Algorithms Practical 4 - Ruth Dooley 19300753 - Answers

Quick Questions

1. How many comparisons does insertion sort make on an input array that is already sorted?

Linear

2. What is the stable sorting algorithm?

A stable sorting algorithm ensures that if items are equal to each other, that they aren't reordered in the sorting process. If so the sorting algorithm would be considered unstable

3. What is an external sorting algorithm?

A. Algorithm that uses tape or disk during the sort.

4. Identify 6 ways of characterising a sorting algorithm?

- Does the sorting algorithm compare? A comparison type sort looks at two elements at a time whereas non-comparison sort do not compare two items at a time
- The time complexity? How much time is required to complete the tasks based on the size of the input
- Space complexity? How much memory is required for the algorithm to complete the task based on the size of the input
- Stability? Stable or non stable sorting. Does the sorting algorithm preserve the existing relative order of elements when comparing equal keys
- Internal or external? Is the sorting contained in the main memory or RAM or externally sorted where external memory is used
- Recursive or non-recursive? Recursive implements the divide or conquer approach splitting up the dataset into smaller tasks and calls itself over and over. Non - recursive does not use this technique to split the dataset up

Sorting Algorithm Results

* Time in nanoseconds
* Numbers to be sorted range from 0 - 100
* In each of the cells in the sorting algorithm columns in the below table there will be three results. Each of these represent the time taken for the task to be completed given the random input

No. of Ints	SelectionSort	InsertionSort	BogoSort
1	529250 622100 556513	419189 614685 870662	417743 584443 988413
2	564267 407294 522845	507228 601548 534546	474438 500725 501301
3	554792 811898 476946	597846 724158 523708	504625 998062 524939
5	515635 492971 744983	522392 451943 614579	Continued to break at this point and onward ...
10	508988 449414 536747	473779 458977 590890	

50	604038 574156 566151	530340 434079 464236	
100	574010 635075 677512	419242 390165 495111	
500	3471809 3447591 3287476	1698036 1648206 1929661	
1,000	6569229 6155473 9083569	3358555 2551712 3981533	
5,000	38672833 36536099 35361803	8195758 8837918 9310658	
10,000	123119910 122056806 124664826	16708585 16188238 16694405	
20,000	465137993 458091812 468754217	49150921 46376791 47213502	
25,000	728762428 736085349 746206794	72599616 71865391 73731513	

* After 25,000 elapsed time could not be calculated correctly and consistently

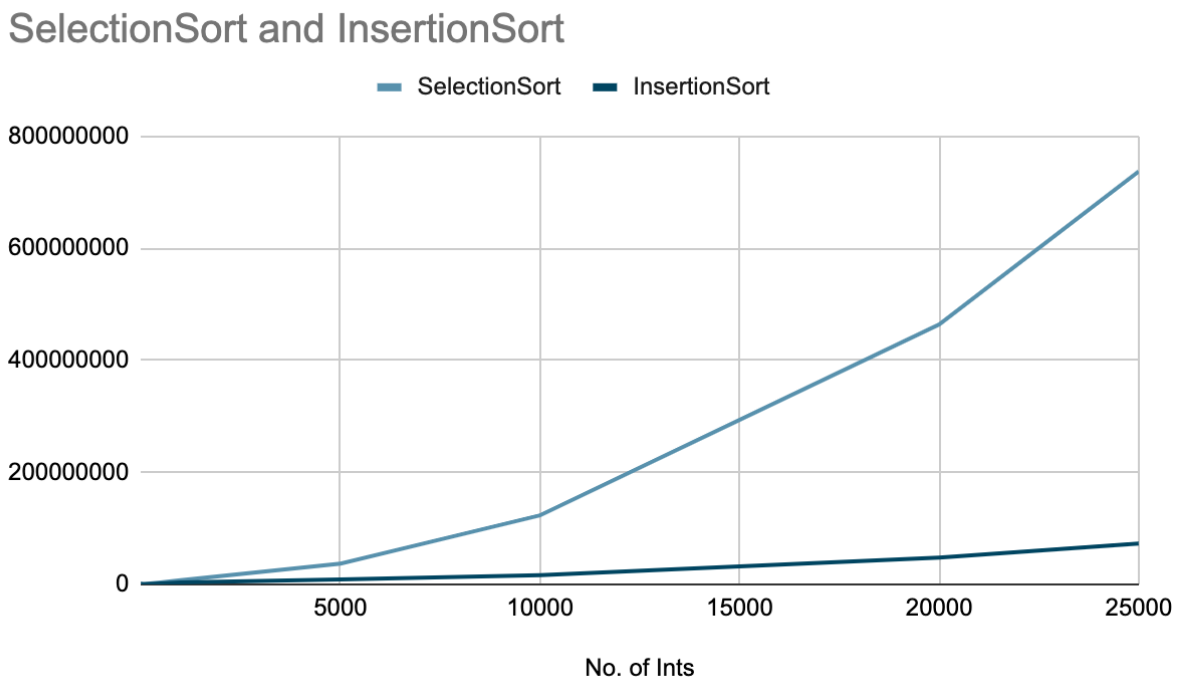
Average Times

* Given the results above the average elapsed time for the task to be completed is calculated

No. of Ints	SelectionSort	InsertionSort	BogoSort
1	569287.66666667	634845.33333333	663533
2	498135.33333333	547774	492154.66666667
3	614545.33333333	615237.33333333	675875.33333333
5	584529.66666667	529638	NA
10	498383	507882	NA
50	581448.33333333	476218.33333333	NA
100	628865.66666667	434839.33333333	NA
500	3402292	1758634.3333333	NA
1,000	7269423.6666667	3297266.6666667	NA

5,000	36856911.666667	8781444.6666667	NA
10,000	123280514	16530409.333333	NA
20,000	463994674	47580404.666667	NA
25,000	737018190.33333	72732173.333333	NA

Selection Sort and Insertion Sort vs Number of Ints



Selection Sort O(n2): Selection sort took the most time consistently as visualised in the graph above. Selection sorts' big O notation for this algorithm is o(n2) as the average slope for this graph is ½.

Insertion Sort O(n^2): As the nature of my input data was integers between the range of 0-100 the insertion sort worked better. Insertion sorts' big O notation is O(n^2)

Bogo Sort O(n*n!)(On average): Best case scenario → O(n): Worst case scenario → O(∞): This an unpredictable sorting algorithm as shown in the data collected above. I could not get the algorithm working for a data set longer than 3, even then the algorithm was working inconsistently. Therefore the bogoSort is omitted from the graph above.