Runtime analysis of sorting algorithms

By

Vikrant

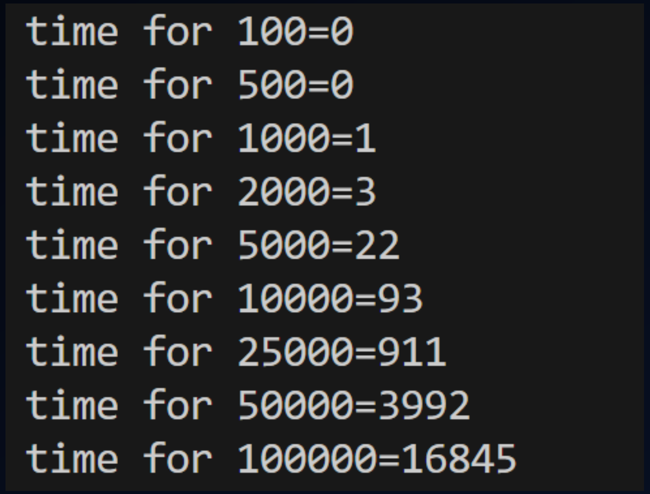
Roll.no: 64

Code for Analysis

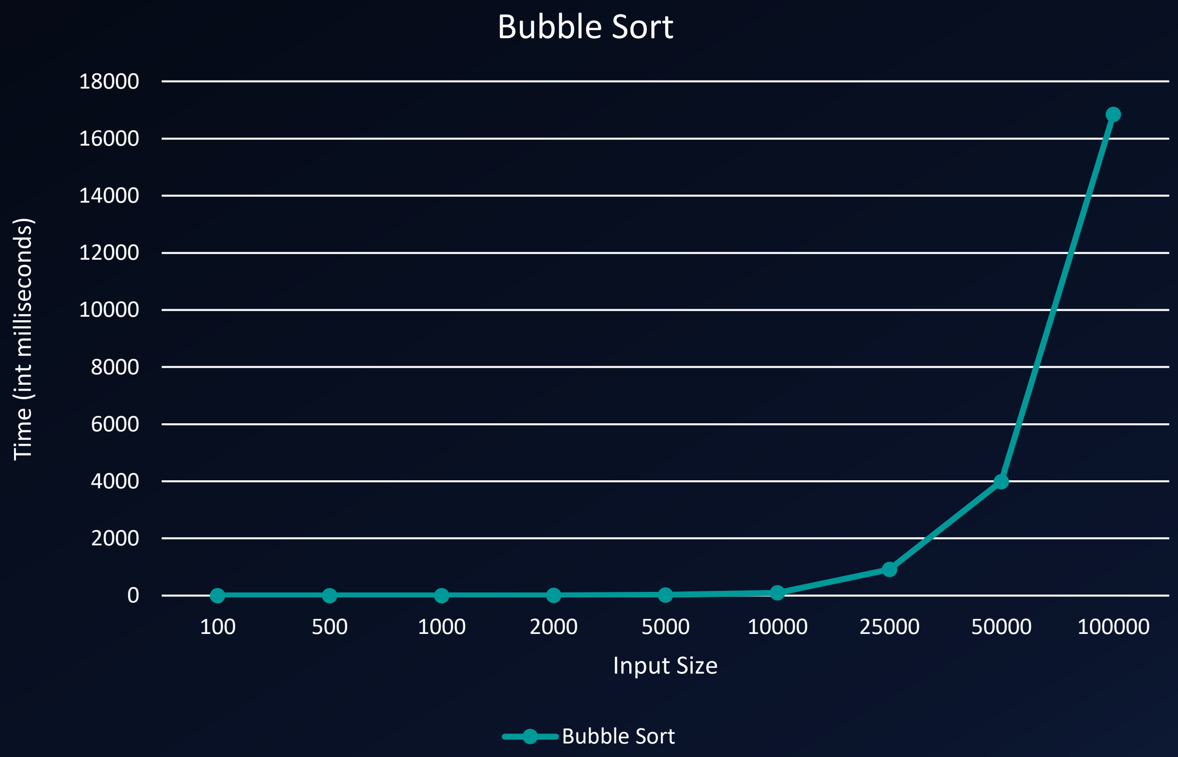
```
auto before=high_resolution_clock::now();
SORT(arr,0,n);
auto after=high_resolution_clock::now();
auto duration = duration_cast<milliseconds>(after-
before);
cout<<"time for"<<n<<"="<<duration.count()<<endl;
```

