Argentina programa 4.0

# Conceptos Computación

## Algoritmos

# Ciencias de la computación

**Es la disciplina que estudia como resolver problemas con computadoras**

**Deriva de las Matemáticas**

- **Resolución de problemas**
- **Algoritmos**

## Qué es un Algoritmo?

**Informalmente:** *"una serie de reglas que definen en forma precisa una secuencia de operaciones"*

# Qué es un algoritmo?

## Un algoritmo puede ser especificado

- En inglés, español, francés, etc.
- En un lenguaje formal: matemático, o de programación
    - *C, Java, Perl, Python* 🐍
    - En forma de un diseño de hardware

## A qué se parece?

- A una receta de cocina
    - Input = ingredientes + materiales
    - **Algoritmo = receta (instrucciones, pasos, operaciones, parámetros)**
    - Output = comida

# Programas vs Algoritmos

Un algoritmo es nuestra ***receta*** para resolver un problema.
Si escribimos esa receta en Python, tenemos un ***programa.***
Si escribimos esa receta en Java, tenemos ***otro programa***.
Si re-escribimos nuestro programa en Python para mejorarlo, tenemos ***otro programa***.

Un ***programa*** es una implementación de un algoritmo!
Puede haber varias implementaciones!

# Problemas computacionales

Tienen que cumplir con las siguientes condiciones

- Tiene que estar bien definido
- Tiene que tener una solución
- Tiene que ser genérico

***Un problema es una colección infinita de instancias, junto con una solución para cada una de esas instancias***

Ejemplo: ordenar números en forma creciente

- Input: una secuencia de n números (a1, a2, ..., an)
- Output: una permutación de la secuencia original (a1', a2', ..., an') tal que a1' > a2' > ... > an'

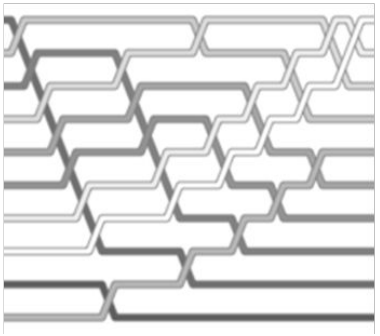
Una instancia del problema:

- Input: (31, 41, 59, 26, 41, 58)
- Output: (26, 31, 41, 41, 58, 59)

Hay muchas maneras de ordenar datos!

**Algoritmos para ordenar:**

- Insertion sort
- Merge sort
- Selection sort
- Bubble sort
- …



Cuál es el mejor?

**Hay que analizar el algoritmo!**

# Wikipedia es un libro de algoritmos!

ikipedia.org/wiki/Insertion_sort

Apture Editor | Altmetric it | Scoop.it! | Spotify Web Player | Open Access Button | Other bookmarks

## Insertion sort

From Wikipedia, the free encyclopedia

**Insertion sort** is a simple sorting algorithm that builds the final sorted array (or list) one item at a time. It is much less efficient on large lists than more advanced algorithms such as quicksort, heapsort, or merge sort. However, insertion sort provides several advantages:

- Simple implementation
- Efficient for (quite) small data sets
- Adaptive (i.e., efficient) for data sets that are already substantially sorted: the time complexity is $O(n + d)$, where $d$ is the number of inversions
- More efficient in practice than most other simple quadratic (i.e., $O(n^2)$) algorithms such as selection sort or bubble sort; the best case (nearly sorted input) is $O(n)$
- Stable; i.e., does not change the relative order of elements with equal keys
- In-place; i.e., only requires a constant amount O(1) of additional memory space
- Online; i.e., can sort a list as it receives it

When people manually sort something (for example, a deck of playing cards), most use a method that is similar to insertion sort.[1]

Contents [hide]



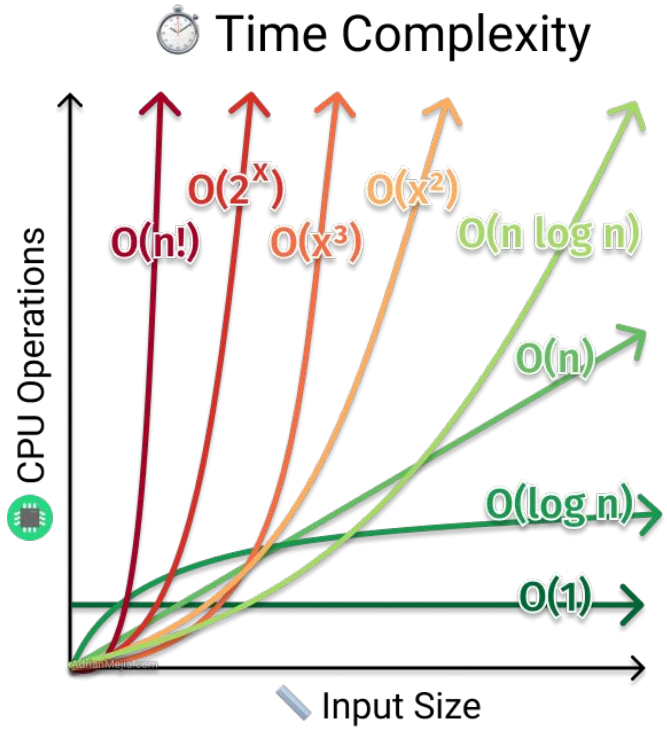
**Insertion sort**

Graphical illustration of insertion sort

| | |
|---|---|
| **Class** | Sorting algorithm |
| **Data structure** | Array |
| **Worst case performance** | $O(n^2)$ comparisons, swaps |
| **Best case performance** | $O(n)$ comparisons, $O(1)$ swaps |
| **Average case performance** | $O(n^2)$ comparisons, swaps |
| **Worst case space complexity** | $O(n)$ total, $O(1)$ auxiliary |

# Complejidad de algoritmos

- Es una métrica para comparar algoritmos
- Mide relación entre cantidad de datos de entrada (input) y alguna otra variable relacionada al costo computacional
  - tiempo
  - espacio
  - CPU
  - RAM



# Complejidad

- Además de la relación con la cantidad de datos (input)
  - $n$, $\log n$, $n^2$, $n^3$, $2^n$, $n!$
- Normalmente se especifica para
  - **θ, mejor caso (best case)**
  - **□, caso promedio (average case)**
  - **O, peor caso (worst case)**

# Problema ejemplo

Dado un conjunto de números, existe algún subconjunto cuya suma sea ***N?***

Una instancia del problema:

- **Conjunto:** [1, 2, 4, 5, 8, 9, 10, 11, 23, 76, 89]
- **N:** 119

# Qué hacemos?

## Evaluar y Analizar

1. **Viabilidad**
   a. Existen impedimentos teóricos?
   b. Responder SI/NO
2. **Solución (algoritmo)**
   a. Tomar subconjuntos y evaluar la suma
   b. Responder SI/NO
3. **Complejidad**
   a. Cómo escala con el input?