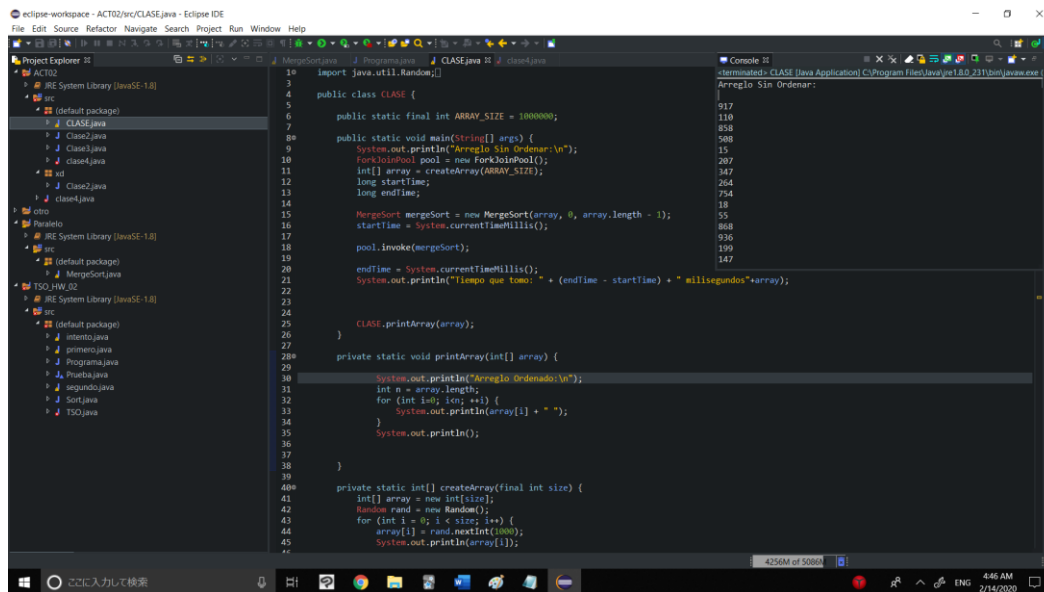


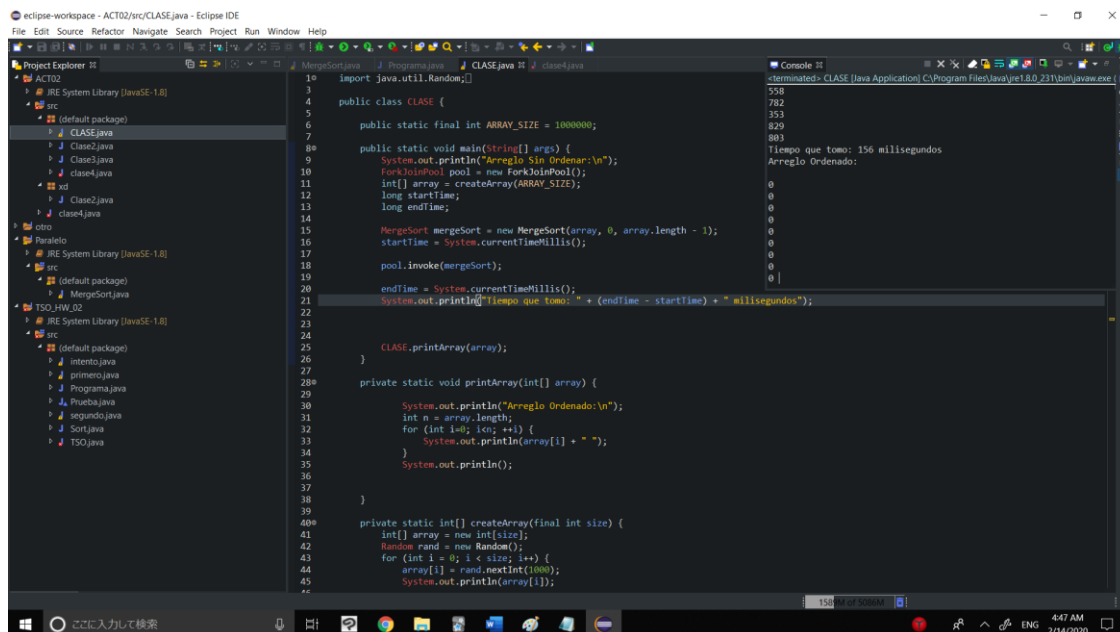
EJEMPLO

En este programa de Java se crea un arreglo de tamaño: 1000000(Un millon) con numeros aleatorios de entre 0 a 999, se procede a ordenarlo mediante un sistema paralelo, gracias a esto nos da un tiempo muy rapido de solamente 156 milisegundos.