Bubble Sort

- Worst Case : O(n^2)
- Best Case : O(n^2)

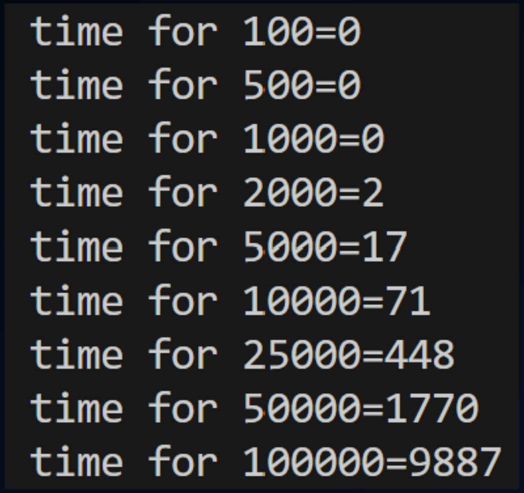


```
time for 100=0
time for 500=0
time for 1000=1
time for 2000=3
time for 5000=22
time for 10000=93
time for 25000=911
time for 50000=3992
time for 100000=16845
```

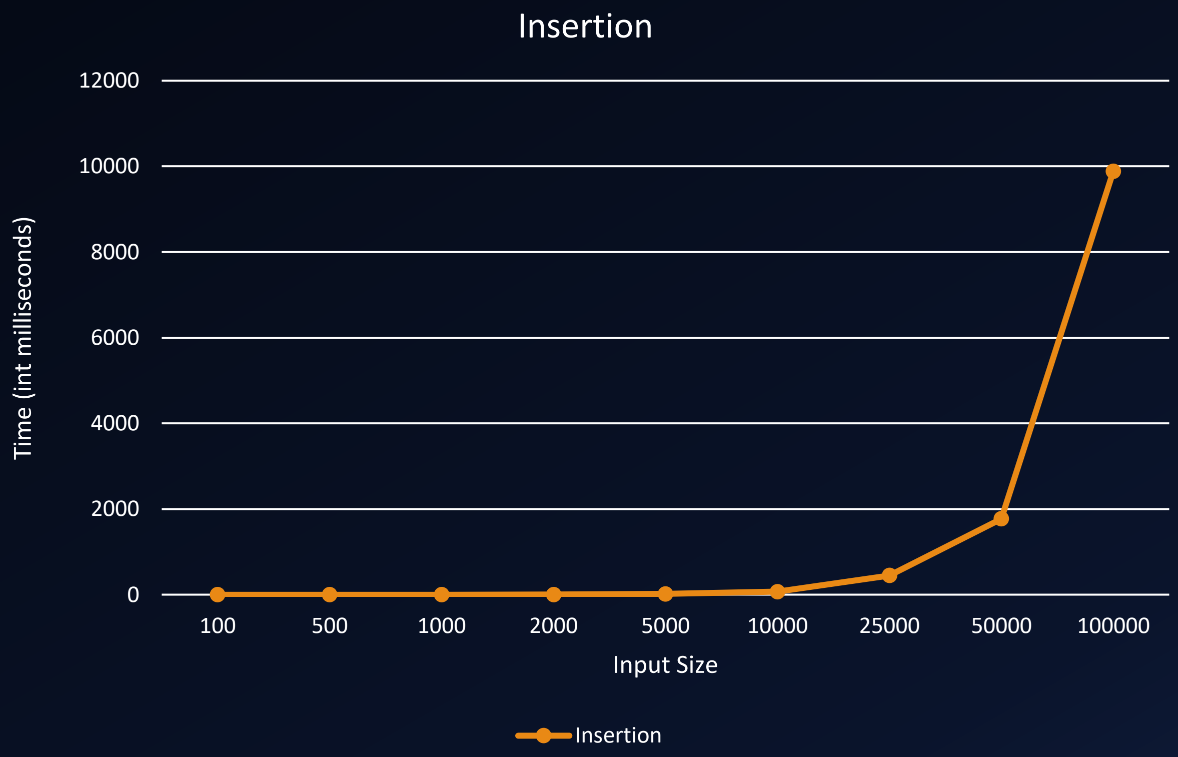


Insetion Sort

- Best Case : O(n)
- Worst Case : O(n^2)

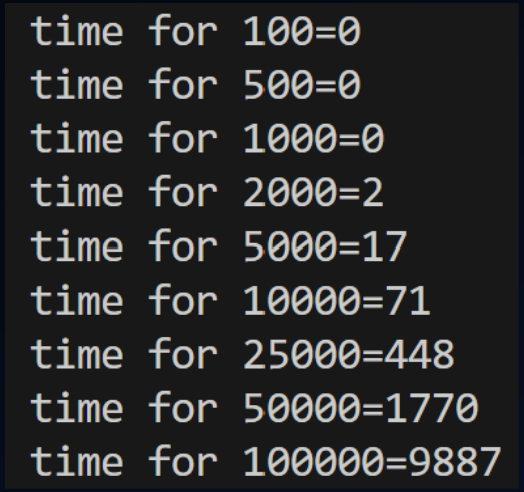


```
time for 100=0
time for 500=0
time for 1000=0
time for 2000=2
time for 5000=17
time for 10000=71
time for 25000=448
time for 50000=1770
time for 100000=9887
```

