INFO 6205

PROGRAM STRUCTURES AND ALGORITHMS

FALL 2018

ASSIGNMENT 3

1. CONCLUSION:

- For Selection Sort with any of the worst or average cases that is for randomly ordered, reverse-ordered or partially-ordered or even sorted the time taken for N elements to be sorted is the same that is time complexity T(N)~c*(N^2)/2
- Insertion Sort works best for sorted array followed by partially ordered array followed by random ordered array but for reverse ordered array it tends to be worse than the selection sort due the constant "k" in the time time complexity equation, i.es T(N)~ k*(N^2)/2.

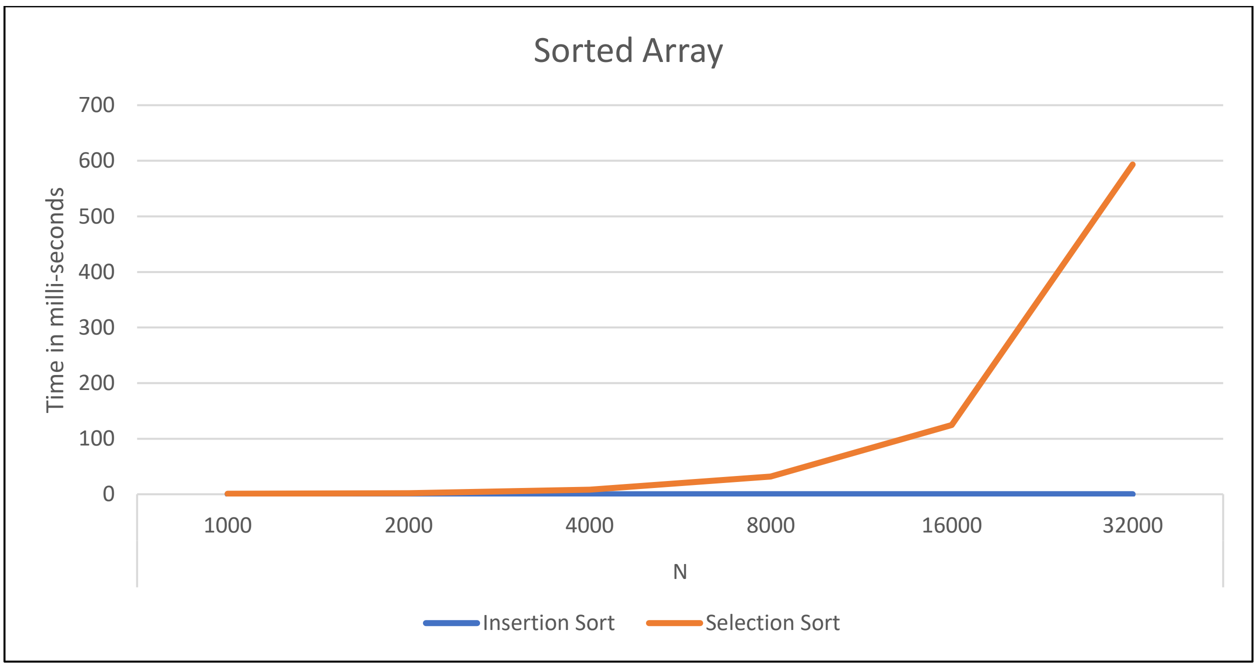
  Where k>c.

