CSCE 221 Cover Page Programming Assignment #3

First Name: SEUNGJIN Last Name: KIM UIN:525006913

Please list all below all sources (people, books, webpages, etc) consulted regarding this assignment:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CSCE 221 Students | Other People | Printed Material | Web Material (URL) | Other |
| 1. | 1. | 1. | 1. | 1. |
| 2. | 2. | 2. | 2. | 2. |
| 3. | 3. | 3. | 3. | 3. |
| 4. | 4. | 4. | 4. | 4. |
| 5. | 5. | 5. | 5. | 5. |

Recall that University Regulations, Section 42, define scholastic dishonesty to include acquiring answers from any unauthorized source, working with another person when not specifically permitted, observing the work of other students during any exam, providing answers when not specifically authorized to do so, informing any person of the contents of an exam prior to the exam, and failing to credit sources used.

Disciplinary actions range from grade penalties to expulsion. Please consult the Aggie Honor System Office for additional information regarding academic misconduct – it is your responsibility to understand what constitutes academic misconduct and to ensure that you do not commit it.

I certify that I have listed above all the sources that I consulted regarding this assignment, and that I have not received nor given any assistance that is contrary to the letter or the spirit of the collaboration guidelines for this assignment.

Today’s Date: 2018.4.24

Printed Name (in lieu of a signature): seungjin kim

1. *Introduction.*

The objective of this assignment is to implement three different type of sort; Selection Sort, Heap Sort and Merge sort. After done with implementing these three different sorting functions, it needs to be analyzed and compared the time complexity by measuring each time taken for these different functions to sort numbers which is placed in ordered, reversely ordered or randomly.

1. *Theoretical Analysis*
   1. The time complexity of sorting already sorted numbers for each functions.

Selection Sort: O(n^2)

Heap Sort: O(n log n)

Merge Sort: O(n log n)

* 1. The time complexity of sorting reversely sorted numbers for each functions.

Selection Sort: O (n^2)

Heap Sort: O(n log n)

Merge Sort: O(n log n)

* 1. The time complexity of sorting randomly generated numbers for each function.

Selection Sort: O(n^2)

Heap Sort: O(n log n)

Merge Sort: O(n log n)

1. Experimental Setup
   1. Machine specification

Processor : 2.6 GHz Intel Core i5

Memory: 8 GM 1867 MHz DDR3

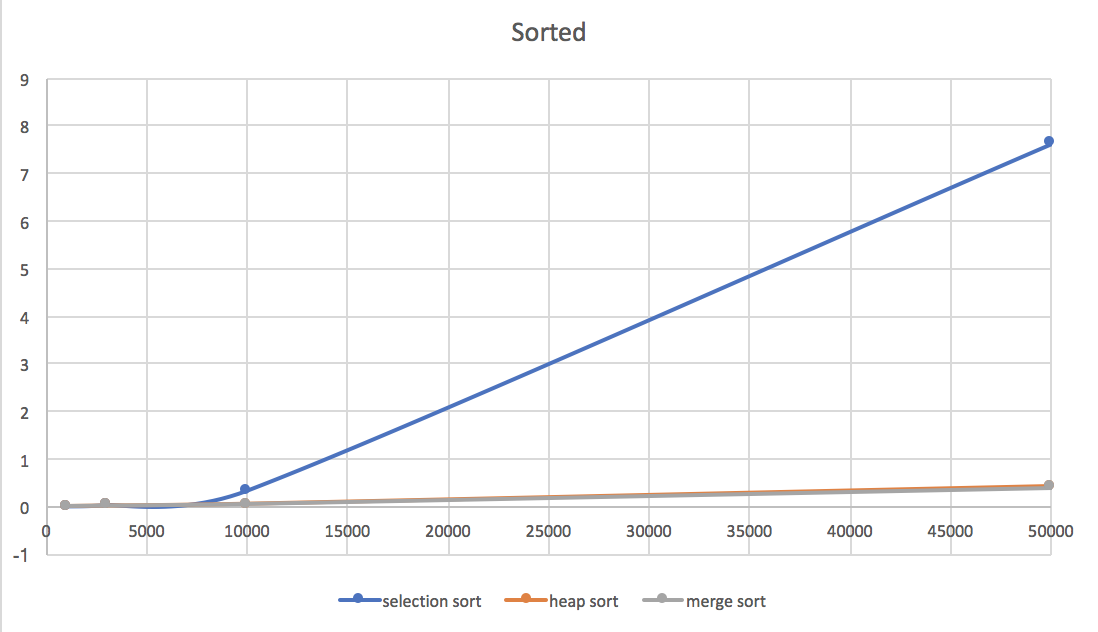
Graphics Intel Iris Graphics 6100 1536 MB