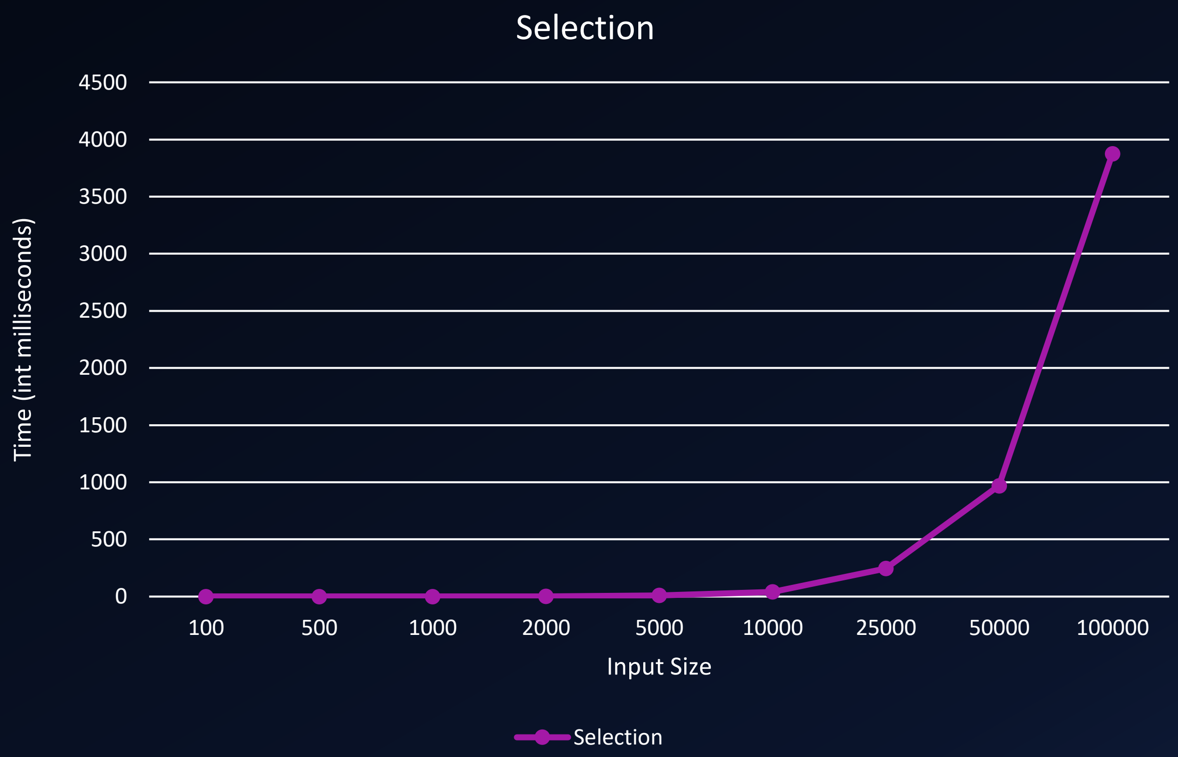


Selection Sort

- Best Case : O(n^2)
- Worst Case : O(n^2)

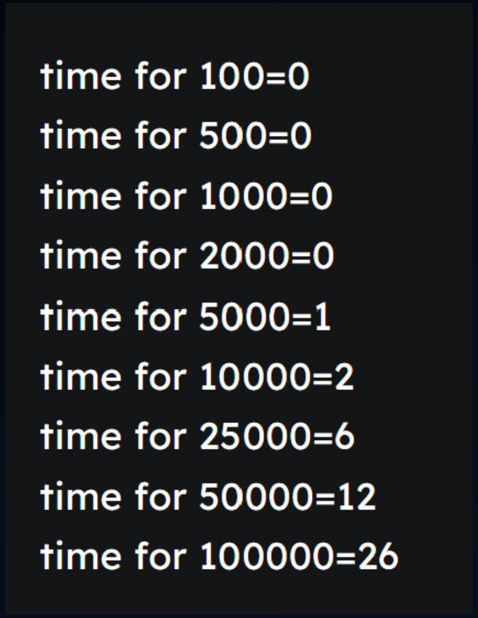


```
time for 100=0
time for 500=0
time for 1000=0
time for 2000=2
time for 5000=17
time for 10000=71
time for 25000=448
time for 50000=1770
time for 100000=9887
```

