
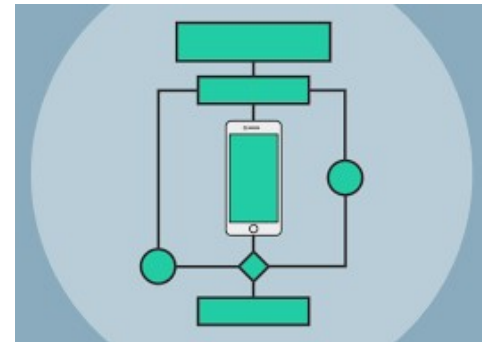


TEMA 2: Análisis de algoritmos.

OBJETIVO: EL ALUMNO APLICARÁ DIVERSAS TÉCNICAS PARA EL ANÁLISIS Y EL DISEÑO DE ALGORITMOS ORIENTADOS A LA SOLUCIÓN DE PROBLEMAS COMPUTACIONALES.





2.1 Fundamentos de Algoritmia

2.1 Fundamentos de Algorítmica

¿Qué es la algoritmia?

¿Para qué estudiar temas de teoría de la computación?

- Una formación más sólida, un ingeniero más capaz.
- Desarrollo y uso de herramientas complejas.
- Seguir adelante a un posgrado.
- Dedicarse a la teoría de la computación (redes de datos, complejidad, seguridad, etc.) en investigación y docencia.



2.1 Fundamentos de Algorítmica

- El objetivo principal de esta rama de las Ciencias de la Computación es analizar algoritmos
- Mediante este análisis se puede estimar la cantidad de recursos que se necesitan para ejecutarlo.
- La mayoría de los algoritmos se diseñan para trabajar con entradas de longitud arbitraria.

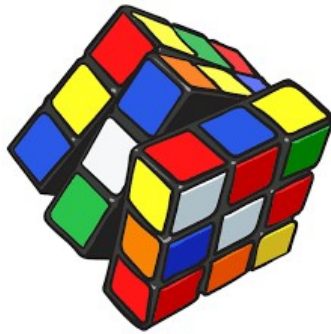


2.1 Fundamentos de Algorítmica

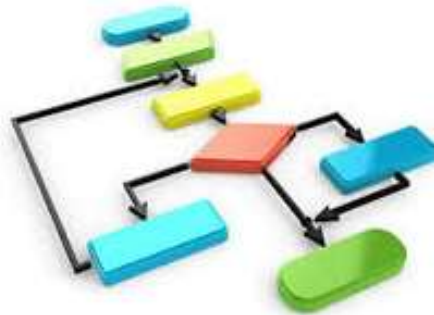
- Usualmente la eficiencia de un algoritmo se expresa como una función matemática que relaciona la longitud de la entrada con el tiempo de ejecución.
- Para ello consideran diferentes factores:
 - Número de instrucciones
 - Cantidad de memoria
 - Operaciones significativas de las instrucciones.

2.1 Fundamentos de Algorítmica

- El punto de partida es la necesidad de resolver problemas y de encontrar estrategias óptimas para resolverlos.



Problema



Algoritmo



Programa

2.1.1 Definición de algoritmo

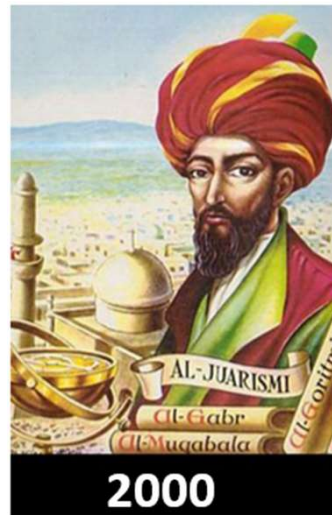
¿Qué es un algoritmo?

- Es un conjunto finito de instrucciones (pasos) libres de ambigüedades que sirven para realizar una tarea específica.
- Un programa computacional es la representación de un algoritmo



2.1.1 Definición de algoritmo

- El término “algoritmo” se debe al matemático persa Al-Khowarizmi cuyas principales aportaciones están en la aritmética, astronomía, geografía, etc.



