**Algorithms and Data Structures**

**CH08-320201**

**Homework 2**

**Problem 2.1**

**Merge Sort**

1. Please refer to program file
2. Please note that these do not include the plots from HW 1 since my algorithm was incorrect for HW1

|  |  |  |  |
| --- | --- | --- | --- |
| **Input(n)** | **Subseries(k)** | **Case** | **Time** |
| 100 | 10 | Average | 0.001 |
|  |  | Best | 0.001 |
|  |  | Worst | 0.001 |
| 500 | 50 | Average | 0.039 |
|  |  | Best | 0.001 |
|  |  | Worst | 0.075 |
| 1000 | 100 | Average | 0.305 |
|  |  | Best | 0.003 |
|  |  | Worst | 0.597 |
| 2000 | 500 | Average | 2.283 |
|  |  | Best | 0.011 |
|  |  | Worst | 4.428 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Input(n)** | **Subseries(k)** | **Case** | **Time** |
| 100 | 50 | Average | 0.001 |
|  |  | Best | 0.001 |
|  |  | Worst | 0.001 |
| 500 | 100 | Average | 0.04 |
|  |  | Best | 0.01 |
|  |  | Worst | 0.078 |
| 1000 | 500 | Average | 0.217 |
|  |  | Best | 0.001 |
|  |  | Worst | 0.445 |
| 2000 | 1500 | Average | 1.779 |
|  |  | Best | 0.007 |
|  |  | Worst | 3.535 |

1. Insertion sort takes *Θ(k2)* time per *k*-element list in the worst case. Therefore, sorting *n/k* lists of k elements each takes *Θ(k2 n/k)=Θ(nk)* worst-case time. In theory, the merge-insertion sort algorithm improves on the overall time and achieves *Θ(n log(n/k))* by merging lists, and then merging pairwise lists until there is just one list. A pairwise merging would require *Θ(n)* work at each sub-level.

In theory this algorithm works on *n* elements and even if they are partitioned in different subsequences, it finishes with 1 sorted list with *n* elements which amounts to *log(n/k)* levels. Therefore, in theory and possibly not my code, the total time required for sorting becomes *Θ(nlog(n/k)*

1. The value for *k* should be the largest length of integers on which insertion sort is faster than merge sort.