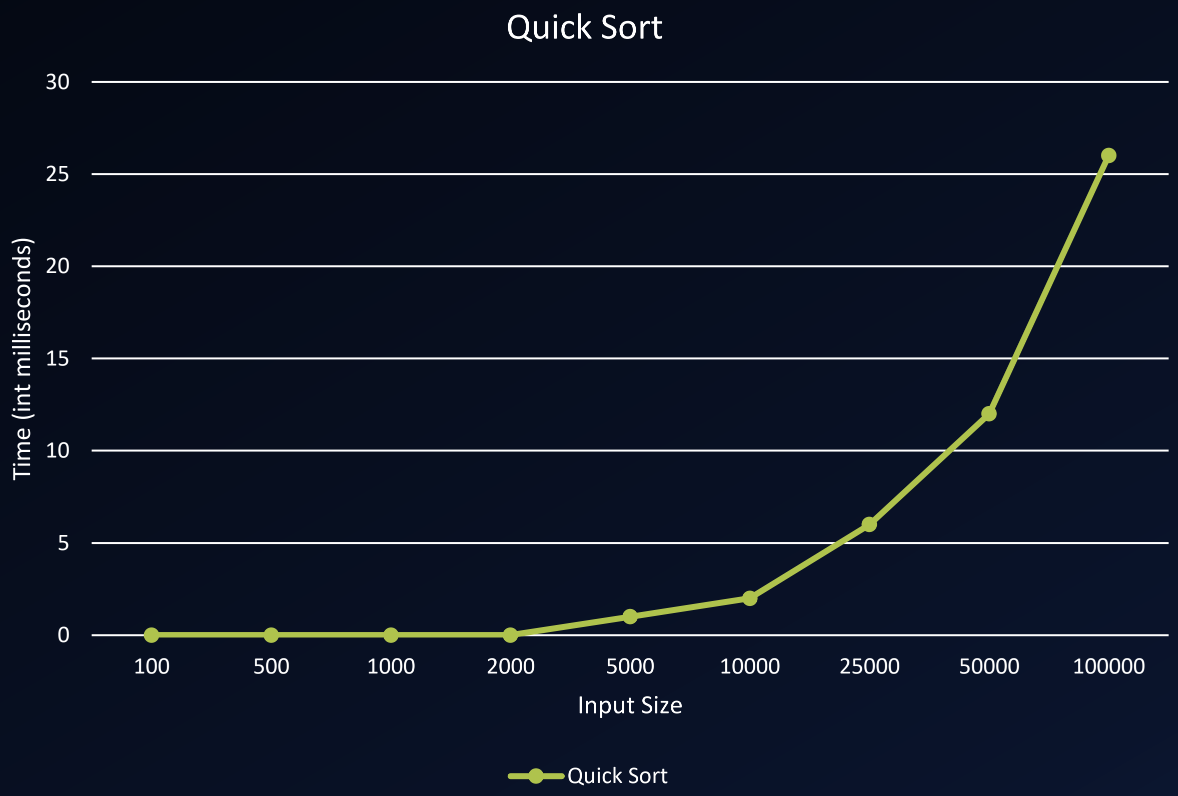


Quick Sort

- Best Case : O(n* Log(n))
- Worst Case : O(n^2)

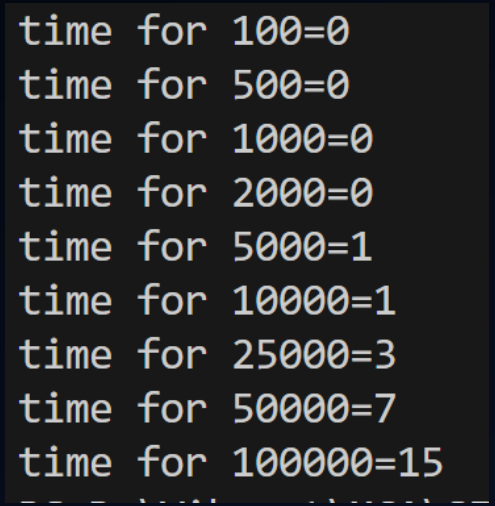


```
time for 100=0
time for 500=0
time for 1000=0
time for 2000=0
time for 5000=1
time for 10000=2
time for 25000=6
time for 50000=12
time for 100000=26
```

