tarea en específico; al tomar este concepto y llevarlo a la automatización industrial, podría hablarse de un conjunto de secuencias lógicas estructuradas que determinan un paso a paso de instrucciones, diseñadas con el objetivo de automatizar un sistema que realiza tareas repetitivas de manera eficiente y sin intervención humana.



La principal función de estos algoritmos, es la optimización de los procesos, incrementar la producción, reducir errores y mejorar la precisión. Su aplicación se enfoca en áreas como la robótica, control de calidad, inventarios, mantenimiento predictivo, operación de máquinas, etc.

En la unidad anterior, se abordaron los algoritmos básicos para dar a conocer su funcionalidad, aplicabilidad y lógica; ahora, para explicar los algoritmos aplicados en la automatización industrial, se amplía el espectro por las condiciones que se requieren para su implementación; a continuación, se definen cada uno de ellos.

Tabla 1. Tipos de algoritmos

| Descripción | Ejemplo | Aplicación |
|---|--|---|
| 1. Secuenciales | | |
| Tienen un orden de pasos fijos, uno a la vez, hasta terminar. | Realizar operaciones básicas, encender un equipo. | Procesos sencillos. |
| 2. Condicionales | | |
| Su operación está ligada a las condiciones y toma de decisiones ; de acuerdo con el resultado, el algoritmo toma uno de los dos caminos. | Si la temperatura es mayor a 25 grados, encender aire acondicionado. Si no, apagarlo. | Control de procesos industriales, climatización, gestión del tráfico. |
| 3. Iterativos | | |
| También conocidos como cíclicos , basan su funcionamiento en estructuras repetitivas hasta cumplir con el objetivo de la condición. | Semáforo que cada 60 segundos repite el ciclo: rojo – naranja – verde. | Cintas transportadoras, robots. |



| Descripción | Ejemplo | Aplicación |
|--|---|---|
| 4. Recursivos | | |
| Son capaces de invocarse a sí mismos, dentro del código. Se usan cuando se requiere convertir un problema, en subproblemas más pequeños. | Factorial de un número. | Procesamiento de gráficos, manipulación de datos. |
| 5. De búsqueda | | |
| <p>Su función dentro del código, es la de encontrar un elemento dentro de la estructura, como una lista o una base de datos.</p> <p>Lineal: uno por uno.</p> <p>Binaria: dividir en dos partes y buscar en la más relevante.</p> | <p>Dado un conjunto de números, queremos saber si el número 7 está presente en la lista: [3, 8, 4, 1, 7, 9]</p> <p>Pasos del algoritmo lineal:</p> <ol style="list-style-type: none">1. Empezar por el primer elemento.2. Comparar el elemento actual con el número buscado (7).3. Si es igual, detener el proceso y mostrar que se encontró el número.4. Si no es igual, pasar al siguiente elemento.5. Repetir hasta que se encuentre el número o se llegue al final de la lista. | Sistemas de inventarios, gestión de bases de datos. |
| 6. De ordenación | | |
| <p>Permiten organizar los elementos en orden ascendente o descendente.</p> <p>Quicksort: Divide para ordenar.</p> <p>BubbleSort: Compara y cambia, según sus adyacentes.</p> | <p>Ordenar la lista [5, 2, 9, 1, 5, 6] en orden ascendente.</p> <p>Pasos para el algoritmo <i>BubbleSort</i>:</p> <ol style="list-style-type: none">1. Comparar el primer elemento con el segundo.2. Si el primer elemento es mayor, intercambiarlos.3. Repetir el proceso para todos los pares adyacentes de la lista.4. Al finalizar una pasada completa, el elemento más grande estará al final de la lista.5. Repetir los pasos hasta que no se necesiten más intercambios. | Gestión de inventarios, <i>ranking</i> de resultados. |



| Descripción | Ejemplo | Aplicación |
|--|--|--|
| 7. Probabilísticos | | |
| O aleatorios , utilizan elementos de azar para resolver problemas y entregan soluciones, aunque no siempre son las más acertadas. | Algoritmo de Montecarlo: “utilización de simulaciones aleatorias para resolver problemas complejos”. | Predicciones financieras, simulaciones científicas. |
| 8. Ávidos | | |
| También conocidos como Greedy , toman decisiones óptimas en cada paso del proceso sin “preocuparse” por las decisiones que siguen; siempre buscan encontrar la solución óptima. | Algoritmo de cambio mínimo: “dar la menor cantidad de monedas posible”. | Optimización de rutas en logística, redes y telecomunicaciones. |
| 9. Genéticos | | |
| Toman sus orígenes en la evolución natural, buscan optimizar la solución mediante procesos como selección, mutación y cruce entre soluciones. | Optimización del diseño de una pieza mecánica, mediante simulación evolutiva. | IA, robótica, diseño industrial. |
| 10. Paralelos | | |
| Ejecutan al tiempo varias tareas en diferentes procesadores, lo que se traduce como una reducción en los tiempos de ejecución. | Procesamiento simultáneo de datos en supercomputadoras. | <i>Big Data</i> , procesamiento de imágenes, simulaciones científicas. |



A continuación, encontrará dos pseudocódigos que utilizan algún tipo de algoritmo de los anteriormente descritos; el **desafío** consiste en **identificar cuál tipo de algoritmo se utilizó** para encontrar la solución.

| Algoritmos | |
|--|---|
| <p>Entrada. Lista de números [3, 8, 4, 1, 7, 9], número a buscar = 7.</p> <p>Para cada elemento en la lista:</p> <ul style="list-style-type: none">Si el elemento es igual a 7:<ul style="list-style-type: none">Imprimir "Número encontrado"DetenerSi no:<ul style="list-style-type: none">Continuar con el siguiente elemento. <p>Si se recorre toda la lista sin encontrar el número:</p> <ul style="list-style-type: none">Imprimir "Número no encontrado". | <p>Entrada. Lista = [5, 2, 9, 1, 5, 6]</p> <p>Repetir para i desde 0 hasta longitud(lista) - 2:</p> <ul style="list-style-type: none">Para cada j desde 0 hasta longitud(lista) - 2 - i:<ul style="list-style-type: none">Si lista[j] > lista[j + 1]:<ul style="list-style-type: none">Intercambiar lista[j] y lista[j + 1] <p>Imprimir lista ordenada.</p> |
| <p>Resultado del ejemplo:</p> <p>Al recorrer la lista [3, 8, 4, 1, 7, 9], el algoritmo encuentra el número 7 en la posición 4 (índice 3 si contamos desde 0) y se detiene.</p> | <p>Simulación:</p> <p>Primera pasada: [2, 5, 1, 5, 6, 9].</p> <p>Segunda pasada: [2, 1, 5, 5, 6, 9].</p> <p>Tercera pasada: [1, 2, 5, 5, 6, 9].</p> <p>Después de varias pasadas, la lista queda ordenada como [1, 2, 5, 5, 6, 9].</p> |
| Búsqueda Lineal | Ordenación por <i>BubbleSort</i> |

Cada uno de los algoritmos anteriormente descritos, tiene su campo de aplicabilidad, dependiendo del problema que se desea resolver; en el contexto de los **algoritmos de procesos industriales**, se usan con mayor frecuencia los algoritmos **iterativos, recursivos** y de **optimización**, debido a que permiten operar los sistemas sin intervención humana.