

## **Codificacion de algoritmo**

Bryan Joaquin Alao Huiracocha

Facultad de Ingeniería y Arquitectura, Universidad Técnica Particular de Loja

Análisis de algoritmos

29 de mayo de 2025

Realice la codificación del siguiente algoritmo:

```
MERGE( $A, p, q, r$ )
1   $n_L = q - p + 1$            // length of  $A[p:q]$ 
2   $n_R = r - q$                // length of  $A[q + 1:r]$ 
3  let  $L[0:n_L - 1]$  and  $R[0:n_R - 1]$  be new arrays
4  for  $i = 0$  to  $n_L - 1$  // copy  $A[p:q]$  into  $L[0:n_L - 1]$ 
5       $L[i] = A[p + i]$ 
6  for  $j = 0$  to  $n_R - 1$  // copy  $A[q + 1:r]$  into  $R[0:n_R - 1]$ 
7       $R[j] = A[q + j + 1]$ 
8   $i = 0$                      //  $i$  indexes the smallest remaining element in  $L$ 
9   $j = 0$                      //  $j$  indexes the smallest remaining element in  $R$ 
10  $k = p$                      //  $k$  indexes the location in  $A$  to fill
11 // As long as each of the arrays  $L$  and  $R$  contains an unmerged element,
12 //   copy the smallest unmerged element back into  $A[p:r]$ .
13 while  $i < n_L$  and  $j < n_R$ 
14     if  $L[i] \leq R[j]$ 
15          $A[k] = L[i]$ 
16          $i = i + 1$ 
17     else  $A[k] = R[j]$ 
18          $j = j + 1$ 
19          $k = k + 1$ 
20 // Having gone through one of  $L$  and  $R$  entirely, copy the
21 //   remainder of the other to the end of  $A[p:r]$ .
22 while  $i < n_L$ 
23      $A[k] = L[i]$ 
24      $i = i + 1$ 
25      $k = k + 1$ 
26 while  $j < n_R$ 
27      $A[k] = R[j]$ 
28      $j = j + 1$ 
29      $k = k + 1$ 
```

```
public class MergeSort {
    public static void merge(int[] A, int p, int q, int r) {
        int nL = q - p + 1;
        int nR = r - q;

        int[] L = new int[nL];
        int[] R = new int[nR];

        for (int i = 0; i < nL; i++) {
            L[i] = A[p + i];
        }

        for (int j = 0; j < nR; j++) {
            R[j] = A[q + 1 + j];
        }

        int i = 0, j = 0;
        int k = p;

        while (i < nL && j < nR) {
            if (L[i] <= R[j]) {
                A[k] = L[i];
                i++;
            } else {
                A[k] = R[j];
                j++;
            }
            k++;
        }

        while (i < nL) {
            A[k] = L[i];
            i++;
            k++;
        }

        while (j < nR) {
            A[k] = R[j];
            j++;
            k++;
        }
    }
}
```

```

        i++;
        k++;
    }

    while (j < nR) {
        A[k] = R[j];
        j++;
        k++;
    }
}

public static void mergeSort(int[] A, int p, int r) {
    if (p < r) {
        int q = (p + r) / 2;
        mergeSort(A, p, q);
        mergeSort(A, q + 1, r);
        merge(A, p, q, r);
    }
}

public static void main(String[] args) {
    int[] arreglo = {38, 27, 43, 3, 9, 82, 10};
    System.out.println("Arreglo original:");
    for (int num : arreglo) System.out.print(num + " ");

    mergeSort(arreglo, 0, arreglo.length - 1);

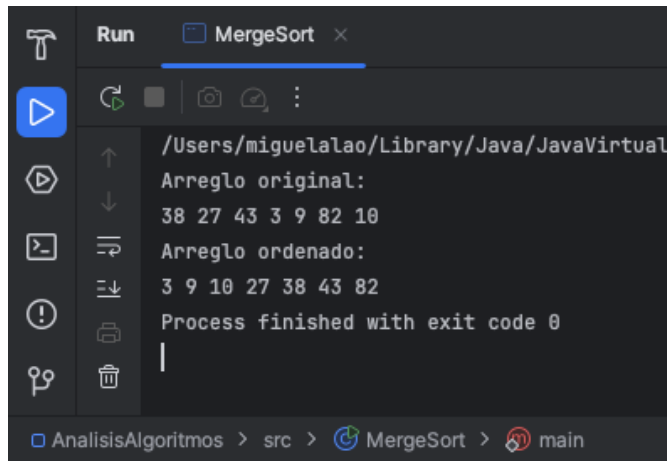
    System.out.println("\nArreglo ordenado:");
    for (int num : arreglo) System.out.print(num + " ");
}
}

```

## Descripción del Desarrollo

1. **Análisis del Pseudocódigo:** El pseudocódigo proporciona claramente la lógica de cómo dividir dos subarreglos (izquierdo y derecho) y fusionarlos ordenadamente en el arreglo original.
2. **Traducción a Java:**
  - ⇒ Se crearon arreglos temporales *L* y *R* para contener las mitades del arreglo original.
  - ⇒ Se usaron bucles *for* para copiar los datos en estos subarreglos.
  - ⇒ Se usó un bucle *while* para comparar y fusionar los elementos de forma ordenada.
  - ⇒ Finalmente, se copian los elementos restantes de *L* o *R* si aún quedan elementos por insertar.
3. **Validación y prueba:** Se incluyó una función *main* que prueba con un arreglo de ejemplo y muestra el antes y después de la ordenación.

## Resultado:



The screenshot shows the 'Run' console of an IDE. The title bar indicates the file 'MergeSort'. The console output is as follows:

```
/Users/miguelalao/Library/Java/JavaVirtual  
Arreglo original:  
38 27 43 3 9 82 10  
Arreglo ordenado:  
3 9 10 27 38 43 82  
Process finished with exit code 0  
|
```

The breadcrumb at the bottom of the console shows the file path: `AnalysisAlgoritmos > src > MergeSort > main`.