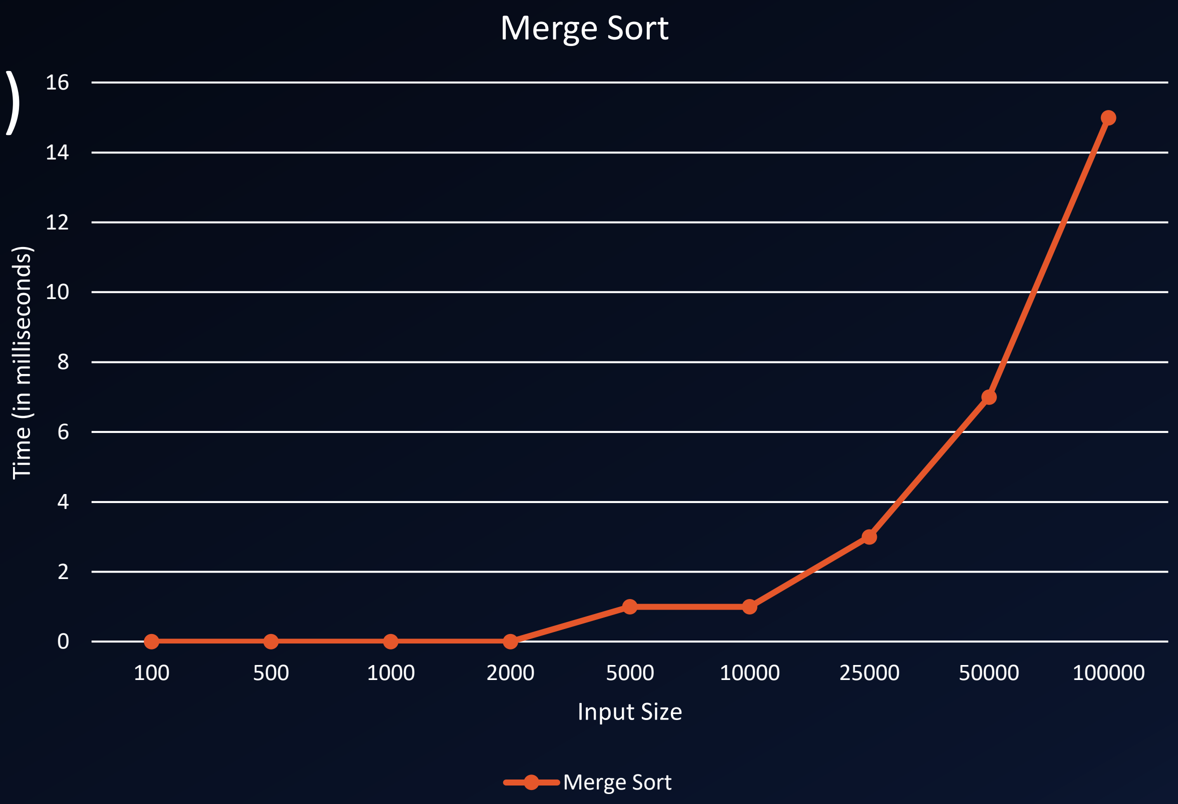


Merge Sort

- Best Case : O(n*Log(n))
- Worst Case : O(n*Log(n))

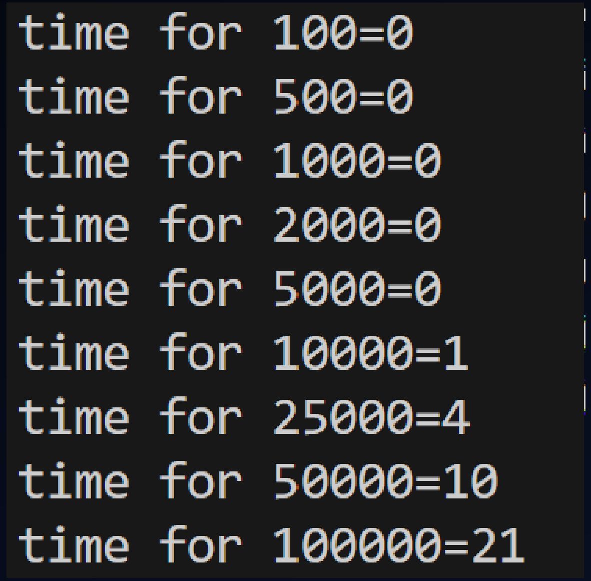


```
time for 100=0
time for 500=0
time for 1000=0
time for 2000=0
time for 5000=1
time for 10000=1
time for 25000=3
time for 50000=7
time for 100000=15
```

