|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba No. 1 | Objetivo de la prueba:  *Comprobar el funcionamiento del algoritmo de ordenamiento insertionSort(int[]) en el peor caso posible(Mayor a menor) y que devuelva un arreglo con los números ordenados de(Menor a mayor)*  Firma del método:  *insertionSort* | | | |
| Clase para probar | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| SortNumbers | insertionSort(int[]) | setupStage1(): Se creó un nuevo objeto de tipo SortNumbers | Se crea un arreglo de 10 elementos con números en el peor caso posible (Numero están ordenados de mayor a menor) | Se devuelve un arreglo con sus elementos ordenados de menor a mayor |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba No. 2 | Objetivo de la prueba:  *Comprobar el funcionamiento del algoritmo de ordenamiento insertionSortReverse(int[]), el cual recibe un arreglo de numero(Menor a mayor) y que devuelva un arreglo ordenado de mayor a menor*  Firma del método:  *insertionSortReverse* | | | |
| Clase para probar | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| SortNumbers | insertionSortReverse(int[]) | setupStage1(): Se creó un nuevo objeto de tipo SortNumbers | Se crea un arreglo de 10 elementos, el cual contiene números ordenados de menor a mayor | Se devuelve un arreglo con sus elementos ordenados de mayor a menor |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba No. 3 | Objetivo de la prueba:  *Comprobar el funcionamiento del algoritmo de ordenamiento mergeSort (int[], int), el cual recibe un arreglo y el tamaño del arreglo como parámetro y que devuelva un arreglo ordenado de menor a mayor*  Firma del método:  *mergeSort* | | | |
| Clase para probar | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| SortNumbers | mergeSort (int[], int) | setupStage1(): Se creó un nuevo objeto de tipo SortNumbers | Se crea un arreglo de 10 posiciones, el cual tiene números generados aleatoriamente | Se devuelve un arreglo con sus elementos ordenados de menor a mayor |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba No. 4 | Objetivo de la prueba:  *Comprobar el funcionamiento del algoritmo de ordenamiento heapSort (int[]), el cual recibe como parámetro un arreglo en el cual todos sus elementos son generados aleatoriamente, se espera que devuelva un arreglo con sus elementos ordenados de menor a mayor*  Firma del método:  *heapSort* | | | |
| Clase para probar | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| SortNumbers | heapSort (int[]) | setupStage1(): Se creó un nuevo objeto de tipo SortNumbers | Se crea un arreglo de 10 posiciones, el cual contiene números generados aleatoriamente | Se devuelve un arreglo, en el cual sus elementos están ordenados de menor a mayor |