1. Implement Mergesort algorithm; run the program on nine different arrays of real numbers, *Array\_1*, *Array\_2*, …, and *Array\_9*; fill out the following table in an Excel spreadsheet (named “Mergesort\_Time.xls(x)”).

|  |  |  |  |
| --- | --- | --- | --- |
| **Input size *n* for *Array\_i*** | **Value of *n· logn*** | **Time spent (seconds or milliseconds or nanoseconds)** | **Value of (*n· logn*)/time (using scientific notation: x · 10y, with x being a rounded integer)** |
| 1682 | 5425.83931764 | 1011157 | 0.00536597117 |
| 1067 | 3231.05143553 | 237040 | 0.01363082785 |
| 1826 | 5955.50041186 | 296891 | 0.02005955186 |
| 1533 | 4883.43612339 | 541151 | 0.00902416538 |
| 1223 | 3775.92255696 | 253830 | 0.01487579307 |
| 1458 |  | 394738 |  |
| 1403 |  | 375806 |  |
| 1405 |  | 221328 |  |
| 1229 |  | 366727 |  |

* The size of *Array\_i* should be no less than 1000 · *i*, but the actual array sizes are of your choice.
* Elements in each array are also of your choice, and you are highly suggested to use the program to generate random real numbers.
* The Excel spreadsheet can be manually or automatically filled out.

1. Design an interface that allows the user to choose an arbitrary number *i* (the value of *i* is from 1 to 9). The interface will then display the original *Array\_i* followed by the sorted *Array\_i*.
2. The interface will keep waiting for the user selection until the user chooses to exit.