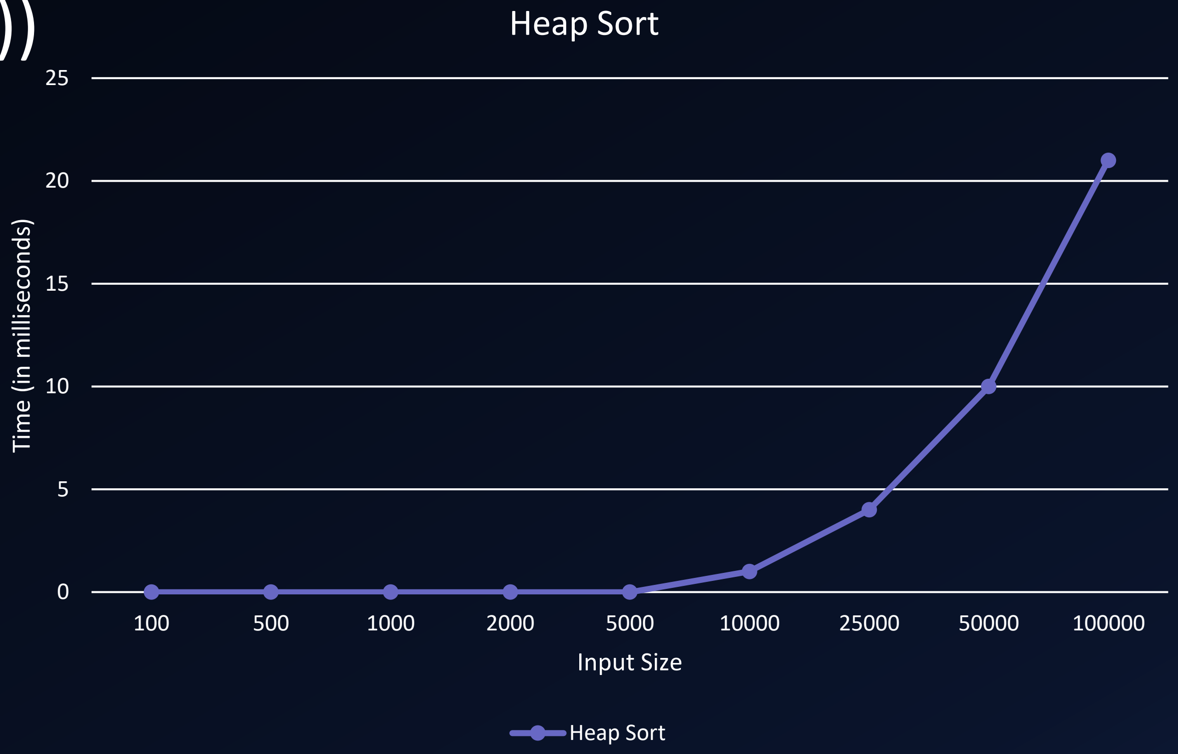


Heap Sort

- Best Case : O(n*Log(n))
- Worst Case : O(n*Log(n))



```
time for 100=0
time for 500=0
time for 1000=0
time for 2000=0
time for 5000=0
time for 10000=1
time for 25000=4
time for 50000=10
time for 100000=21
```



INPUT SIZE VS TIME(in milliseconds)

	100	500	1000	2000	5000	10000	25000	50000	100000
Bubble Sort	0	0	1	3	22	93	911	3992	16845
Insertion	0	0	0	2	17	71	448	1770	9887
Selection	0	0	0	1	10	41	245	970	3875
Quick Sort	0	0	0	0	1	2	6	12	26
Merge Sort	0	0	0	0	1	1	3	7	15
Heap Sort	0	0	0	0	0	1	4	10	21

