

# Taller de Algoritmos y Estructura de Datos

Taller: Ordenación

June 18, 2025

## 1 Ejercicio 1: Insertion Sort

En el directorio **ej1** vas a encontrar algunos archivos sobre los que vas a tener que trabajar.

- **ArrayHelpers.java**: Contiene prototipos y descripciones de métodos auxiliares para manipular/trabajar con arreglos.
- **SortHelpers.java**: Contiene los métodos **goesBefore**, **swap**, **arrayIsSorted**, **insert** y **insertionSort**
- **Main.java**: Contiene el programa principal que carga un array de números, luego lo ordena con la función **insertionSort()** y finalmente comprueba que el arreglo sea permutación ordenada del que se cargó inicialmente.

### 1.1 Parte A: Ordenación por Inserción

Vas a hacer una implementación del algoritmo de ordenación por inserción. Para esta parte es necesario que abras el archivo **SortHelpers.java** e implementes el procedimiento **insert()**. Para guiarte, no dudes en examinar el resto del archivo **SortHelpers.java** y la definición del algoritmo de ordenación por inserción. El algoritmo debe ordenar con respecto a la relación **goesBefore()**, provista en **SortHelpers.java**.

### 1.2 Parte B: Chequeo de Invariante

Aquí será necesario que modifiques el procedimiento **insertionSort()** agregando la verificación de cumplimiento de la invariante. Por simplicidad solo verificaré la siguiente parte del Invariante:

- El segmento inicial **a[0,i)** del arreglo está ordenado.

Para esto debes hacer uso de las funciones **assert()** y **arrayIsSorted()**. Una vez implementado los incisos a) y b), compilá ejecutando: **javac \*.java** y ya puedes correr el programa ejecutando: **java Main ../input/example-unsorted.in**

## 2 Ejercicio 2: Quick Sort

En este ejercicio vas a hacer una implementación del algoritmo de ordenación rápida vista en el teórico. En la carpeta ej2 se encuentran los siguientes archivos:

- `ArrayHelpers.java`: Contiene prototipos y descripciones de métodos auxiliares para manipular/trabajar con arreglos.
- `SortHelpers.java`: Contiene los métodos `goesBefore`, `swap`, `arrayIsSorted`, `partition`, `quickSortRec` y `quickSort`
- `Main.java`: Contiene el programa principal que carga un array de números, luego lo ordena con la función `quickSort()` y finalmente comprueba que el arreglo sea permutación ordenada del que se cargó inicialmente.

### 2.1 Parte A: Implementacion de Algoritmos

Implementará todos los procedimientos del archivo `SortHelpers.java` para una implementación completa del algoritmo de ordenación Quick Sort.

### 2.2 Parte B: Función `main()`

Para esta parte es necesario que abras el archivo `Main.java` y completes la función `main()` con una llamada al procedimiento `quickSort()`. Una vez implementado los incisos a) y b), compilá ejecutando: `javac *.java` y ya podés correr el programa ejecutando: `java Main ../input/example-unsorted.in`

## 3 Ejercicio 3: Refactorizar el Código (Opcional)

Opcionalmente pueden refactorizar el código de los ejercicios 1 y 2. Aquí pueden agregar clases para separar mejor la lógica. Pueden extender con funcionalidades que indiquen como quiero ordenar, de forma creciente, decreciente, etc.