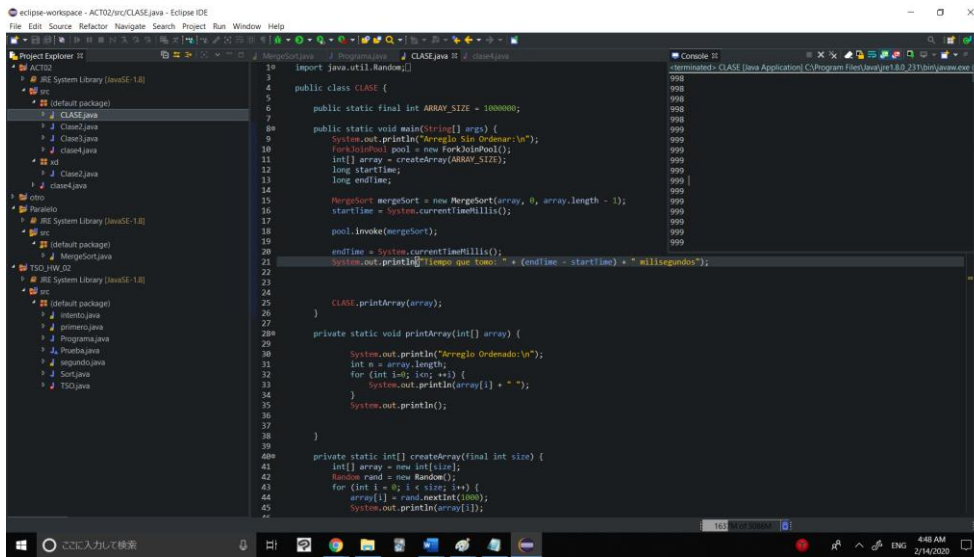


```
10 import java.util.Random;
11
12 public class CLASE {
13     public static final int ARRAY_SIZE = 1000000;
14
15     public static void main(String[] args) {
16         System.out.println("Arreglo Sin Ordenar:\n");
17         ForkJoinPool pool = new ForkJoinPool();
18         int[] array = createArray(ARRAY_SIZE);
19         long startTime;
20         long endTime;
21         MergeSort mergeSort = new MergeSort(array, 0, array.length - 1);
22         startTime = System.currentTimeMillis();
23         pool.invoke(mergeSort);
24         endTime = System.currentTimeMillis();
25         System.out.println("Tiempo que tomo: " + (endTime - startTime) + " milisegundos"+array);
26     }
27
28     private static void printArray(int[] array) {
29         System.out.println("Arreglo Ordenado:\n");
30         int n = array.length;
31         for (int i=0; i<n; ++i) {
32             System.out.println(array[i] + " ");
33         }
34         System.out.println();
35     }
36
37     private static int[] createArray(final int size) {
38         int[] array = new int[size];
39         Random rand = new Random();
40         for (int i = 0; i < size; i++) {
41             array[i] = rand.nextInt(1000);
42             System.out.println(array[i]);
43         }
44     }
45 }
```

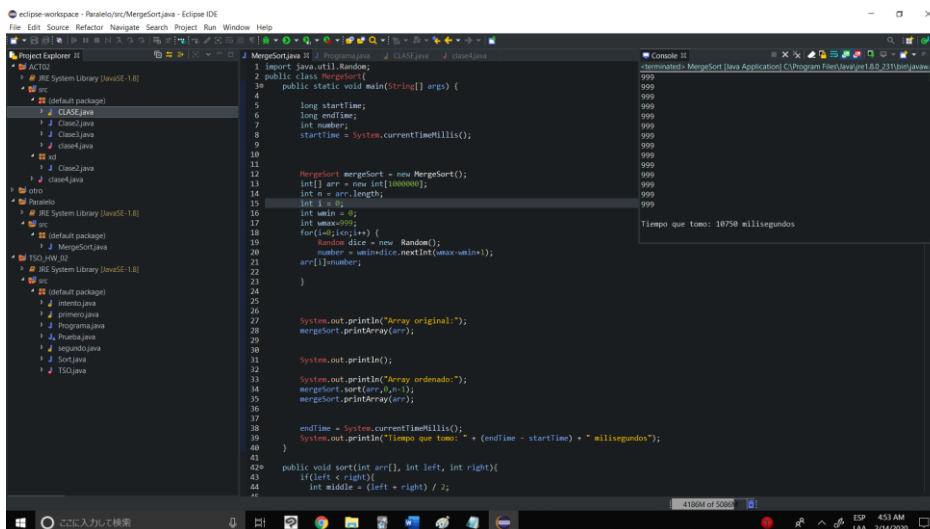
En la captura de abajo se puede ver el tiempo de 156 milisegundos



```
10 import java.util.Random;
11
12 public class CLASE {
13     public static final int ARRAY_SIZE = 1000000;
14
15     public static void main(String[] args) {
16         System.out.println("Arreglo Sin Ordenar:\n");
17         ForkJoinPool pool = new ForkJoinPool();
18         int[] array = createArray(ARRAY_SIZE);
19         long startTime;
20         long endTime;
21         MergeSort mergeSort = new MergeSort(array, 0, array.length - 1);
22         startTime = System.currentTimeMillis();
23         pool.invoke(mergeSort);
24         endTime = System.currentTimeMillis();
25         System.out.println("Tiempo que tomo: " + (endTime - startTime) + " milisegundos");
26     }
27
28     private static void printArray(int[] array) {
29         System.out.println("Arreglo Ordenado:\n");
30         int n = array.length;
31         for (int i=0; i<n; ++i) {
32             System.out.println(array[i] + " ");
33         }
34         System.out.println();
35     }
36
37     private static int[] createArray(final int size) {
38         int[] array = new int[size];
39         Random rand = new Random();
40         for (int i = 0; i < size; i++) {
41             array[i] = rand.nextInt(1000);
42             System.out.println(array[i]);
43         }
44     }
45 }
```



En el codigo llamado: “Sin sistema parello” se hizo exactamente lo mismo pero sin el uso de un sistema parello y se puedde ver que el tiempo que tardo fue: 10750 milisegundos



A la izquierda (Con sistema paralelo) a la derecha (Sin sistema paralelo)

