|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TestCombSort** | **Objetivo de la prueba:** Comprobar que el método **CombSort** ordena el conjunto de datos de manera ascendente    **Firma del método: public void CombSort(int[] arr)** | | | |
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| Ordenamientos | CombSort(int[] arr) | Arreglo de tamaño **10^4.** Usando el método private void fillArray(int[] a)  llenamos el arreglo con datos aleatorios. | Arreglo a ordenar | El arreglo esta ordenado de forma ascendente. Siendo el arreglo **A**, como esta ordenado ascendentemente, es correcto afirmar que la resta **(A[i] – A[i + 1])** tendrá valor negativo o cero. |
| Ordenamientos | CombSort(int[]) arr | Arreglo de tamaño **10^4.** Usando el método private void fillArray(int[] a)  llenamos el arreglo con datos aleatorios. | Arreglo a ordenar | El arreglo esta ordenado de forma ascendente. Siendo el arreglo **A**, como esta ordenado ascendentemente, es correcto afirmar que la resta **(A[i] – A[i + 1])** tendrá valor negativo o cero |
| Ordenamientos | CombSort(int[]) arr | Arreglo de tamaño **10^4.** Usando el método private void fillArray(int[] a)  llenamos el arreglo con datos aleatorios. | Arreglo a ordenar | El arreglo esta ordenado de forma ascendente. Siendo el arreglo **A**, como esta ordenado ascendentemente, es correcto afirmar que la resta **(A[i] – A[i + 1])** tendrá valor negativo o cero |

* **Diseño de pruebas unitarias**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TestMergeSort** | **Objetivo de la prueba:** Comprobar que el método **MergeSort** ordena el conjunto de datos de manera ascendente    **Firma del método: public void MergeSort(int[] arr)** | | | |
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| Ordenamientos | MergeSort(int[] input) | Arreglo de tamaño **10^4.** Usando el método private void fillArray(int[] a)  llenamos el arreglo con datos aleatorios. | Arreglo a ordenar | El arreglo esta ordenado de forma ascendente. Siendo el arreglo **A**, como esta ordenado ascendentemente, es correcto afirmar que la resta **(A[i] – A[i + 1])** tendrá valor negativo o cero. |
| Ordenamientos | MergeSort(int[] input) | Arreglo de tamaño **10^4.** Usando el método private void fillArray(int[] a)  llenamos el arreglo con datos aleatorios. | Arreglo a ordenar | El arreglo esta ordenado de forma ascendente. Siendo el arreglo **A**, como esta ordenado ascendentemente, es correcto afirmar que la resta **(A[i] – A[i + 1])** tendrá valor negativo o cero |
| Ordenamientos | MergeSort(int[] input) | Arreglo de tamaño **10^4.** Usando el método private void fillArray(int[] a)  llenamos el arreglo con datos aleatorios. | Arreglo a ordenar | El arreglo esta ordenado de forma ascendente. Siendo el arreglo **A**, como esta ordenado ascendentemente, es correcto afirmar que la resta **(A[i] – A[i + 1])** tendrá valor negativo o cero |

* **Diagrama de clases**

**Imagen que contiene captura de pantalla

Descripción generada automáticamente**

Imagen que contiene captura de pantalla

Descripción generada automáticamente

* **Diagrama de Objetos**

Imagen que contiene captura de pantalla

Descripción generada automáticamente

Imagen que contiene captura de pantalla

Descripción generada automáticamente