Algorithm

Programming Assignment 1

Name: 吳睿哲

Student ID: r06921095

* **Comparison of performance**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | IS | IS | MS | MS | QS | QS | HS | HS |
|  | CPU time (s) | Memory (KB) | CPU time (s) | Memory (KB) | CPU time (s) | Memory (KB) | CPU time (s) | Memory (KB) |
| 4000.case2 | 0 | 12500 | 0.001 | 12500 | 0.001 | 12500 | 0 | 12500 |
| 4000.case3 | 0.015998 | 12500 | 0.001 | 12500 | 0 | 12500 | 0 | 12500 |
| 4000.case1 | 0.004999 | 12500 | 0.001 | 12500 | 0.006999 | 12500 | 0 | 12500 |
| 16000.case2 | 0 | 12648 | 0.001999 | 12648 | 0.001 | 12648 | 0.001999 | 12648 |
| 16000.case3 | 0.215967 | 12648 | 0.001999 | 12648 | 0.007998 | 12648 | 0.001 | 12648 |
| 16000.case1 | 0.101985 | 12648 | 0.006 | 12648 | 0.011999 | 12648 | 0.003 | 12648 |
| 32000.case2 | 0 | 12648 | 0.007999 | 12648 | 0.003999 | 12648 | 0.012998 | 12648 |
| 32000.case3 | 0.770883 | 12648 | 0.008999 | 12648 | 0.004998 | 12648 | 0.003998 | 12648 |
| 32000.case1 | 0.489925 | 12648 | 0.011998 | 12648 | 0.006999 | 12648 | 0.010998 | 12648 |
| 1000000.case2 | 0.002 | 18668 | 0.170974 | 20524 | 0.07998 | 18668 | 0.170974 | 18668 |
| 1000000.case3 | 570.957 | 18668 | 0.175972 | 20524 | 0.078988 | 18668 | 0.113983 | 18668 |
| 1000000.case1 | 285.260 | 18668 | 0.253961 | 20524 | 0.144978 | 18668 | 0.305954 | 18668 |

* **Analyzation of execution time**

|  |
| --- |
|  |
|  |
|  |

**Case1**:

In case 1, the input data is distributed averagely between minimum and maximum value. Thus, it can be observed that the execution time mainly related to the time complexity of the algorithm in average case. The time complexity of Insertion sort in average case is O(), while the others are

O(n log n). The bigger the input size, the more obvious the gap.

**Case2**:

In case 2, the input data are in ascent order. Thus, in this case, it requires few operations for insertion sort. Even the time complexity for the other algorithms are lower, the execution time grows larger when the input size becomes larger. Note that the quick sort is implemented using randomized partition. Thus, the complexity is still O(n log n). If normal partition method is utilized, the time complexity will be O().

**Case3**:

In case 3, the input data are in descent order. Thus, it becomes the worst case for insertion sort. The execution time grows significantly with large input size. For other algorithms, due to the fact that the average complexity are all O(n log n). Thus, the growth of the execution time may not be so obvious.