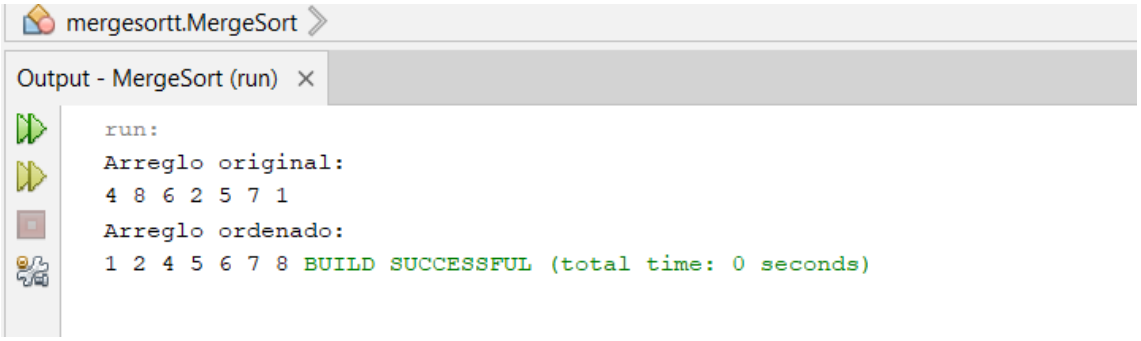
**Taller: Codificación de algoritmo**

**Nombre:** Valentina Samaniego

```java
package mergesortt;

public class MergeSort {
    // Método para ordenar un arreglo usando Merge Sort
    public static void mergeSort(int[] A, int p, int r) {
        if (p < r) {
            int q = (p + r) / 2;
            mergeSort(A, p, r: q);
            mergeSort(A, q + 1, r);
            merge(A, p, q, r);
        }
    }

    // Método merge basado en el algoritmo de la imagen
    public static void merge(int[] A, int p, int q, int r) {
        int nL = q - p + 1;  // longitud de A[p:q]
        int nR = r - q;      // longitud de A[q+1:r]

        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, k = p;

        // Mezclar elementos mientras haya en ambas listas
        while (i < nL && j < nR) {
            if (L[i] <= R[j]) {
                A[k] = L[i];
                i++;
            } else {
                A[k] = R[j];
                j++;
            }
            k++;
        }

        // Copiar cualquier elemento restante en L
        while (i < nL) {
            A[k] = L[i];
            i++;
            k++;
        }

        // Copiar cualquier elemento restante en R
        while (j < nR) {
            A[k] = R[j];
            j++;
            k++;
        }
    }

    // Método principal para probar el algoritmo
    public static void main(String[] args) {
        int[] arreglo = {4, 8, 6, 2, 5, 7, 1};

        System.out.println(x: "Arreglo original:");
```

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

**SALIDA POR PANTALLA**




**CODIGO TEXTO:**

```
package mergesortt;

public class MergeSort {

  // Método para ordenar un arreglo usando Merge Sort
  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);
    }
  }


  // Método merge basado en el algoritmo de la imagen
  public static void merge(int[] A, int p, int q, int r) {
    int nL = q - p + 1;  // longitud de A[p:q]
    int nR = r - q;      // longitud de A[q+1:r]
    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, k = p;
    // Mezclar elementos mientras haya en ambas listas
    while (i < nL && j < nR) {
```

```
            if (L[i] <= R[j]) {
                A[k] = L[i];
                i++;
            } else {
                A[k] = R[j];
                j++;
            }
            k++;
        }
        // Copiar cualquier elemento restante en L
        while (i < nL) {
            A[k] = L[i];
            i++;
            k++;
        }
        // Copiar cualquier elemento restante en R
        while (j < nR) {
            A[k] = R[j];
            j++;
            k++;
        }
    }
    // Método principal para probar el algoritmo
    public static void main(String[] args) {
        int[] arreglo = {4, 8, 6, 2, 5, 7, 1};
        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 + " ");
        }
    }
}
```