B04901162 林皓謙

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Input size | IS | | MS | | HS | |
|  | CPU time  (s) | Memory  (KB) | CPU time  (s) | Memory  (KB) | CPU time  (s) | Memory  (KB) |
| 4000.bc | 0.000077 | 13284 | 0.000153 | 13284 | 0.000205 | 13284 |
| 4000.wc | 0.005357 | 13284 | 0.000207 | 13284 | 0.000225 | 13284 |
| 4000.ac | 0.001941 | 13284 | 0.000372 | 13284 | 0.00034 | 13284 |
| 16000.bc | 0.000088 | 13432 | 0.000488 | 13432 | 0.000864 | 13432 |
| 16000.wc | 0.083658 | 13432 | 0.000904 | 13432 | 0.000884 | 13432 |
| 16000.ac | 0.043941 | 13432 | 0.002145 | 13432 | 0.002377 | 13432 |
| 32000.bc | 0.000226 | 13432 | 0.000952 | 13436 | 0.001911 | 13432 |
| 32000.wc | 0.304047 | 13432 | 0.00117 | 13436 | 0.001907 | 13432 |
| 32000.ac | 0.171376 | 13432 | 0.002844 | 13446 | 0.002131 | 13432 |
| 1000000.bc | 0.001387 | 19456 | 0.035143 | 21192 | 0.082954 | 19456 |
| 1000000.wc | 260.180 | 19456 | 0.039822 | 21192 | 0.079941 | 19456 |
| 1000000.ac | 153.642 | 19456 | 0.105422 | 21192 | 0.143788 | 19456 |

In terms of speed, both Heap Sort and Merge Sort seem to require similar CPU time to sort input files of various sizes. This is due to the fact that their worst and average case sorting times are all to the nlgn relation. From what was observed, both of them were able to sort efficiently even for inputs of very big sizes. On the other hand, Insertion sort also performed quite well(still worse compared to MS and HS) when there were less input, however the CPU time increased drastically with the input size. This is as a result of the n^2 sorting rate of Insertion sort. Both the average case and worst case sorts less efficiently than MS and HS. Yet, it is interesting to note that IS performs more efficiently than MS and HS in all best-case input scenarios. IS is good for a list that is almost sorted since it doesn’t have to go through the “while loop” which compares if the value in previous indexes are higher than the key. This limits IS to an “n” rate in time complexity.

In terms of memory, IS and HS performed identically; moreover, they outperformed MS. This is mainly due to the fact that IS and HS are both in place, thus requiring only the original set of arrays for data storage; which also explains why IS and HS require identical amounts of memory.

**bc**