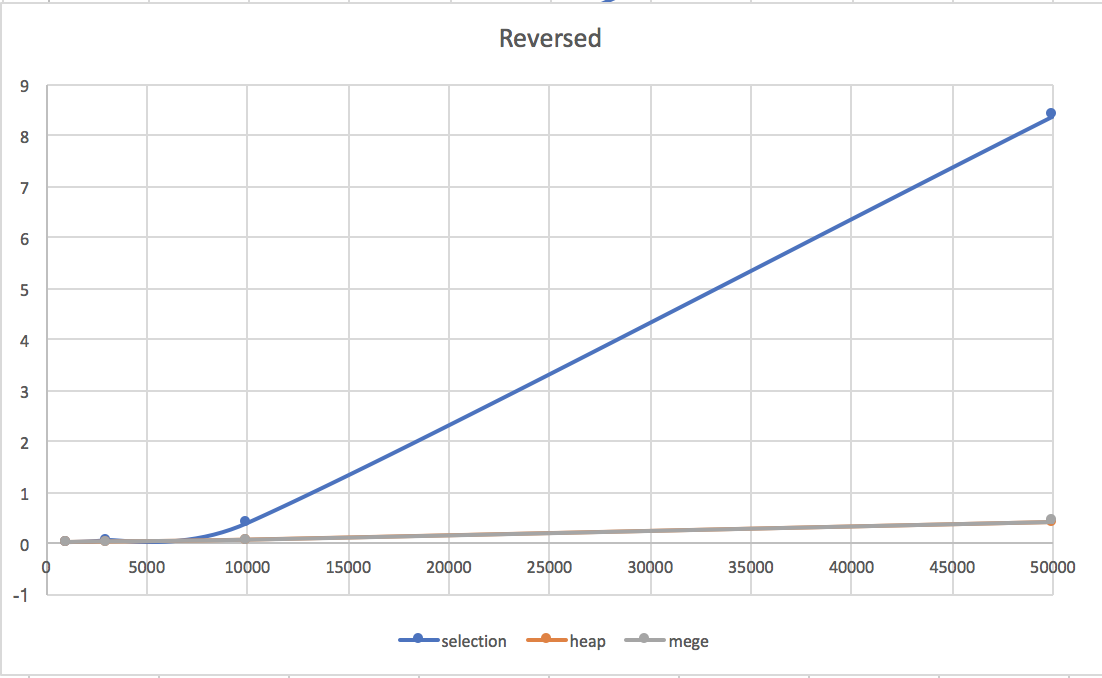
* 1. How did you generate the test input? What input sized did you test? Why?

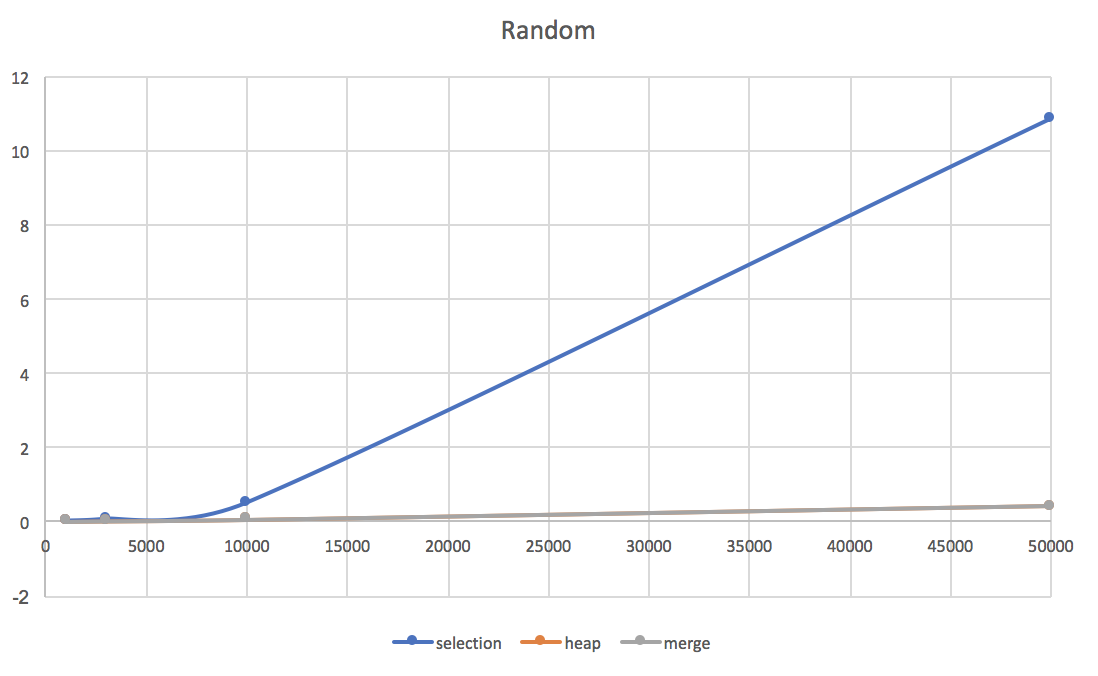
The test input file was generated using a C++ function which takes the type of sorting function that will be used, number of input elements, type of input that indicates if the number will be generated in order, reverse order or randomly and file name; based on these parameter, the function generates required inputs. The input sizes that are used for test are 1000, 3000, 10000, 50000. The Input sizes are decided to make noticeable difference of time taken for each functions to sort given numbers.

* 1. How many time did you repeat each experiment?

Each experiment repeated ten times to make an accurate consequence.

1. Experimental Results.
   1. 





* 1. It seems like heap sort and merge sort performs similarly, but merge sort performs slightly better than heap sort. It doesn’t depend on the input. The result agrees with theoretical analysis except for that the time complexity of sort for selection sort is O(n^2) no matter how the inputs are placed. However, selection sort takes more time to sort randomly placed numbers then when it does sorted or reversely sorted numbers.