2. GRAPHICAL REPRESENTATION

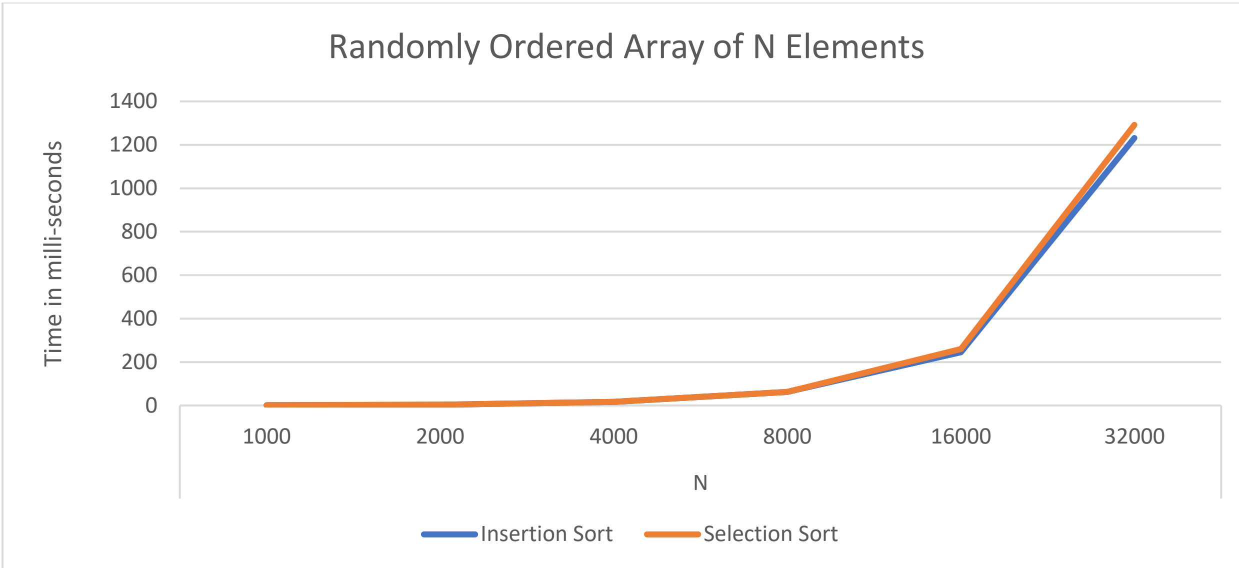
	N=1000	N=2000	N=4000	N=8000	N=16000	N=32000
Insertion (Sorted Order)	0.05328	0.1024	0.0182	0.03454	0.0864	0.1256
Insertion (Random Order)	1.506	4.2241	16.5939	62.75	245.45	1231.58
Insertion (Partially ordered)	0.23137	0.5189	2.0359	7.8869	32.413	131.23
Insertion (Reverse Ordered)	2.0607	8.1327	31.4487	122.80	493.2811	2076.211
Selection (Sorted Order)	0.918	2.046	7.988	31.76	124.24	593.25