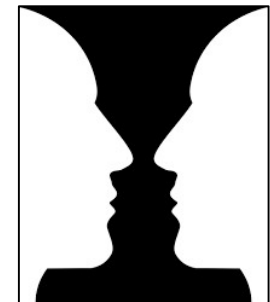
2.1.1 Definición de algoritmo

- Computacionalmente hablando, la teoría de algoritmos inicia con la definición de “**Entscheidungsproblem**”
- Alonzo Church (en 1936) con el cálculo lambda y por Alan Turing con máquinas de Turing resolvieron este problema



2.1.1 Definición de algoritmo

- Un algoritmo computacional debe cumplir con tres requisitos.
 - Descripción exacta de las instrucciones, de tal forma que cualquiera pueda tener éxito al ejecutarlas, incluso si no tiene idea del objetivo del algoritmo.
 - Debe ser libre de ambigüedades excluyendo interpretaciones diferentes.



2.1.1 Definición de algoritmo

- Un algoritmo computacional debe cumplir con tres requisitos.
 - No importa en qué momento del tiempo se ejecute, el resultado debe ser el mismo cada vez que se aplique.



2.1.1 Definición de algoritmo

- Un programa es una secuencia de instrucciones representadas en forma que sean posibles de ejecutar en una computadora.
- Por lo tanto, la programación puede verse como la actividad de “reescribir” (representar, plasmar) algoritmos en instrucciones de algún lenguaje de programación.

2.1.1 Definición de algoritmo

¿Cómo diseñar un algoritmo?

- Identificar el problema
- Establecer: entradas, procedimientos, salidas.
- Describir los procedimientos de manera detallada
- Plasmar esa descripción en una representación convencional



2.1.2 Representación de algoritmos

Cuando se desarrollan algoritmos computacionales, el objetivo es que posteriormente se pueda escribir un programa a partir de ese algoritmo.

Existen dos formas de representar algoritmos computacionales.

- Pseudocódigo
- Diagramas de Flujo

2.1.2.1 Pseudocódigo

- El pseudocódigo, es un tipo de lenguaje “pseudonatural” estructurado utilizado para describir algoritmos computacionales.
- Es una descripción compacta de alto nivel de un algoritmo que combina lenguaje natural con algunas convenciones sintácticas similares a los lenguajes de programación

Algorithm (noun.)

Word used by programmers when...
they do not want to explain what they did.

2.1.2.1 Pseudocódigo

- El objetivo es que sea entendido por personas, sin embargo debe ser lo más completo posible.
- No existe un estándar para escribir pseudocódigos, sin embargo se pueden establecer mejores prácticas para su escritura



2.1.2.1 Pseudocódigo




Buenas prácticas del pseudocódigo

- Verbos en infinitivo.
- Enumerar las instrucciones.
- Distinguir órdenes y estructuras de control tipo lenguaje de programación del resto de los elementos del algoritmo, usando mayúsculas o negritas.
- Establecer distinción entre asignación y comparación .
- Utilizar sangrías correctamente para los distintos bloques de código.

2.1.2.2 Diagrama de Flujo

- Un diagrama de flujo (flowchart) es la representación gráfica de un algoritmo; se compone de símbolos estandarizados que representan acciones específicas.
- El estándar más utilizado está definido por la International Organization for Standardization (ISO) en el estándar ISO5807:1985

2.1.2.2 Diagrama de Flujo

Simbología Estándar: Diagramas de Flujo de Datos.	
Inicio / Fin	
Procesos	
Entrada Datos	
Condición	
Conector	

Más: <https://www.smartdraw.com/flowchart/simbolos-de-diagramas-de-flujo.htm>

2.1.2.2 Diagrama de Flujo

Buenas prácticas para diagramas de flujo

Además de utilizar símbolos estandarizados en los diagramas de flujo, existen otros elementos que facilitan la elaboración y la comprensión de los mismos:

- Nombre del algoritmo
- Fecha de creación o actualización (identificación de versiones).
- Nombre de autor(es) del diagrama



2.1.2.2 Diagrama de Flujo

Buenas prácticas para diagramas de flujo

- Puntos de inicio y fin claros para facilitar el recorrido del diagrama
- Dirección de flujo clara. (arriba -> abajo, izquierda -> derecha)
- En caso de utilizar un símbolo no estandarizado, especificar claramente su uso