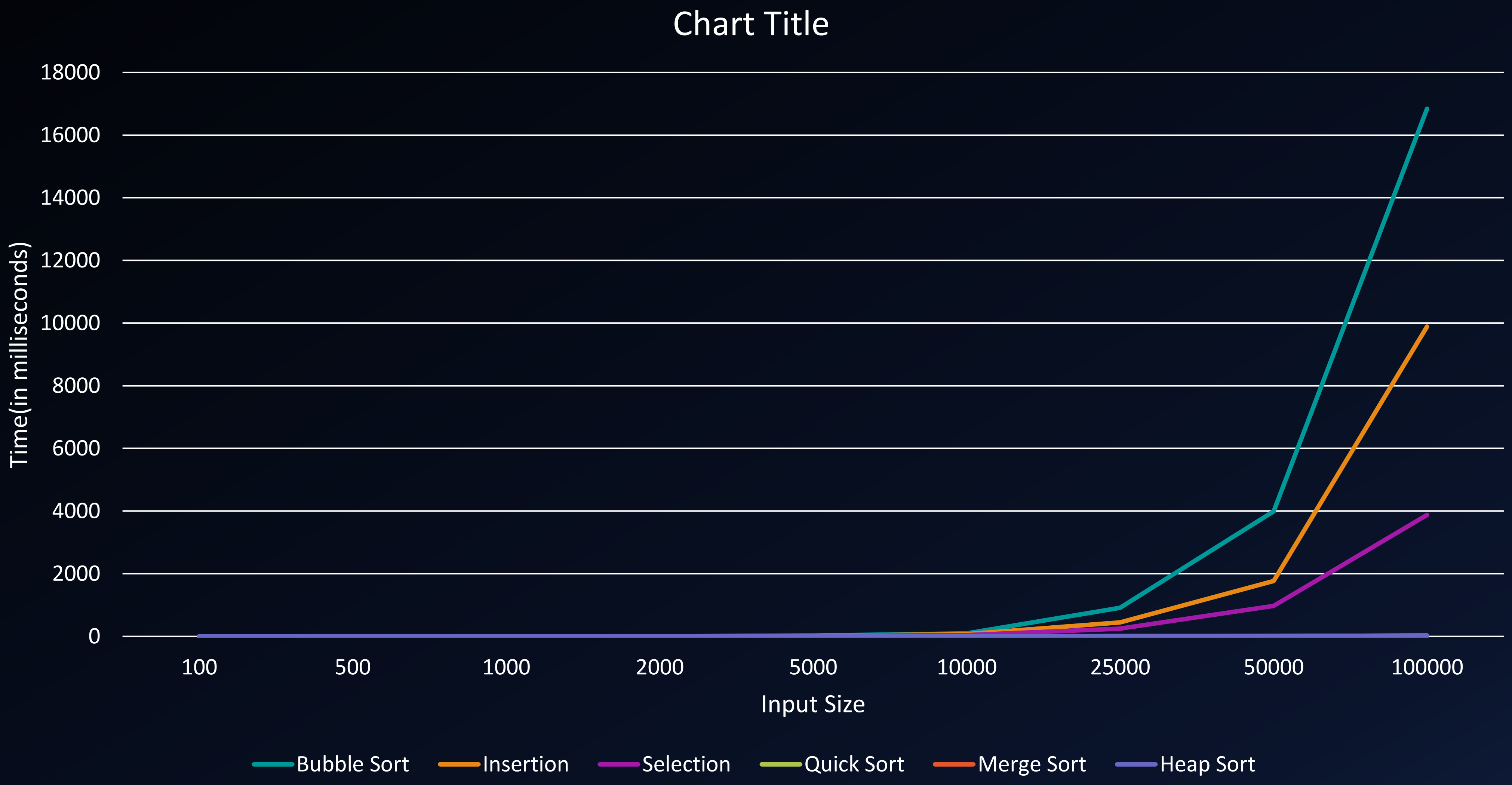


Chart Title
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
Time(in milliseconds)
100
500
1000
2000
5000
10000
25000
50000
100000
Input Size
Bubble Sort
Insertion
Selection
Quick Sort
Merge Sort
Heap Sort