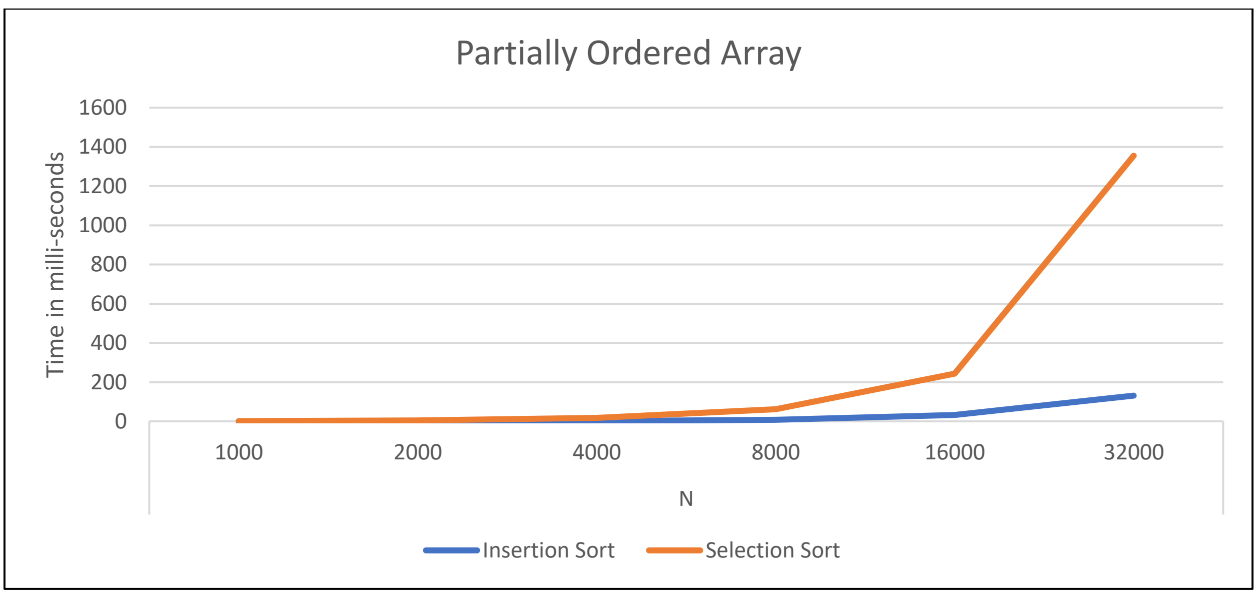
Jaswal, Rajat: NUID-001443168

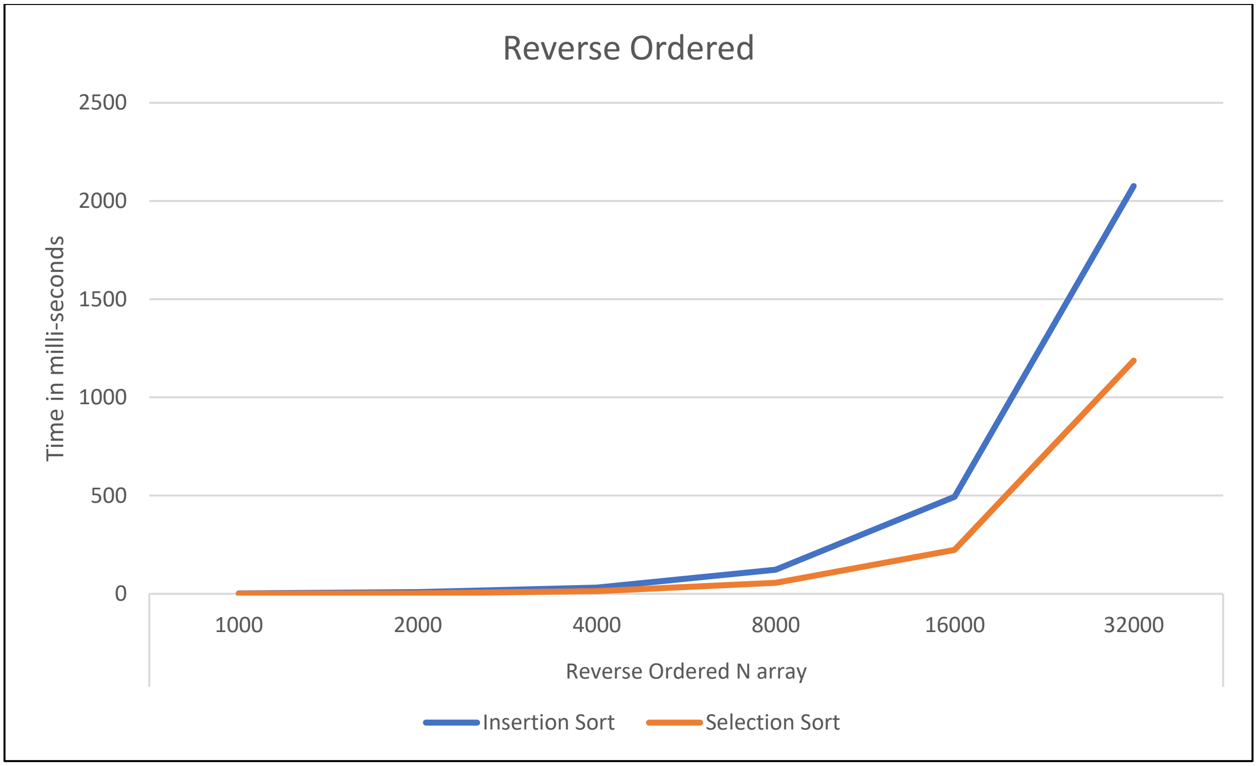
Selection (Random Order)	1.188	4.1474	17.1904	61.91	259.57	1291.87
Selection (Partially Ordered)	1.4188	4.02358	18.533	62.0533	243.85	1355.51
Selection (Reverse ordered)	0.9085	3.022	14.2851	56.1367	224.48	1187.44



Jaswal, Rajat: NUID-001443168







PROOF:

Let's consider the values from the graph for N=8000 for

a) **For Partially Sorted array –**

   Insertion Sort = 7.8869milliseconds.

   Since we know Insertion sort takes ~cN time that is linear time to sort for partially sorted array and we can also depict it from the graph.
   for N=8000

   T(insertion)=cN (Let's assume)

   7.8869*10^-3/8000=c

   c=0.9858*10^-6

Now let’s prove for

T(insertion) for N=1600

32.413*10^-3/16000=c(new)

C(new)= 2.02*10^-6

Since c(new)~c as there are many computational operations that are being performed and the timing also depends on processor hence Time taken by Insertion sort to sort the partially sorted array is comparatively very less as compared to selection sort and we can also depict that clearly from the graph.

b) For Reverse Sorted array-

Insertion Sort = 122.8031 milli seconds.

The insertion sort in reverse sorted array is the perfect example of a worst-case scenario for the sort to occur when one uses insertion sort.

This is when it hits N^2/2 complexity for reverse.

Since for average/random case, the time is 62.75 milli seconds and the time is proportional to N^2/4 it’s clearly visible that –

T(insertion for reverse sorted array)/T(insertion for random array)=2

Hence time complexity for reverse sorted array for insertion sort is the worst

Selection Sort = 56.1367 milli seconds.

The selection sort works the same as in all the cases for the reverse sorted too.

As we compare with selection sort for random as well as partially sorted , it is almost the same.

3. OBSERVATIONS

The time for larger value of N>16000 variables initializing took much time since they have to be initialized again and again. Hence by taking the variable initialization out of the loops I was able to reduce the time by a significant number almost 1/10th times.